axiom™

The 30 Year Horizon

Manuel Bronstein *William Burge* *Timothy Daly*
James Davenport *Michael Dewar* *Martin Dunstan*
Albrecht Fortenbacher *Patrizia Gianni* *Johannes Grabmeier*
Jocelyn Guidry *Richard Jenks* *Larry Lambe*
Michael Monagan *Scott Morrison* *William Sit*
Jonathan Steinbach *Robert Sutor* *Barry Trager*
Stephen Watt *Jim Wen* *Clifton Williamson*

Volume 11: Axiom Browser

Portions Copyright (c) 2005 Timothy Daly

The Blue Bayou image Copyright (c) 2004 Jocelyn Guidry

Portions Copyright (c) 2004 Martin Dunstan
Portions Copyright (c) 2007 Alfredo Portes
Portions Copyright (c) 2007 Arthur Ralfs
Portions Copyright (c) 2005 Timothy Daly

Portions Copyright (c) 1991-2002,
The Numerical ALgorithms Group Ltd.
All rights reserved.

This book and the Axiom software is licensed as follows:

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are
met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- Neither the name of The Numerical ALgorithms Group Ltd. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Inclusion of names in the list of credits is based on historical information and is as accurate as possible. Inclusion of names does not in any way imply an endorsement but represents historical influence on Axiom development.

Michael Albaugh	Cyril Alberga	Roy Adler
Christian Aistleitner	Richard Anderson	George Andrews
S.J. Atkins	Jeremy Avigad	Henry Baker
Martin Baker	Stephen Balzac	Yurij Baransky
David R. Barton	Thomas Baruchel	Gerald Baumgartner
Gilbert Baumslag	Michael Becker	Nelson H. F. Beebe
Jay Belanger	David Bindel	Fred Blair
Vladimir Bondarenko	Mark Botch	Raoul Bourquin
Alexandre Bouyer	Karen Braman	Wolfgang Brehm
Peter A. Broadbery	Martin Brock	Manuel Bronstein
Christopher Brown	Stephen Buchwald	Florian Bundschuh
Luanne Burns	William Burge	Ralph Byers
Quentin Carpent	Pierre Casteran	Robert Cavines
Bruce Char	Ondrej Certik	Tzu-Yi Chen
Bobby Cheng	Cheekai Chin	David V. Chudnovsky
Gregory V. Chudnovsky	Mark Clements	James Cloos
Jia Zhao Cong	Josh Cohen	Christophe Conil
Don Coppersmith	George Corliss	Robert Corless
Gary Cornell	Meino Cramer	Jeremy Du Croz
David Cyganski	Nathaniel Daly	Timothy Daly Sr.
Timothy Daly Jr.	James H. Davenport	David Day
James Demmel	Didier Deshommes	Michael Dewar
Inderjit Dhillon	Jack Dongarra	Jean Della Dora
Gabriel Dos Reis	Claire DiCrescendo	Sam Dooley
Zlatko Drmac	Lionel Ducos	Iain Duff
Lee Duhem	Martin Dunstan	Brian Dupee
Dominique Duval	Robert Edwards	Heow Eide-Goodman
Lars Erickson	Mark Fahey	Richard Fateman
Bertfried Fauser	Stuart Feldman	John Fletcher
Brian Ford	Albrecht Fortenbacher	George Frances
Constantine Frangos	Timothy Freeman	Korrinn Fu
Marc Gaetano	Rudiger Gebauer	Van de Geijn
Kathy Gerber	Patricia Gianni	Gustavo Goertkin
Samantha Goldrich	Holger Gollan	Teresa Gomez-Diaz
Laureano Gonzalez-Vega	Stephen Gortler	Johannes Grabmeier
Matt Grayson	Klaus Ebbe Grue	James Griesmer
Vladimir Grinberg	Oswald Gschnitzer	Ming Gu
Jocelyn Guidry	Gaetan Hache	Steve Hague
Satoshi Hamaguchi	Sven Hammarling	Mike Hansen
Richard Hanson	Richard Harke	Bill Hart
Vilya Harvey	Martin Hassner	Arthur S. Hathaway
Dan Hatton	Waldek Hebisch	Karl Hegbloom
Ralf Hemmecke	Henderson	Antoine Hersen
Nicholas J. Higham	Hoon Hong	Roger House
Gernot Hueber	Pietro Iglio	Alejandro Jakubi
Richard Jenks	Bo Kagstrom	William Kahan
Kyriakos Kalorkoti	Kai Kaminski	Grant Keady
Wilfrid Kendall	Tony Kennedy	David Kincaid
Keshav Kini	Ted Kosan	Paul Kosinski
Igor Kozachenko	Fred Krogh	Klaus Kusche

Bernhard Kutzler	Tim Lahey	Larry Lambe
Kaj Laurson	Charles Lawson	George L. Legendre
Franz Lehner	Frederic Lehobey	Michel Levaud
Howard Levy	J. Lewis	Ren-Cang Li
Rudiger Loos	Craig Lucas	Michael Lucks
Richard Luczak	Camm Maguire	Francois Maltey
Osni Marques	Alasdair McAndrew	Bob McElrath
Michael McGettrick	Edi Meier	Ian Meikle
David Mentre	Victor S. Miller	Gerard Milmeister
Mohammed Mobarak	H. Michael Moeller	Michael Monagan
Marc Moreno-Maza	Scott Morrison	Joel Moses
Mark Murray	William Naylor	Patrice Naudin
C. Andrew Neff	John Nelder	Godfrey Nolan
Arthur Norman	Jinzhong Niu	Michael O'Connor
Summat Oemrawsingh	Kostas Oikonomou	Humberto Ortiz-Zuazaga
Julian A. Padget	Bill Page	David Parnas
Susan Pelzel	Michel Petitot	Didier Pinchon
Ayal Pinkus	Frederick H. Pitts	Frank Pfenning
Jose Alfredo Portes	E. Quintana-Orti	Gregorio Quintana-Orti
Beresford Parlett	A. Petitet	Andre Platzer
Peter Poromaas	Claude Quitte	Arthur C. Ralfs
Norman Ramsey	Anatoly Raportirenko	Guilherme Reis
Huan Ren	Albert D. Rich	Michael Richardson
Jason Riedy	Renaud Rioboo	Jean Rivlin
Nicolas Robidoux	Simon Robinson	Raymond Rogers
Michael Rothstein	Martin Rubey	Jeff Rutter
Philip Santas	Alfred Scheerhorn	William Schelter
Gerhard Schneider	Martin Schoenert	Marshall Schor
Frithjof Schulze	Fritz Schwarz	Steven Segletes
V. Sima	Nick Simicich	William Sit
Elena Smirnova	Jacob Nyffeler Smith	Matthieu Sozeau
Ken Stanley	Jonathan Steinbach	Fabio Stumbo
Christine Sundaresan	Klaus Sutner	Robert Sutor
Moss E. Sweedler	Eugene Surowitz	Max Tegmark
T. Doug Telford	James Thatcher	Laurent Thery
Balbir Thomas	Mike Thomas	Dylan Thurston
Francoise Tisseur	Steve Toleque	Raymond Toy
Barry Trager	Themos T. Tsikas	Gregory Vanuxem
Kresimir Veselic	Christof Voemel	Bernhard Wall
Stephen Watt	Andreas Weber	Jaap Weel
Juergen Weiss	M. Weller	Mark Wegman
James Wen	Thorsten Werther	Michael Wester
R. Clint Whaley	James T. Wheeler	John M. Wiley
Berhard Will	Clifton J. Williamson	Stephen Wilson
Shmuel Winograd	Robert Wisbauer	Sandra Wityak
Waldemar Wiwianka	Knut Wolf	Yanyang Xiao
Liu Xiaojun	Clifford Yapp	David Yun
Qian Yun	Vadim Zhytnikov	Richard Zippel
Evelyn Zoernack	Bruno Zuercher	Dan Zwillinger

Contents

New Foreword

On October 1, 2001 Axiom was withdrawn from the market and ended life as a commercial product. On September 3, 2002 Axiom was released under the Modified BSD license, including this document. On August 27, 2003 Axiom was released as free and open source software available for download from the Free Software Foundation's website, Savannah.

Work on Axiom has had the generous support of the Center for Algorithms and Interactive Scientific Computation (CAISS) at City College of New York. Special thanks go to Dr. Gilbert Baumslag for his support of the long term goal.

The online version of this documentation is roughly 1000 pages. In order to make printed versions we've broken it up into three volumes. The first volume is tutorial in nature. The second volume is for programmers. The third volume is reference material. We've also added a fourth volume for developers. All of these changes represent an experiment in print-on-demand delivery of documentation. Time will tell whether the experiment succeeded.

Axiom has been in existence for over thirty years. It is estimated to contain about three hundred man-years of research and has, as of September 3, 2003, 143 people listed in the credits. All of these people have contributed directly or indirectly to making Axiom available. Axiom is being passed to the next generation. I'm looking forward to future milestones.

With that in mind I've introduced the theme of the "30 year horizon". We must invent the tools that support the Computational Mathematician working 30 years from now. How will research be done when every bit of mathematical knowledge is online and instantly available? What happens when we scale Axiom by a factor of 100, giving us 1.1 million domains? How can we integrate theory with code? How will we integrate theorems and proofs of the mathematics with space-time complexity proofs and running code? What visualization tools are needed? How do we support the conceptual structures and semantics of mathematics in effective ways? How do we support results from the sciences? How do we teach the next generation to be effective Computational Mathematicians?

The "30 year horizon" is much nearer than it appears.

Tim Daly
CAISS, City College of New York
November 10, 2003 ((iHy))

Chapter 1

Overview

This book contains the Firefox browser AJAX routines.

Build Instructions

```
 mkdir -p /home/silver/bitmaps
 cp bookvol11.pamphlet /home/silver
 cd /home/silver
 export AXIOM=(where)
 export PATH=$AXIOM/bin/lib:$AXIOM/bin:$PATH
 ( cd books ; gcc -o tanglec tanglec.c )
 books/tanglec bookvol11.pamphlet > Makefile
 make -j 10
 axiom -nox
  -> )set mes auto off
  -> )set out mathml on
  -> axServer(8085,multiServ)$AXSERV

Now start your browser and go to:
  file:///home/silver/rootpage.xhtml
and then do:
  Basic Commands -> Calculus -> Differentiate -> Continue
  Basic Commands -> Matrix -> Continue

You should see the result of the differentiate appear inline in the
page. You can change the values in the text areas, click continue,
and see the new result.
```

The Makefile

This Makefile assumes that it is being run in the final target directory of the build system. We walk the list of PAGES and untangle them from this file, along with any support files referenced by the pages.

— * —

```
BOOK=${SPD}/books/bookvol11.pamphlet
OUT=${MNT}/${SYS}/doc/hypertex

%.xhtml: ${BOOK}
@ echo making $*.xhtml
@ ${BOOKS}/tanglec ${BOOK} "$*.xhtml" > $*.xhtml

\getchunk{PAGES}

all: announce ${PAGES} finish

announce:
@ echo ==============================================
@ echo Making Browser-base Hypertex pages bookvol11
@ echo ==============================================

finish:
@ echo ==============================================
@ echo Finish Browser-base Hypertex pages bookvol11
@ echo ==============================================

@ mkdir -p bitmaps
@ ${BOOKS}/tanglec ${BOOK} axiom1.bitmap  >bitmaps/axiom1.bitmap
@ ${BOOKS}/tanglec ${BOOK} rcm3720.input  >rcm3720.input
@ ${BOOKS}/tanglec ${BOOK} strang.input   >strang.input
@ ${BOOKS}/tanglec ${BOOK} signatures.txt >signatures.txt
@ cp ${SPD}/books/ps/doctitle.png ${OUT}/doctitle.png
@ cp ${SPD}/books/ps/lightbayou.png ${OUT}/lightbayou.png
```

Building new pages

To add a new page you need to create a page with the default layout below and add the name of the page to the PAGES variable below.

Most of the pages have a default layout of the form:

```
\subsection{pagename.xhtml}
\begin{chunk}{pagename.xhtml}
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Page subtitle goes here</div>
  <hr/>
your basic page text goes here.
\getchunk{page foot}
\end{chunk}
```

There are several things to observe here: ¡ol¿ ¡li¿Each page lives in its own subsection and its own chunk.¡/li¿ ¡li¿The pagename and the chunkname are the same¡/li¿ ¡li¿The chunk includes the ¡¡standard head¿¿¡/li¿ ¡li¿The chunk includes the ¡¡page head¿¿¡/li¿ ¡li¿The chunk includes the ¡¡page foot¿¿¡/li¿ ¡/ol¿ The default page layout cannot communicate with Axiom.

Communicating with Axiom

If your page needs to communicate with Axiom you need to add some information in the header of the page. The default page that talks to Axiom has the form:

```
\subsection{pagename.xhtml}
<<pagename.xhtml>>=
<<standard head>>
  <script type="text/javascript">
<<handlefreevars>>
<<axiom talker>>
  </script>
 </head>
 <body onload="resetvars();">
<<page head>>
  <div align="center">Page subtitle goes here</div>
  <hr/>
your text goes here
your communication blocks go here
<<page foot>>
```

Handling statements with no free variables

Use a makeRequest call with a parameter of the id. Note that the div with id of "ansXX" will get replaced automatically and the "ans" prefix is required.

```
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="sin(x)" />
  <div id="ansp3"><div></div></div>
 </li>
```

Handling statements with free variables

Free variables exist are used in statements but they are defined in other statements. To make sure the free variables have the correct values you need to include an explicit list of the other ids that need to be executed *before* this statement. You do this with a call to "handleFree". It expects a list, enclosed in brackets, of the ids to execute in order. Be certain that the current id is at the end of the list.

```
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p9','p10']);"
    value="roman y" />
  <div id="ansp10"><div></div></div>
 </li>
```

Handling domain database lookups

Use an anchor tag of the form:

```
<a href="db.xhtml?Vector">Vector</a>
```

This will be interpreted by Axiom to mean that you want to do a lookup on a domain, category, or package whose name follows the question mark. Note that the domain name should NOT be an abbreviation.

Handling)show domain

Use a block containing a showcall of the form:

```
<li>
 <input type="submit" id="p17" class="subbut"
   onclick="showcall('p17');"
  value=")show DoubleFloat"/>
 <div id="ansp17"><div></div></div>
</li>
```

Note that the ")show" must be at the beginning of the line and that there can only be one space between the word show and the following argument.

Handling lisp expressions

Use a block containing a lispcall of the form: ¡li¿ ¡input type=”submit” id=”p2” class=”subbut” onclick=”lispcall(’p2’);” value=”(GETDATABASE ’—Matrix— ’CONSTRUCTORMODEMAP)” /¿ ¡div id=”ansp2”¿¡div¿¡/div¿¡/div¿ ¡/li¿ Note that this works but you can easily blow away your Axiom session with random statements. Let the coder beware.

Handling expressions that have no output

Use the CSS class="noresult" tag on the input form. This causes the item to show up in black text. It is still executable and is generally executed by handleFree calls because it contains definitions. However, things like function definitions in Axiom return no interesting output so there is no point in clicking on them.

```
<li>
 <input type="submit" id="p5" class="noresult"
   onclick="makeRequest('p5');"
   value=")set streams calculate 5" />
 <div id="ansp5"><div></div></div>
</li>
```

Defined Pages

Every page in this file is extracted by the Makefile. This is the list of pages that will be extracted. It is organized roughly in the hierarchy that you see in the browser pages. This is convention and is not required.

The page hierarchy (used by the Makefile) is:

— PAGES —

```
PAGES=rootpage.xhtml \
        testpage.xhtml \
        commandline.xhtml \
          menufileopen.xhtml \
          menufileread.xhtml \
          menufilesave.xhtml \
          menufilesaveas.xhtml \
          menufileloadlibrary.xhtml \
          menufileinputfile.xhtml \
          menufiletogglespool.xhtml \
          menufileprint.xhtml \
          menufileexit.xhtml \
          menueditcopy.xhtml \
          menueditcopytext.xhtml \
          menueditcopytex.xhtml \
          menueditdeleteselection.xhtml \
          menueditcopyasimage.xhtml \
          menueditselectiontoimage.xhtml \
          menueditselectiontoinput.xhtml \
          menueditcut.xhtml \
          menueditpaste.xhtml \
          menuaxiominterrupt.xhtml \
          menuaxiomrestart.xhtml \
          menuaxiomclearmemory.xhtml \
          menuaxiomaddtopath.xhtml \
          menuaxiomshowfunctions.xhtml \
          menuaxiomshowdefinition.xhtml \
          menuaxiomshowvariables.xhtml \
          menuaxiomdeletefunction.xhtml \
          menuaxiomdeletevariable.xhtml \
          menuaxiomtoggletimedisplay.xhtml \
          menuaxiomset.xhtml \
          menuaxiomdisplay.xhtml \
          menuequationssolve.xhtml \
          menuequationssolvenumerically.xhtml \
          menuequationsrootsofpolynomial.xhtml \
          menuequationsrealrootsofpolynmial.xhtml \
          menuequationssolvelinearsystem.xhtml \
          menuequationssolvealgebraicsystem.xhtml \
          menuequationseliminatevariable.xhtml \
          menuequationssolveode.xhtml \
          menuequationsinitialvalueproblem1.xhtml \
          menuequationsinitialvalueproblem2.xhtml \
          menuequationsboundaryvalueproblem.xhtml \
          menuequationssolveodewithlaplace.xhtml \
          menuequationsatvalue.xhtml \
          menualgebrageneratematrix.xhtml \
          menualgebraentermatrix.xhtml \
          menualgebrainvertmatrix.xhtml \
          menualgebracharacteristicpolynomial.xhtml \
          menualgebradeterminant.xhtml \
```

```
  menualgebraeigenvalues.xhtml \
  menualgebraeigenvectors.xhtml \
  menualgebraadjointmatrix.xhtml \
  menualgebratransposematrix.xhtml \
  menualgebramakelist.xhtml \
  menualgebraapplytolist.xhtml \
  menualgebramaptolist.xhtml \
  menualgebrareducelist.xhtml \
  menualgebramaptomatrix.xhtml \
  menucalculuslevel3.xhtml \
    menucalculuslevel3a.xhtml \
    menucalculuslevel3b.xhtml \
    menucalculuslevel3c.xhtml \
  menucalculusintegrate.xhtml \
  menucalculusrischintegrate.xhtml \
  menucalculuschangevariable.xhtml \
  menucalculusdifferentiate.xhtml \
  menucalculusfindlimit.xhtml \
  menucalculusgetseries.xhtml \
  menucalculuspadeapproximation.xhtml \
  menucalculuscalculussum.xhtml \
  menucalculuscalculusproduct.xhtml \
  menucalculuslaplacetransform.xhtml \
  menucalculusinverselaplacetransform.xhtml \
  menucalculusgreatestcommondivisor.xhtml \
  menucalculusleastcommonmultiple.xhtml \
  menucalculusdividepolynomials.xhtml \
  menucalculuspartialfractions.xhtml \
  menucalculuscontinuedfractions.xhtml \
  menusimplifysimplifyexpression.xhtml \
  menusimplifysimplifyradicals.xhtml \
  menusimplifyfactorexpression.xhtml \
  menusimplifyfactorcomplex.xhtml \
  menusimplifyexpandexpression.xhtml \
  menusimplifyexpandlogarithms.xhtml \
  menusimplifycontractlogarithms.xhtml \
  menusimplifyfactorialsandgamma.xhtml \
  menusimplifytrigsimplification.xhtml \
  menusimplifycomplexsimplification.xhtml \
  menusimplifysubtitute.xhtml \
  menusimplifyevaluatenounform.xhtml \
  menusimplifytogglealgebraicflag.xhtml \
  menusimplifyaddalgebraicequality.xhtml \
  menusimplifymoduluscomputation.xhtml \
  menunumerictogglenumericoutput.xhtml \
  menunumerictofloat.xhtml \
  menunumerictobigfloat.xhtml \
  menunumericsetprecision.xhtml \
basiccommand.xhtml \
tutorial.xhtml \
jenks.xhtml \
  calculus.xhtml \
    differentiate.xhtml \
    indefiniteintegral.xhtml \
```

```
        definiteintegral.xhtml \
        basiclimit.xhtml \
          reallimit.xhtml \
          complexlimit.xhtml \
          summation.xhtml \
      bcmatrix.xhtml \
      bcexpand.xhtml \
      draw.xhtml \
        draw2donevariable.xhtml \
        draw2ddefinedcurve.xhtml \
        draw2dpolynomialequation.xhtml \
        draw3dtwovariable.xhtml \
        draw3ddefinedtube.xhtml \
        draw3ddefinedsurface.xhtml \
      series.xhtml \
        seriesexpand.xhtml \
        taylorseries.xhtml \
        laurentseries.xhtml \
        puiseuxseries.xhtml \
      solve.xhtml \
        solvelinearequations.xhtml \
        solvelinearmatrix.xhtml \
        solvesystempolynomials.xhtml \
        solvesinglepolynomial.xhtml \
    topreferencepage.xhtml \
      releasenotes.xhtml \
      usersguidepage.xhtml \
      foundationlibrarydocpage.xhtml \
      topicspage.xhtml \
        cats.xhtml \
        dlmf.xhtml \
        dlmfapproximations.xhtml \
        dlmfasymptoticexpansions.xhtml \
        dlmfbarnesgfunction.xhtml \
        dlmfbetafunction.xhtml \
        dlmfcontinuedfractions.xhtml \
        dlmfdefinitions.xhtml \
        dlmffunctionrelations.xhtml \
        dlmfgraphics.xhtml \
        dlmfinequalities.xhtml \
        dlmfinfiniteproducts.xhtml \
        dlmfintegrals.xhtml \
        dlmfintegralrepresentations.xhtml \
        dlmfmathematicalapplications.xhtml \
        dlmfmethodsofcomputation.xhtml \
        dlmfmultidimensionalintegral.xhtml \
        dlmfnotation.xhtml \
        dlmfphysicalapplications.xhtml \
        dlmfpolygammafunctions.xhtml \
        dlmfqgammaandbetafunctions.xhtml \
        dlmfseriesexpansions.xhtml \
        dlmfsums.xhtml \
        dlmfsoftware.xhtml \
        dlmfspecialvaluesandextrema.xhtml \
```

```
  dlmftables.xhtml \
uglangpage.xhtml \
examplesexposedpage.xhtml \
ugsyscmdpage.xhtml \
operations.xhtml \
dblookup.xhtml \
  dbcharacteristic.xhtml \
    dbcomplexcomplex.xhtml \
    dbcomplexconjugate.xhtml \
    dbcomplexfactor.xhtml \
    dbcompleximag.xhtml \
    dbcomplexnorm.xhtml \
    dbcomplexreal.xhtml \
  dbcomplexdoublefloat.xhtml \
  dbcomplexfloat.xhtml \
  dbcomplexinteger.xhtml \
  dbexpressioninteger.xhtml \
  dbfractioninteger.xhtml \
  dbfractionpolynomialinteger.xhtml \
  dbopacos.xhtml \
  dbopacosh.xhtml \
  dbopacot.xhtml \
  dbopacoth.xhtml \
  dbopacsc.xhtml \
  dbopacsch.xhtml \
  dbopaddmod.xhtml \
  dbopairyai.xhtml \
  dbopairybi.xhtml \
  dbopapproximants.xhtml \
  dbopasin.xhtml \
  dbopasinh.xhtml \
  dbopasec.xhtml \
  dbopasech.xhtml \
  dbopatan.xhtml \
  dbopatanh.xhtml \
  dbopbernoullib.xhtml \
  dbopbesseli.xhtml \
  dbopbesselj.xhtml \
  dbopbesselk.xhtml \
  dbopbessely.xhtml \
  dbopbeta.xhtml \
  dbopbinary.xhtml \
  dbopcardinalnumber.xhtml \
  dbopchebyshevt.xhtml \
  dbopchebyshevu.xhtml \
  dbopcoefficient.xhtml \
  dbopcoefficients.xhtml \
  dbopcoerce.xhtml \
  dbopcolumn.xhtml \
  dbopcompactfraction.xhtml \
  dbopcomplexeigenvectors.xhtml \
  dbopcomplexelementary.xhtml \
  dbopcomplexintegrate.xhtml \
  dbopcomplexlimit.xhtml \
```

```
dbopcomplexsolve.xhtml \
dbopcontent.xhtml \
dbopcontinuedfraction.xhtml \
dbopconvergents.xhtml \
dbopconvert.xhtml \
dbopcopy.xhtml \
dbopcos.xhtml \
dbopcosh.xhtml \
dbopcot.xhtml \
dbopcoth.xhtml \
dbopcount.xhtml \
dbopcountableq.xhtml \
dbopcreate3space.xhtml \
dbopcsc.xhtml \
dbopcsch.xhtml \
dbopcurve.xhtml \
dbopcycleragits.xhtml \
dbopcyclotomic.xhtml \
dbopd.xhtml \
dbopdecimal.xhtml \
dbopdefiningpolynomial.xhtml \
dbopdegree.xhtml \
dbopdenom.xhtml \
dbopdraw.xhtml \
dbopdeterminant.xhtml \
dbopdiagonalmatrix.xhtml \
dbopdigamma.xhtml \
dbopdigits.xhtml \
dbopdimension.xhtml \
dbopdivide.xhtml \
dbopdivisors.xhtml \
dbopei.xhtml \
dbopeigenmatrix.xhtml \
dbopeigenvalues.xhtml \
dbopeigenvector.xhtml \
dbopeigenvectors.xhtml \
dbopelt.xhtml \
dbopequal.xhtml \
dbopeulere.xhtml \
dbopeulerphi.xhtml \
dbopeval.xhtml \
dbopevenq.xhtml \
dbopexp.xhtml \
dbopexquo.xhtml \
dbopfactor.xhtml \
dbopfactorfraction.xhtml \
dbopfibonacci.xhtml \
dbopfiniteq.xhtml \
dbopfirstdenom.xhtml \
dbopfirstnumer.xhtml \
dbopfractragits.xhtml \
dbopfractionpart.xhtml \
dbopgamma.xhtml \
dbopgcd.xhtml \
```

```
dbophermiteh.xhtml \
dbophex.xhtml \
dbophorizconcat.xhtml \
dbophtrigs.xhtml \
dbophypergeometric0f1.xhtml \
dbopinteger.xhtml \
dbopintegrate.xhtml \
dbopinverse.xhtml \
dbopinvmod.xhtml \
dbopjacobi.xhtml \
dboplaguerrel.xhtml \
dboplaurent.xhtml \
dboplcm.xhtml \
dbopleadingcoefficient.xhtml \
dbopleadingmonomial.xhtml \
dboplegendre.xhtml \
dboplength.xhtml \
dboplimit.xhtml \
dboplog.xhtml \
dboploggamma.xhtml \
dbopmainvariable.xhtml \
dbopmakegraphimage.xhtml \
dbopmakeobject.xhtml \
dbopmakeviewport3d.xhtml \
dbopmap.xhtml \
dbopmapbang.xhtml \
dbopmatrix.xhtml \
dbopmax.xhtml \
dbopmemberq.xhtml \
dbopmin.xhtml \
dbopminimumdegree.xhtml \
dbopminus.xhtml \
dbopmoebiusmu.xhtml \
dbopmonicdivide.xhtml \
dbopmulmod.xhtml \
dbopncols.xhtml \
dbopnegativeq.xhtml \
dbopnew.xhtml \
dbopnextprime.xhtml \
dbopnorm.xhtml \
dbopnrows.xhtml \
dbopnthfractionalterm.xhtml \
dbopnthroot.xhtml \
dbopnullity.xhtml \
dbopnullspace.xhtml \
dbopnumberoffractionalterms.xhtml \
dbopnumer.xhtml \
dbopnumeric.xhtml \
dbopoddq.xhtml \
dboponedimensionalarray.xhtml \
dbopoperator.xhtml \
dboporthonormalbasis.xhtml \
dbopoutputfixed.xhtml \
dbopoutputfloating.xhtml \
```

```
dbopoutputgeneral.xhtml \
dbopoutputspacing.xhtml \
dboppadicfraction.xhtml \
dboppartialfraction.xhtml \
dboppartialquotients.xhtml \
dboppattern.xhtml \
dboppermanent.xhtml \
dboppi.xhtml \
dbopplus.xhtml \
dboppolygamma.xhtml \
dboppositiveq.xhtml \
dboppositiveremainder.xhtml \
dbopprefixragits.xhtml \
dbopprevprime.xhtml \
dbopprimefactor.xhtml \
dbopprimeq.xhtml \
dbopprimes.xhtml \
dboppuiseux.xhtml \
dbopqelt.xhtml \
dbopqseteltbang.xhtml \
dbopquatern.xhtml \
dbopquo.xhtml \
dbopradicaleigenvectors.xhtml \
dbopradicalsolve.xhtml \
dboprank.xhtml \
dbopratdenom.xhtml \
dboprealeigenvectors.xhtml \
dboprealelementary.xhtml \
dbopreduce.xhtml \
dbopreductum.xhtml \
dboprem.xhtml \
dbopresetvariableorder.xhtml \
dbopresultant.xhtml \
dboprootof.xhtml \
dboprootsimp.xhtml \
dboprootsof.xhtml \
dbopround.xhtml \
dboprow.xhtml \
dboprowechelon.xhtml \
dbopsetcolumnbang.xhtml \
dbopsetelt.xhtml \
dbopseteltbang.xhtml \
dbopsetrowbang.xhtml \
dbopsetsubmatrixbang.xhtml \
dbopsign.xhtml \
dbopsimplify.xhtml\
dbopsec.xhtml \
dbopsech.xhtml \
dbopseries.xhtml \
dbopseriessolve.xhtml \
dbopsin.xhtml \
dbopsingleintegerand.xhtml \
dbopsingleintegernot.xhtml \
dbopsingleintegeror.xhtml \
```

```
      dbopsingleintegerxor.xhtml \
      dbopsinh.xhtml \
      dbopsetvariableorder.xhtml \
      dbopsolve.xhtml \
      dbopsqrt.xhtml \
      dbopstar.xhtml \
      dbopstarstar.xhtml \
      dbopsubmatrix.xhtml \
      dbopsubmatrix.xhtml \
      dbopsubmod.xhtml \
      dbopsurface.xhtml \
      dbopsumofkthpowerdivisors.xhtml \
      dboptan.xhtml \
      dboptanh.xhtml \
      dboptaylor.xhtml \
      dboptimes.xhtml \
      dboptotaldegree.xhtml \
      dboptrace.xhtml \
      dboptranspose.xhtml \
      dboptrigs.xhtml \
      dboptruncate.xhtml \
      dbopvariables.xhtml \
      dbopvectorise.xhtml \
      dbopvectorspace.xhtml \
      dbopvertconcat.xhtml \
      dbopwholepart.xhtml \
      dbopwholeragits.xhtml \
      dbopwrite.xhtml \
      dbopzeroof.xhtml \
      dbopzerosof.xhtml \
      dbopzeroq.xhtml \
      dbpolynomialinteger.xhtml \
      dbpolynomialfractioninteger.xhtml \
    systemvariables.xhtml \
    glossarypage.xhtml \
    htxtoppage.xhtml \
    refsearchpage.xhtml \
  topicspage.xhtml \
    numberspage.xhtml \
      numintegers.xhtml \
        numgeneralinfo.xhtml \
          numbasicfunctions.xhtml \
            numintegerfractions.xhtml \
          numnumbertheoreticfunctions.xhtml \
        numfactorization.xhtml \
        numfunctions.xhtml \
        numexamples.xhtml \
        numproblems.xhtml \
      numfractions.xhtml \
        numrationalnumbers.xhtml \
        numquotientfields.xhtml \
      nummachinefloats.xhtml \
      numfloat.xhtml \
        introtofloat.xhtml \
```

```
    conversionfunctions.xhtml \
    outputfunctions.xhtml \
    determinantofhilbert.xhtml \
  numcomplexnumbers.xhtml \
  numfinitefields.xhtml \
  numnumericfunctions.xhtml \
  numcardinalnumbers.xhtml \
  nummachinesizedintegers.xhtml \
  numromannumerals.xhtml \
  numcontinuedfractions.xhtml \
  numpartialfractions.xhtml \
  numquaternions.xhtml \
  numoctonions.xhtml \
  numrepeatingdecimals.xhtml \
  numrepeatingbinaryexpansions.xhtml \
  numrepeatinghexexpansions.xhtml \
  numotherbases.xhtml \
polynomialpage.xhtml \
  polybasicfunctions.xhtml \
  polysubstitutions.xhtml \
  polyfactorization.xhtml \
    polyfactorization1.xhtml \
    polyfactorization2.xhtml \
    polyfactorization3.xhtml \
    polyfactorization4.xhtml \
  polygcdandfriends.xhtml \
  polyroots.xhtml \
    polyroots1.xhtml \
    polyroots2.xhtml \
    polyroots3.xhtml \
    polyroots4.xhtml \
  polyspecifictypes.xhtml \
    polyspecifictypes1.xhtml \
      factored.xhtml \
    polyspecifictypes2.xhtml \
    polyspecifictypes3.xhtml \
    polyspecifictypes4.xhtml \
functionpage.xhtml \
  funrationalfunctions.xhtml \
  funalgebraicfunctions.xhtml \
  funelementaryfunctions.xhtml \
  funsimplification.xhtml \
  funpatternmatching.xhtml \
  funoperatoralgebra.xhtml \
equationpage.xhtml \
  equsystemlinear.xhtml \
  equdifferential.xhtml \
    equdifferentiallinear.xhtml \
    equdifferentialnonlinear.xhtml \
    equdifferentialpowerseries.xhtml \
calculuspage.xhtml \
  callimits.xhtml \
  calderivatives.xhtml \
  calintegrals.xhtml \
```

```
    calmoreintegrals.xhtml \
    callaplace.xhtml \
    calseries.xhtml \
      calseries1.xhtml \
      calseries2.xhtml \
      calseries3.xhtml \
      calseries4.xhtml \
      calseries5.xhtml \
      calseries6.xhtml \
      calseries7.xhtml \
      calseries8.xhtml \
  linalgpage.xhtml \
    linintro.xhtml \
    lincreate.xhtml \
    linoperations.xhtml \
    lineigen.xhtml \
    linhilbert.xhtml \
    linpermaent.xhtml \
    linvectors.xhtml \
    linsquarematrices.xhtml \
    lin1darrays.xhtml \
    lin2darrays.xhtml \
    linconversion.xhtml \
  graphicspage.xhtml \
    graphexamples.xhtml \
      graphexamplesassorted.xhtml \
      graphexamplesthreed.xhtml \
      graphexamplesonevariable.xhtml \
      graphexamplesparametric.xhtml \
      graphexamplespolar.xhtml \
      graphexamplesimplicit.xhtml \
      graphexampleslistofpoints.xhtml \
    graph2d.xhtml \
      graph2dimplicit.xhtml \
      graph2dlistsofpoints.xhtml \
      graph2donevariable.xhtml \
      graph2dparametric.xhtml \
      graph2dpolar.xhtml \
    graph3d.xhtml \
      graph3dobjects.xhtml \
      graph3dparametric.xhtml \
      graph3dsurfaces.xhtml \
      graph3dtubeplots.xhtml \
      graph3dtwovariables.xhtml \
    graphviewports.xhtml \
  algebrapage.xhtml \
    algnumbertheory.xhtml \
      algnumbertheorygalois.xhtml \
    alggrouptheory.xhtml \
      alggrouptheorygroup.xhtml \
      alggrouptheoryrepa6.xhtml \
      alggrouptheoryreptheory.xhtml \
  cryptopage.xhtml \
    cryptoclass1.xhtml \
```

```
          cryptoclass2.xhtml \
          cryptoclass3.xhtml \
          cryptoclass4.xhtml \
          cryptoclass5.xhtml \
          cryptoclass6.xhtml \
          cryptoclass7.xhtml \
          cryptoclass8.xhtml \
          cryptoclass9.xhtml \
          cryptoclass10.xhtml \
          cryptoclass11.xhtml \
        ocwmit18085.xhtml \
          ocwmit18085lecture1.xhtml \
          ocwmit18085lecture2.xhtml \
      man0page.xhtml \
      topexamplepage.xhtml \
      topsettingspage.xhtml \
      axiomfonts.xhtml \
      pagelist.xhtml \
      pagematrix.xhtml \
      pageonedimensionalarray.xhtml \
      pagepermanent.xhtml \
      pageset.xhtml \
      pagesquarematrix.xhtml \
      pagetable.xhtml \
      pagetwodimensionalarray.xhtml \
      pagevector.xhtml
```

———-

The Standard Layout

Generally a page has a standard layout using a couple of chunks to minimize the typing. The defined chunks are:

- "standard head" which includes the head element, xmlns, meta, and title element. It also contains the "style" element for CSS information.
- "page head" contains the banner information
- "page foot" contains the trailing page information and the body-end and html-end tags

So the basic layout looks like

```
<<standard head>>
    (local and general javascript goes here)
 </head>
 <body>
<<page head>>
    (local page definition goes here)
<<page foot>>
```

So all you need to worry about are the actual page forms and the javascript to fetch those forms.

For "active pages", that is those that communicate with Axiom they generally define a javascript function called "commandline" which formats the request to be sent to the host. You also need to include the "axiom talker" chunk. Note that "axiom talker" expects the "commandline" function to exist and calls it. Thus, for the page that handles differentiation calls to Axiom we add the local javascript:

```
  <script type="text/javascript">
   function commandline(arg) {
    return(document.getElementById('comm').value);
   }
<<axiom talker>>
  </script>
```

This defined the "commandline" function and embeds the "axiom talker". The "commandline" function knows how to fetch fields from the rest of the page and format them into a single Axiom string. This is page specific code. For example, this shows a single input line which will be sent to the host when the "Continue" is pressed:

```
  <form id="commreq">
    <p>
      Type an input command line to Axiom:<br/>
       <input type="text" id="comm" name="command" size="80"/>
     <<continue button>>
    </p>
  </form>
 <<answer field>>
```

Note that the commandline function takes an argument which it gets from the caller, makeRequest. This argument can be used to distinguish which button was pressed.

The **div** section with **id="mathAns"** is replaced by the result sent from the server.

Cascading Style Sheet

Standard Style Sheet

This is the standard CSS style section that gets included with every page. We do this here but it could be a separate style sheet. It hardly matters either way as the style sheet is trivial.

— **standard background** —

```
body {
  margin: 0px;
  padding: 0px;
  height: 512px;
  background: url(lightbayou.png) no-repeat;
  -webkit-background-size: cover;
  -moz-background-size: cover;
  -o-background-size: cover;
  background-size: cover;
}
```

————

— style —

```
  <style>

   html {
     background-color: #ECEA81;
   }

   div.command {
     color:red;
   }

   div.center {
     color:blue;
   }

   div.reset {
     visibility:hidden;
   }

   div.mathml {
     color:blue;
   }

   input.subbut {
     background-color:#ECEA81;
     border: 0;
     color:green;
     font-family: "Courier New", Courier, monospace;
   }

   input.noresult {
     background-color:#ECEA81;
     border: 0;
     color:black;
     font-family: "Courier New", Courier, monospace;
   }

   span.cmd {
     color:green;
     font-family: "Courier New", Courier, monospace;
   }

   pre {
     font-family: "Courier New", Courier, monospace;
   }

\getchunk{standard background}
  </style>
```

———–

Menu style sheet

— menu style —

```
<style>

  form {
    margin-top: 0;
    margin-bottom: 0;
    padding-left: 10px;
  }

  table.main {
    background-color: #ECEA81;
    font-size: 10pt;
    font-family: arial;
  }

  .main A:link {
    font-family: arial;
    color:#016bbd;
  }

  .main A:hover {
    font-family: arial;
    color: #64747A;
  }

  .main A:visited {
    font-family: arial;
    color:#336699;
  }

  /* style the outer div to give it width */
  .menu {
    font-size:0.85em;
  }

  /* remove all the bullets, borders and padding
     from the default list styling */
  .menu ul {
    padding:0;
    width:1000px;
    margin:0;
    list-style-type:none;
    white-space: normal;
  }

  .menu ul ul {
    width:90px;
  }

  /* float the list to make it horizontal and a relative position
```

```
   so that you can control the dropdown menu positon */
.menu li {
  float:left;
  width:90px;
  position:relative;
}

/* style the links for the top level */
.menu a, .menu a:visited {
  display:block;
  font-size:12px;
  text-decoration:none;
  font-weight:bold;
  color:#2952a7;
  width:99px;
  height:32px;
  line-height:29px;
  border:0px solid #fff;
  border-width:0px 0px 0 0px;
  text-align:center;
}

/* style the second level links
   if this breaks all the level 2 links appear at once */
.menu ul ul a, .menu ul ul a:visited {
  font-size:10px;
  font-weight:normal;
  background:#d4d8bd;
  color:#000;
  height:auto;
  line-height:1em;
  padding:5px 10px;
  width:78px
}

/* style the top level hover */
.menu a:hover, .menu ul ul a:hover{
  border:1px solid #000;
  border-width:1px 1px 0 1px;
}

.menu :hover > a, .menu ul ul :hover > a {
  border:1px solid #000;
  border-width:1px 1px 0 1px;
}

/* style the second level background */
.menu ul ul a.drop, .menu ul ul a.drop:visited {
  background:#e0d8d0;
}

/* style the second level hover */
.menu ul ul a.drop:hover{
  background:#c9ba65;
```

```
}

.menu ul ul :hover > a.drop {
  background:#c9ba65;
}

/* style the third level background */
.menu ul ul ul a, .menu ul ul ul a:visited {
  background:#e2dfa8;
}

/* style the third level hover */
.menu ul ul ul a:hover {
  background:#b2ab9b;
}

.menu ul ul ul :hover > a {
  background:#b2ab9b;
}

/* hide the sub levels and give them a positon absolute
   so that they take up no room */
.menu ul ul {
  visibility:hidden;
  position:absolute;
  height:0;
  top:31px;
  left:0;
  width:150px;
}

/* position the third level flyout menu */
.menu ul ul ul{
  left:100px;
  top:0;
  width:90px;
}

/* position the third level flyout menu for a left flyout */
.menu ul ul ul.left {
  left:-90px;
}

/* style the table so that it takes no part in the layout -
   required for IE to work */
.menu table {
  position:absolute;
  top:0;
  left:0;
}

/* make the second level visible when hover on
  first level list OR link */
.menu ul li:hover ul,
```

```
   .menu ul a:hover ul {
     visibility:visible;
  }

  /* keep the third level hidden when you hover on
     first level list OR link */
  .menu ul :hover ul ul {
    visibility:hidden;
  }

  /* keep the fourth level hidden when you hover on
     second level list OR link */
  .menu ul :hover ul :hover ul ul {
    visibility:hidden;
  }

  /* make the third level visible when you hover over second level list
     OR link */
  .menu ul :hover ul :hover ul {
    visibility:visible;
  }

  /* make the fourth level visible when you hover over third level list
     OR link */
  .menu ul :hover ul :hover ul :hover ul {
    visibility:visible;
  }

 </style>
```

——–

standard head

This is the standard head section. It is used on pages that do not include javascript. Note that it does NOT include the ¡/head¿ so the javascript can be added easily.

— **standard head** —

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:xlink="http://www.w3.org/1999/xlink"
      xmlns:m="http://www.w3.org/1998/Math/MathML">
 <head>
  <meta http-equiv="Content-Type" content="text/html" charset="us-ascii"/>
  <title>Axiom Documentation</title>
\getchunk{style}
```

——–

This is the standard page header.

— **page head** —

```
 <div><center><img src="bitmaps/axiom2.png"/></center></div>
 <hr/>
```

```
 <div><center><br/>
```

———-

This is the standard page foot.

— **page foot** —

```
 </center></div>
 </body>
</html>
```

———-

This is the standard continue button

— **continue button** —

```
   <center>
     <input type="button" value="Continue" name="continue"
       onclick="javascript:makeRequest('');"/>
   </center>
```

———-

This is where to place the math answer

— **answer field** —

```
  <div id="mathAns"><div></div></div>
```

———-

Javascript functions

Show only mathml

This function will show only the mathml result in the response. It is useful for particular pages that have lists of equations where all you care about are the answers.

— **showonlymathml** —

```
<![CDATA[
// The structure returned from Axiom now is
// <div class="stepnum"></div>
// <div class="command"></div>
// <div class="algebra"></div>
// <div class="mathml"></div>
// <div class="type"></div>
// This function will pick up the mathml and put it into 'indiv'
 function showanswer(mathString,indiv) {
    var mystr = mathString.split("</div>");
    var mymathstr = mystr[3].concat("</div>");
    // this turns the string into a dom fragment
    var mathRange = document.createRange();
    var mathBox=document.createElementNS('http://www.w3.org/1999/xhtml','div');
    mathRange.selectNodeContents(mathBox);
    var mymath = mathRange.createContextualFragment(mymathstr);
    mathBox.appendChild(mymath);
    // now we need to format it properly
```

```
    // and we stick the result into the requested div block as a child.
    var mathAns = document.getElementById(indiv);
    mathAns.removeChild(mathAns.firstChild);
    mathAns.appendChild(mathBox);
  }
]]>
```

———-

Show Full Answer

This function will show the full answer in the response including the step number, the command, the mathml and the type. The algebra portion is currently ignored.

— **showfullanswer** —

```
// The structure returned from Axiom now is
// <div class="stepnum"></div>
// <div class="command"></div>
// <div class="algebra"></div>
// <div class="mathml"></div>
// <div class="type"></div>
// This function will format the output as a console session
<![CDATA[
 function showanswer(mathString,indiv) {
    var mystr = mathString.split("</div>");
      // first we prepare the step number
    var mystept1 = mystr[0].lastIndexOf(">");
    var mystepstr = mystr[0].substr(mystept1+1);
      // now we get the command
    var mycmdt1 = mystr[1].lastIndexOf(">");
    var mycmdstr = mystr[1].substr(mycmdt1+1);
    var myprompt = '('+mystepstr+') -> '+mycmdstr;
      // now we handle the mathml
    var mymathstr = mystr[3].concat("</div>");
      // and the type, we need to insert the string "Type: "
    var mytypet1 = mystr[4].lastIndexOf(">");
    var mytypet2 = mystr[4].substr(mytypet1+1).concat("</div>");
    var mytypestr = '<div> Type: '.concat(mytypet2);
      // bang the whole thing together
    var finaldiv='<div class="command">'+myprompt+'</div>'+mymathstr+mytypestr;
      // this turns the string into a dom fragment
    var mathRange = document.createRange();
    var mathBox=document.createElementNS('http://www.w3.org/1999/xhtml','div');
    mathRange.selectNodeContents(mathBox);
    var answer = mathRange.createContextualFragment(finaldiv);
    mathBox.appendChild(answer);
      // and we stick the result into the requested div block as a child.
    var mathAns = document.getElementById(indiv);
    mathAns.removeChild(mathAns.firstChild);
    mathAns.appendChild(mathBox);
 }
]]>
```

———-

Handle Free Variables

— handlefreevars —

```
<![CDATA[
     // This is a hash table of the values we've evaluated.
     // This is indexed by a string argument.
     // A value of 0 means we need to evaluate the expression
     // A value of 1 means we have evaluated the expression
   Evaled = new Array();
     // this says we should modify the page
   hiding = 'show';
     // and this is the id of the div tag to modify (defaulted)
   thediv = 'mathAns';
     // commandline will mark that its arg has been evaled so we don't repeat
   function commandline(arg) {
     Evaled[arg] = 0;  // remember that we have set this value
     thediv='ans'+arg; // mark where we should put the output
     var ans = document.getElementById(arg).value;
     return(ans);
   }
   // the function only modifies the page if when we're showing the
   // final result, otherwise it does nothing.
   function showanswer(mathString,indiv) {
     if (hiding == 'show') { // only do something useful if we're showing
       indiv = thediv;  // override the argument so we can change it
       var mystr = mathString.split("</div>");
       for (var i=0; i < mystr.length; i++) {
         if (mystr[i].indexOf("mathml") > 0) {
           var mymathstr = mystr[i].concat("</div>");
         }
       }
       // this turns the string into a dom fragment
       var mathRange = document.createRange();
       var mathBox=
               document.createElementNS('http://www.w3.org/1999/xhtml','div');
       mathRange.selectNodeContents(mathBox);
       var mymath = mathRange.createContextualFragment(mymathstr);
       mathBox.appendChild(mymath);
       // now we need to format it properly
       // and we stick the result into the requested div block as a child.
       var mathAns = document.getElementById(indiv);
       mathAns.removeChild(mathAns.firstChild);
       mathAns.appendChild(mathBox);
     }
   }
   // this function takes a list of expressions ids to evaluate
   // the list contains a list of "free" expression ids that need to
   // be evaluated before the last expression.
   // For each expression id, if it has not yet been evaluated we
   // evaluate it "hidden" otherwise we can skip the expression.
   // Once we have evaluated all of the free expressions we can
   // evaluate the final expression and modify the page.
```

```
  function handleFree(arg) {
    var placename = arg.pop();      // last array val is real
    var mycnt = arg.length;         // remaining free vars
      // we handle all of the prerequired expressions quietly
    hiding = 'hide';
    for (var i=0; i<mycnt; i++) {   // for each of the free variables
      if (Evaled[arg[i]] == null) { // if we haven't evaled it
        Evaled[arg[i]] = 0;         // remember we evaled it
        makeRequest(arg[i]);        // initialize the free values
      }
    }
      // and now we start talking to the page again
    hiding = 'show';                // we want to show this
    thediv = 'ans'+placename;       // at this div id
    makeRequest(placename);         // and we eval and show it
  }
]]>
```

——–

axiom talker

— axiom talker —

```
<![CDATA[
  function ignoreResponse() {}
  function resetvars() {
    http_request = new XMLHttpRequest();
    http_request.open('POST', '127.0.0.1:8085', true);
    http_request.onreadystatechange = ignoreResponse;
    http_request.setRequestHeader('Content-Type', 'text/plain');
    http_request.send("command=)clear all");
    return(false);
  }
]]>
 function init() {
 }
 function makeRequest(arg) {
   http_request = new XMLHttpRequest();
   var command = commandline(arg);
   //alert(command);
   http_request.open('POST', '127.0.0.1:8085', true);
   http_request.onreadystatechange = handleResponse;
   http_request.setRequestHeader('Content-Type', 'text/plain');
   http_request.send("command="+command);
   return(false);
 }
 function lispcall(arg) {
   http_request = new XMLHttpRequest();
   var command = commandline(arg);
   //alert(command);
   http_request.open('POST', '127.0.0.1:8085', true);
   http_request.onreadystatechange = handleResponse;
```

```
   http_request.setRequestHeader('Content-Type', 'text/plain');
   http_request.send("lispcall="+command);
   return(false);
 }
 function showcall(arg) {
   http_request = new XMLHttpRequest();
   var command = commandline(arg);
   //alert(command);
   http_request.open('POST', '127.0.0.1:8085', true);
   http_request.onreadystatechange = handleResponse;
   http_request.setRequestHeader('Content-Type', 'text/plain');
   http_request.send("showcall="+command);
   return(false);
 }
 function interpcall(arg) {
   http_request = new XMLHttpRequest();
   var command = commandline(arg);
   //alert(command);
   http_request.open('POST', '127.0.0.1:8085', true);
   http_request.onreadystatechange = handleResponse;
   http_request.setRequestHeader('Content-Type', 'text/plain');
   http_request.send("interpcall="+command);
   return(false);
 }
 function handleResponse() {
  if (http_request.readyState == 4) {
   if (http_request.status == 200) {
    showanswer(http_request.responseText,'mathAns');
   } else
   {
     alert('There was a problem with the request.'+ http_request.statusText);
   }
  }
 }
```

Pages

— testpage.xhtml —

```
\getchunk{standard head}
\getchunk{menu style}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Test Page</div>
```

```
<hr/>
<div align="center">
 <!--main menu-->
 <form method="get" action="foo.xhtml">
  <table class="mainmenu" border="0" cellspacing="0" cellpadding="0">
   <tr>
    <td align="center" valign="middle" nowrap="nowrap">
     <div class="menu">
      <ul>
       <!-- Begin File Menu -->
       <li>
        <a class="toplevel" href="/">
         File<!--span class="nabla">&nabla;</span-->
        </a>
        <ul>
         <li>
          <a class="drop" href="file/open.xhtml">
           Open
          </a>
         </li>
         <li>
          <a class="drop" href="file/readfile.xhtml">
           Read file
          </a>
         </li>
         <li>
          <a class="drop" href="file/save.xhtml">
           Save
          </a>
         </li>
         <li>
          <a class="drop" href="file/saveas.xhtml">
           Save as
          </a>
         </li>
         <li>
          <a class="drop" href="file/input.xhtml">
           Embed
          </a>
         </li>
         <li>
          <a class="drop" href="file/batchfile.xhtml">
           Batch file
          </a>
         </li>
         <li>
          <a class="drop" href="file/exporttohtml.xhtml">
           Export to html
          </a>
         </li>
         <li>
          <a class="drop" href="file/selectfile.xhtml">
           Select file
          </a>
```

```
  </li>
  <li>
   <a class="drop" href="file/monitorfile.xhtml">
    Monitor file
   </a>
  </li>
  <li>
   <a class="drop" href="file/print.xhtml">
    Print
   </a>
  </li>
  <li>
   <a class="drop" href="file/exit.xhtml">
    Exit
   </a>
  </li>
 </ul>
</li>
<!-- End File Menu -->
<!-- Start Edit Menu -->
<li>
 <a class="toplevel" href="/">
  Edit<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop" href="edit/copy.xhtml">
    Copy
   </a>
  </li>
  <li>
   <a class="drop" href="edit/copytext.xhtml">
    Copy text
   </a>
  </li>
  <li>
   <a class="drop" href="edit/copytex.xhtml">
    Copy TeX
   </a>
  </li>
  <li>
   <a class="drop" href="edit/deleteselection.xhtml">
     Delete selection
   </a>
  </li>
  <li>
   <a class="drop" href="edit/copyasimage.xhtml">
    Copy as image
   </a>
  </li>
  <li>
   <a class="drop" href="edit/selectiontoimage.xhtml">
    Selection to image
   </a>
```

```
    </li>
    <li>
     <a class="drop" href="edit/selectiontoinput.xhtml">
      Selection to input
     </a>
    </li>
    <li>
     <a class="drop" href="edit/cut.xhtml">
      Cut
     </a>
    </li>
    <li>
     <a class="drop" href="edit/paste.xhtml">
      Paste
     </a>
    </li>
    <li>
     <a class="drop" href="edit/print.xhtml">
      Print
     </a>
    </li>
   </ul>
  </li>
  <!-- End Edit Menu -->
  <!-- Start Axiom Menu -->
  <li>
   <a class="toplevel" href="/">
    Axiom<!--span class="nabla">&nabla;</span-->
   </a>
   <ul>
    <li>
     <a class="drop" href="axiom/interrupt.xhtml">
      Interrupt
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/restart.xhtml">
      Restart
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/clearmemory.xhtml">
      Clear Memory
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/addtopath.xhtml">
       Add to path
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/showfunctions.xhtml">
      Show functions
     </a>
```

```
    </li>
    <li>
     <a class="drop" href="axiom/showdefinition.xhtml">
      Show definition
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/showvariables.xhtml">
      Show variables
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/deletefunction.xhtml">
      Delete function
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/deletevariable.xhtml">
      Delete variable
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/toggletimedisplay.xhtml">
      Toggle time display
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/change2ddisplay.xhtml">
      Change 2D display
     </a>
    </li>
    <li>
     <a class="drop" href="axiom/displaytex.xhtml">
      Display TeX
     </a>
    </li>
   </ul>
  </li>
  <!-- End Axiom Menu -->
  <!-- Start Equations Menu -->
  <li>
   <a class="toplevel" href="/">
    Equations<!--span class="nabla">&nabla;</span-->
   </a>
   <ul>
    <li>
     <a class="drop" href="equations/solve.xhtml">
      Solve
     </a>
    </li>
    <li>
     <a class="drop" href="equations/solvenumerically.xhtml">
      Solve numerically
     </a>
```

```
</li>
<li>
 <a class="drop" href="equations/rootsofpolynomial.xhtml">
  Roots of polynomial
 </a>
</li>
<li>
 <a class="drop" href="equations/Real roots of polynmial.xhtml">
   Real roots of polynomial
 </a>
</li>
<li>
 <a class="drop" href="equations/solvelinearsystem.xhtml">
  Solve linear system
 </a>
</li>
<li>
 <a class="drop" href="equations/solvealgebraicsystem.xhtml">
  Solve algebraic system
 </a>
</li>
<li>
 <a class="drop" href="equations/eliminatevariable.xhtml">
  Eliminate variable
 </a>
</li>
<li>
 <a class="drop" href="equations/solveode.xhtml">
  Solve ODE
 </a>
</li>
<li>
 <a class="drop" href="equations/initialvalueproblem1.xhtml">
  Initial value problem (1)
 </a>
</li>
<li>
 <a class="drop" href="equations/initialvalueproblem2.xhtml">
  Initial value problem (2)
 </a>
</li>
<li>
 <a class="drop" href="equations/boundaryvalueproblem.xhtml">
  Boundary value problem
 </a>
</li>
<li>
 <a class="drop" href="equations/solveodewithlaplace.xhtml">
  Solve ODE with Laplace
 </a>
</li>
<li>
 <a class="drop" href="equations/atvalue.xhtml">
  At value
```

```
   </a>
  </li>
 </ul>
</li>
<!-- End Equations Menu -->
<!-- Start Algebra Menu -->
<li>
 <a class="toplevel" href="/">
  Algebra<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop" href="algebra/generatematrix.xhtml">
    Generate matrix
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/entermatrix.xhtml">
    Enter matrix
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/invertmatrix.xhtml">
    Invert matrix
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/characteristicpolynomial.xhtml">
    Characteristic polynomial
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/determinant.xhtml">
    Determinant
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/eigenvalues.xhtml">
    Eigenvalues
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/eigenvectors.xhtml">
    Eigenvectors
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/adjointmatrix.xhtml">
    Adjoint matrix
   </a>
  </li>
  <li>
   <a class="drop" href="algebra/transposematrix.xhtml">
    Transpose matrix
```

```
      </a>
     </li>
     <li>
      <a class="drop" href="algebra/makelist.xhtml">
       Make list
      </a>
     </li>
     <li>
      <a class="drop" href="algebra/applytolist.xhtml">
       Apply to list
      </a>
     </li>
     <li>
      <a class="drop" href="algebra/maptolist.xhtml">
       Map to list
      </a>
     </li>
     <li>
      <a class="drop" href="algebra/reducelist.xhtml">
       Reduce list
      </a>
     </li>
     <li>
      <a class="drop" href="algebra/maptomatrix.xhtml">
       Map to matrix
      </a>
     </li>
    </ul>
   </li>
   <!-- End Algebra Menu -->
   <!-- Start Calculus Menu -->
   <li>
    <a class="toplevel" href="/">
     Calculus<!--span class="nabla">&nabla;</span-->
    </a>
    <ul>
     <li>
      <a class="drop" href="calculus/level3.xhtml">
       Level 3
      </a>
      <ul>
       <li>
        <a class="drop" href="calculus/integrate.xhtml">
         Level 3 A
        </a>
       </li>
       <li>
        <a class="drop" href="calculus/integrate.xhtml">
         Level 3 B
        </a>
       </li>
       <li>
        <a class="drop" href="calculus/integrate.xhtml">
         Level 3 C
```

```
   </a>
  </li>
 </ul>
</li>
<li>
 <a class="drop" href="calculus/integrate.xhtml">
  Integrate
 </a>
</li>
<li>
 <a class="drop" href="calculus/rischintegrate.xhtml">
  Risch integrate
 </a>
</li>
<li>
 <a class="drop" href="calculus/changevariable.xhtml">
  Change variable
 </a>
</li>
<li>
 <a class="drop" href="calculus/differentiate.xhtml">
  Differentiate
 </a>
</li>
<li>
 <a class="drop" href="calculus/findlimit.xhtml">
  Find limit
 </a>
</li>
<li>
 <a class="drop" href="calculus/getseries.xhtml">
  Get series
 </a>
</li>
<li>
 <a class="drop" href="calculus/padeapproximation.xhtml">
  Pade approximation
 </a>
</li>
<li>
 <a class="drop" href="calculus/calculussum.xhtml">
  Calculus sum
 </a>
</li>
<li>
 <a class="drop" href="calculus/calculusproduct.xhtml">
  Calculus product
 </a>
</li>
<li>
 <a class="drop" href="calculus/laplacetransform.xhtml">
  Laplace transform
 </a>
</li>
```

```
    <li>
     <a class="drop" href="calculus/inverselaplacetransform.xhtml">
      Inverse Laplace transform
     </a>
    </li>
    <li>
     <a class="drop" href="calculus/greatestcommondivisor.xhtml">
      Greatest common divisor
     </a>
    </li>
    <li>
     <a class="drop" href="calculus/leastcommonmultiple.xhtml">
      Least common multiple
     </a>
    </li>
    <li>
     <a class="drop" href="calculus/dividepolynomials.xhtml">
      Divide polynomials
     </a>
    </li>
    <li>
     <a class="drop" href="calculus/partialfractions.xhtml">
      Partial fractions
     </a>
    </li>
    <li>
     <a class="drop" href="calculus/continuedfractions.xhtml">
      Continued fractions
     </a>
    </li>
   </ul>
  </li>
  <!-- End Calculus Menu -->
  <!-- Start Simplify Menu -->
  <li>
   <a class="toplevel" href="/">
    Simplify<!--span class="nabla">&nabla;</span-->
   </a>
   <ul>
    <li>
     <a class="drop" href="simplify/simplifyexpression.xhtml">
      Simplify expression
     </a>
    </li>
    <li>
     <a class="drop" href="simplify/simplifyradicals.xhtml">
      Simplify radicals
     </a>
    </li>
    <li>
     <a class="drop" href="simplify/factorexpression.xhtml">
      Factor expression
     </a>
    </li>
```

```
<li>
 <a class="drop" href="simplify/factorcomplex.xhtml">
  Factor complex
 </a>
</li>
<li>
 <a class="drop" href="simplify/expandexpression.xhtml">
  Expand expression
 </a>
</li>
<li>
 <a class="drop" href="simplify/expandlogarithms.xhtml">
  Expand logarithms
 </a>
</li>
<li>
 <a class="drop" href="simplify/contractlogarithms.xhtml">
  Contract logarithms
 </a>
</li>
<li>
 <a class="drop" href="simplify/factorialsandgamma.xhtml">
  Factorials and Gamma
 </a>
</li>
<li>
 <a class="drop" href="simplify/trigonometricsimplification.xhtml">
  Trigonometric simplification
 </a>
</li>
<li>
 <a class="drop" href="simplify/complexsimplification.xhtml">
  Complex simplification
 </a>
</li>
<li>
 <a class="drop" href="simplify/subtitute.xhtml">
  Substitute
 </a>
</li>
<li>
 <a class="drop" href="simplify/evaluatenounform.xhtml">
  Evaluate noun form
 </a>
</li>
<li>
 <a class="drop" href="simplify/togglealgebraicflag.xhtml">
  Toggle algebraic flag
 </a>
</li>
<li>
 <a class="drop" href="simplify/addalgebraicequality.xhtml">
  Add algebraic equality
 </a>
```

```
          </li>
          <li>
           <a class="drop" href="simplify/moduluscomputation.xhtml">
            Modulus computation
           </a>
          </li>
         </ul>
        </li>
        <!-- End Simplify Menu -->
        <!-- Start Numeric Menu -->
        <li>
         <a class="toplevel" href="/">
          Numeric<!--span class="nabla">&nabla;</span-->
         </a>
         <ul>
          <li>
           <a class="drop" href="numeric/togglenumericoutput.xhtml">
            Toggle numeric output
           </a>
          </li>
          <li>
           <a class="drop" href="numeric/tofloat.xhtml">
            To float
           </a>
          </li>
          <li>
           <a class="drop" href="numeric/tobigfloat.xhtml">
            To bigfloat
           </a>
          </li>
          <li>
           <a class="drop" href="numeric/setprecision.xhtml">
            Set precision
           </a>
          </li>
         </ul>
        </li>
        <!-- End Simplify Menu -->
       </ul>
      </div>
     </td>
    </tr>
   </table>
  </form>
 </div>

<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="showcall('p1');"
   value="Integer" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p2" class="subbut"
    onclick="showcall('p2');"
    value="(GETDATABASE '|Matrix| 'CONSTRUCTORMODEMAP)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="showcall('p3');"
   value="(progn (setq |$options| '((|operations|))) (|show| '|Integer|))" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
   value="summation(i^2,i=a..b)^(d-c)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
   value="summation(i^2^(d-c),i=a..b)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
   value="sum(operator(f) (i)+1,i=1..n)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
   value="sum(operator(f) (i),i=1..n)+1" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
   value="sum(operator(f) (i)+1,i=1..n)^3" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>

\getchunk{page foot}
```

——–

axiomfonts.xhtml

— axiomfonts.xhtml —

```
\getchunk{standard head}
 </head>
```

```
 <body>
\getchunk{page head}
  <div align="center">Special Font Characters</div>
  <hr/>
<table>
 <tr valign="top">
  <th width="80" align="left">Character</th>
  <th width="80" align="left">Decimal</th>
  <th width="80" align="left">Hex</th>
  <th width="80" align="left">Entity</th>
  <th align="left">Name</th>
 </tr>
 <tr valign="top">
  <td>&#x00391;</td>
  <td>913</td>
  <td>00391</td>
  <td>&amp;Alpha;</td>
  <td>greek capital letter alpha</td>
 </tr>
 <tr valign="top">
  <td>&#x00392;</td>
  <td>914</td>
  <td>00392</td>
  <td>&amp;Beta;</td>
  <td>greek capital letter beta</td>
 </tr>
 <tr valign="top">
  <td>&#x00393;</td>
  <td>915</td>
  <td>00393</td>
  <td>&amp;Gamma;</td>
  <td>greek capital letter gamma</td>
 </tr>
 <tr valign="top">
  <td>&#x00394;</td>
  <td>916</td>
  <td>00394</td>
  <td>&amp;Delta;</td>
  <td>greek capital letter delta</td>
 </tr>
 <tr valign="top">
  <td>&#x00395;</td>
  <td>917</td>
  <td>00395</td>
  <td>&amp;Epsilon;</td>
  <td>greek capital letter epsilon</td>
 </tr>
 <tr valign="top">
  <td>&#x00396;</td>
  <td>918</td>
  <td>00396</td>
  <td>&amp;Zeta;</td>
  <td>greek capital letter zeta</td>
 </tr>
```

```
<tr valign="top">
 <td>&#x00397;</td>
 <td>919</td>
 <td>00397</td>
 <td>&amp;Eta;</td>
 <td>greek capital letter eta</td>
</tr>
<tr valign="top">
 <td>&#x00398;</td>
 <td>920</td>
 <td>00398</td>
 <td>&amp;Theta;</td>
 <td>greek capital letter theta</td>
</tr>
<tr valign="top">
 <td>&#x00399;</td>
 <td>921</td>
 <td>00399</td>
 <td>&amp;Iota;</td>
 <td>greek capital letter iota</td>
</tr>
<tr valign="top">
 <td>&#x0039A;</td>
 <td>922</td>
 <td>0039A</td>
 <td>&amp;Kappa;</td>
 <td>greek capital letter kappa</td>
</tr>
<tr valign="top">
 <td>&#x0039B;</td>
 <td>923</td>
 <td>0039B</td>
 <td>&amp;Lambda;</td>
 <td>greek capital letter lambda</td>
</tr>
<tr valign="top">
 <td>&#x0039C;</td>
 <td>924</td>
 <td>0039C</td>
 <td>&amp;Mu;</td>
 <td>greek capital letter mu</td>
</tr>
<tr valign="top">
 <td>&#x0039D;</td>
 <td>925</td>
 <td>0039D</td>
 <td>&amp;Nu;</td>
 <td>greek capital letter nu</td>
</tr>
<tr valign="top">
 <td>&#x0039E;</td>
 <td>926</td>
 <td>0039E</td>
 <td>&amp;Xi;</td>
```

```
 <td>greek capital letter xi</td>
</tr>
<tr valign="top">
 <td>&#x0039F;</td>
 <td>927</td>
 <td>0039F</td>
 <td>&amp;Omicron;</td>
 <td>greek capital letter omicron</td>
</tr>
<tr valign="top">
 <td>&#x003A0;</td>
 <td>928</td>
 <td>003A0</td>
 <td>&amp;Pi;</td>
 <td>greek capital letter pi</td>
</tr>
<tr valign="top">
 <td>&#x003A1;</td>
 <td>929</td>
 <td>003A1</td>
 <td>&amp;Rho;</td>
 <td>greek capital letter rho</td>
</tr>
<tr valign="top">
 <td>&#x003A3;</td>
 <td>931</td>
 <td>003A3</td>
 <td>&amp;Sigma;</td>
 <td>greek capital letter sigma</td>
</tr>
<tr valign="top">
 <td>&#x003A4;</td>
 <td>932</td>
 <td>003A4</td>
 <td>&amp;Tau;</td>
 <td>greek capital letter tau</td>
</tr>
<tr valign="top">
 <td>&#x003A5;</td>
 <td>933</td>
 <td>003A5</td>
 <td>&amp;Upsilon;</td>
 <td>greek capital letter upsilon</td>
</tr>
<tr valign="top">
 <td>&#x003A6;</td>
 <td>934</td>
 <td>003A6</td>
 <td>&amp;Phi;</td>
 <td>greek capital letter phi</td>
</tr>
<tr valign="top">
 <td>&#x003A7;</td>
 <td>935</td>
```

```
  <td>003A7</td>
  <td>&amp;Chi;</td>
  <td>greek capital letter chi</td>
 </tr>
 <tr valign="top">
  <td>&#x003A8;</td>
  <td>936</td>
  <td>003A8</td>
  <td>&amp;Psi;</td>
  <td>greek capital letter psi</td>
 </tr>
 <tr valign="top">
  <td>&#x003A9;</td>
  <td>937</td>
  <td>003A9</td>
  <td>&amp;Omega;</td>
  <td>greek capital letter omega</td>
 </tr>
 <tr valign="top">
  <td>&#x003B1;</td>
  <td>945</td>
  <td>003B1</td>
  <td>&amp;alpha;</td>
  <td>greek small letter alpha</td>
 </tr>
 <tr valign="top">
  <td>&#x003B2;</td>
  <td>946</td>
  <td>003B2</td>
  <td>&amp;beta;</td>
  <td>greek small letter beta</td>
 </tr>
 <tr valign="top">
  <td>&#x003B3;</td>
  <td>947</td>
  <td>003B3</td>
  <td>&amp;gamma;</td>
  <td>greek small letter gamma</td>
 </tr>
 <tr valign="top">
  <td>&#x003B4;</td>
  <td>948</td>
  <td>003B4</td>
  <td>&amp;delta;</td>
  <td>greek small letter delta</td>
 </tr>
 <tr valign="top">
  <td>&#x003B5;</td>
  <td>949</td>
  <td>003B5</td>
  <td>&amp;epsilon;</td>
  <td>greek small letter epsilon</td>
 </tr>
 <tr valign="top">
```

```
 <td>&#x003B6;</td>
 <td>950</td>
 <td>003B6</td>
 <td>&amp;zeta;</td>
 <td>greek small letter zeta</td>
</tr>
<tr valign="top">
 <td>&#x003B7;</td>
 <td>951</td>
 <td>003B7</td>
 <td>&amp;eta;</td>
 <td>greek small letter eta</td>
</tr>
<tr valign="top">
 <td>&#x003B8;</td>
 <td>952</td>
 <td>003B8</td>
 <td>&amp;theta;</td>
 <td>greek small letter theta</td>
</tr>
<tr valign="top">
 <td>&#x003B9;</td>
 <td>953</td>
 <td>003B9</td>
 <td>&amp;iota;</td>
 <td>greek small letter iota</td>
</tr>
<tr valign="top">
 <td>&#x003BA;</td>
 <td>954</td>
 <td>003BA</td>
 <td>&amp;kappa;</td>
 <td>greek small letter kappa</td>
</tr>
<tr valign="top">
 <td>&#x003BB;</td>
 <td>955</td>
 <td>003BB</td>
 <td>&amp;lambda;</td>
 <td>greek small letter lambda</td>
</tr>
<tr valign="top">
 <td>&#x003BC;</td>
 <td>956</td>
 <td>003BC</td>
 <td>&amp;mu;</td>
 <td>greek small letter mu</td>
</tr>
<tr valign="top">
 <td>&#x003BD;</td>
 <td>957</td>
 <td>003BD</td>
 <td>&amp;nu;</td>
 <td>greek small letter nu</td>
```

```
</tr>
<tr valign="top">
 <td>&#x003BE;</td>
 <td>958</td>
 <td>003BE</td>
 <td>&amp;xi;</td>
 <td>greek small letter xi</td>
</tr>
<tr valign="top">
 <td>&#x003BF;</td>
 <td>959</td>
 <td>003BF</td>
 <td>&amp;omicron;</td>
 <td>greek small letter omicron</td>
</tr>
<tr valign="top">
 <td>&#x003C0;</td>
 <td>960</td>
 <td>003C0</td>
 <td>&amp;pi;</td>
 <td>greek small letter pi</td>
</tr>
<tr valign="top">
 <td>&#x003C1;</td>
 <td>961</td>
 <td>003C1</td>
 <td>&amp;rho;</td>
 <td>greek small letter rho</td>
</tr>
<tr valign="top">
 <td>&#x003C2;</td>
 <td>962</td>
 <td>003C2</td>
 <td>&amp;sigmaf;</td>
 <td>greek small letter final sigma</td>
</tr>
<tr valign="top">
 <td>&#x003C3;</td>
 <td>963</td>
 <td>003C3</td>
 <td>&amp;sigma;</td>
 <td>greek small letter sigma</td>
</tr>
<tr valign="top">
 <td>&#x003C4;</td>
 <td>964</td>
 <td>003C4</td>
 <td>&amp;tau;</td>
 <td>greek small letter tau</td>
</tr>
<tr valign="top">
 <td>&#x003C5;</td>
 <td>965</td>
 <td>003C5</td>
```

```
 <td>&amp;upsilon;</td>
 <td>greek small letter upsilon</td>
</tr>
<tr valign="top">
 <td>&#x003C6;</td>
 <td>966</td>
 <td>003C6</td>
 <td>&amp;phi;</td>
 <td>greek small letter phi</td>
</tr>
<tr valign="top">
 <td>&#x003C7;</td>
 <td>967</td>
 <td>003C7</td>
 <td>&amp;chi;</td>
 <td>greek small letter chi</td>
</tr>
<tr valign="top">
 <td>&#x003C8;</td>
 <td>968</td>
 <td>003C8</td>
 <td>&amp;psi;</td>
 <td>greek small letter psi</td>
</tr>
<tr valign="top">
 <td>&#x003C9;</td>
 <td>969</td>
 <td>003C9</td>
 <td>&amp;omega;</td>
 <td>greek small letter omega</td>
</tr>
<tr><td>----</td><td>----</td><td>----</td><td>----</td><td>----</td></tr>
<tr valign="top">
 <td>&#x000AF;</td>
 <td>175</td>
 <td>000AF</td>
 <td>&amp;macr;</td>
 <td>macron</td>
</tr>
<tr valign="top">
 <td>&#x000B1;</td>
 <td>177</td>
 <td>000B1</td>
 <td>&amp;plusmn;</td>
 <td>plus-or-minus sign</td>
</tr>
<tr valign="top">
 <td>&#x000D7;</td>
 <td>215</td>
 <td>000D7</td>
 <td></td>
 <td>multiplication sign</td>
</tr>
<tr valign="top">
```

```
 <td>&#x000E8;</td>
 <td>232</td>
 <td>000E8</td>
 <td>&amp;egrave;</td>
 <td>latin small letter e with grave</td>
</tr>
<tr valign="top">
 <td>&#x003C0;</td>
 <td>960</td>
 <td>003C0</td>
 <td>&amp;pi;</td>
 <td>greek small letter pi</td>
</tr>
<tr valign="top">
 <td>&#x003D5;</td>
 <td>981</td>
 <td>003D5</td>
 <td></td>
 <td>greek phi symbol</td>
</tr>
<tr valign="top">
 <td>&#x02026;</td>
 <td>8230</td>
 <td>02026</td>
 <td>&amp;hellip;</td>
 <td>horizontal ellipsis</td>
</tr>
<tr valign="top">
 <td>&#x022EF;</td>
 <td>8943</td>
 <td>022EF</td>
 <td></td>
 <td>midline horizontal ellipsis</td>
</tr>
<tr valign="top">
 <td>&#x02032;</td>
 <td>8242</td>
 <td>02032</td>
 <td>&amp;prime;</td>
 <td>prime</td>
</tr>
<tr valign="top">
 <td>&#x02061;</td>
 <td>8289</td>
 <td>02061</td>
 <td></td>
 <td>function application</td>
</tr>
<tr valign="top">
 <td>&#x02062;</td>
 <td>8290</td>
 <td>02062</td>
 <td></td>
 <td>invisible times</td>
```

```
</tr>
<tr valign="top">
 <td>&#x02102;</td>
 <td>8450</td>
 <td>02102</td>
 <td></td>
 <td>doube-struck captial c</td>
</tr>
<tr valign="top">
 <td>&#x0210D;</td>
 <td>8461</td>
 <td>0210D</td>
 <td></td>
 <td>double-struck captial h</td>
</tr>
<tr valign="top">
 <td>&#x02111;</td>
 <td>8465</td>
 <td>02111</td>
 <td>&amp;image;</td>
 <td>black-letter captial i</td>
</tr>
<tr valign="top">
 <td>&#x02113;</td>
 <td>8467</td>
 <td>02113</td>
 <td></td>
 <td>script small l</td>
</tr>
<tr valign="top">
 <td>&#x02115;</td>
 <td>8469</td>
 <td>02115</td>
 <td></td>
 <td>double-struck captial n</td>
</tr>
<tr valign="top">
 <td>&#x02119;</td>
 <td>8473</td>
 <td>02119</td>
 <td></td>
 <td>double-struck captial p</td>
</tr>
<tr valign="top">
 <td>&#x0211A;</td>
 <td>8474</td>
 <td>0211A</td>
 <td></td>
 <td>double-struck captial q</td>
</tr>
<tr valign="top">
 <td>&#x0211C;</td>
 <td>8476</td>
 <td>0211C</td>
```

```
 <td>&amp;real;</td>
 <td>black-letter captial r</td>
</tr>
<tr valign="top">
 <td>&#x0211D;</td>
 <td>8477</td>
 <td>0211D</td>
 <td></td>
 <td>double-struck captial r</td>
</tr>
<tr valign="top">
 <td>&#x02124;</td>
 <td>8484</td>
 <td>02124</td>
 <td></td>
 <td>double-struck captial z</td>
</tr>
<tr valign="top">
 <td>&#x02145;</td>
 <td>8517</td>
 <td>02145</td>
 <td></td>
 <td>doube-struck captial d</td>
</tr>
<tr valign="top">
 <td>&#x02146;</td>
 <td>8518</td>
 <td>02146</td>
 <td></td>
 <td>double-struck italic small d</td>
</tr>
<tr valign="top">
 <td>&#x02147;</td>
 <td>8519</td>
 <td>02147</td>
 <td></td>
 <td>double-struck italic small e</td>
</tr>
<tr valign="top">
 <td>&#x02148;</td>
 <td>8520</td>
 <td>02148</td>
 <td></td>
 <td>double-struck italic small i</td>
</tr>
<tr valign="top">
 <td>&#x02192;</td>
 <td>8594</td>
 <td>02192</td>
 <td>&amp;rarr;</td>
 <td>rightwards arrow</td>
</tr>
<tr><td>----</td><td>----</td><td>----</td><td>----</td><td>----</td></tr>
<tr valign="top">
```

```
 <td>&#8704;</td>
 <td>8704</td>
 <td>2200</td>
 <td>&amp;forall;</td>
 <td>for all</td>
</tr>
<tr valign="top">
 <td>&#8705;</td>
 <td>8705</td>
 <td>2201</td>
 <td></td>
 <td>complement</td>
</tr>
<tr valign="top">
 <td>&#8706;</td>
 <td>8706</td>
 <td>2202</td>
 <td>&amp;part;</td>
 <td>partial differential</td>
</tr>
<tr valign="top">
 <td>&#8707;</td>
 <td>8707</td>
 <td>2203</td>
 <td>&amp;exist;</td>
 <td>there exists</td>
</tr>
<tr valign="top">
 <td>&#8708;</td>
 <td>8708</td>
 <td>2204</td>
 <td></td>
 <td>there does not exist</td>
</tr>
<tr valign="top">
 <td>&#8709;</td>
 <td>8709</td>
 <td>2205</td>
 <td>&amp;empty;</td>
 <td>empty set</td>
</tr>
<tr valign="top">
 <td>&#8710;</td>
 <td>8710</td>
 <td>2206</td>
 <td></td>
 <td>increment</td>
</tr>
<tr valign="top">
 <td>&#8711;</td>
 <td>8711</td>
 <td>2207</td>
 <td>&amp;nabla;</td>
 <td>nabla</td>
```

```
</tr>
<tr valign="top">
 <td>&#8712;</td>
 <td>8712</td>
 <td>2208</td>
 <td>&amp;isin;</td>
 <td>element of</td>
</tr>
<tr valign="top">
 <td>&#8713;</td>
 <td>8713</td>
 <td>2209</td>
 <td>&amp;notin;</td>
 <td>not an element of</td>
</tr>
<tr valign="top">
 <td>&#8714;</td>
 <td>8714</td>
 <td>220A</td>
 <td></td>
 <td>small element of</td>
</tr>
<tr valign="top">
 <td>&#8715;</td>
 <td>8715</td>
 <td>220B</td>
 <td>&amp;ni;</td>
 <td>contains as member</td>
</tr>
<tr valign="top">
 <td>&#8716;</td>
 <td>8716</td>
 <td>220C</td>
 <td></td>
 <td>does not contain as member</td>
</tr>
<tr valign="top">
 <td>&#8717;</td>
 <td>8717</td>
 <td>220D</td>
 <td></td>
 <td>small contains as member</td>
</tr>
<tr valign="top">
 <td>&#8718;</td>
 <td>8718</td>
 <td>220E</td>
 <td></td>
 <td>end of proof</td>
</tr>
<tr valign="top">
 <td>&#8719;</td>
 <td>8719</td>
 <td>220F</td>
```

```
  <td>&amp;prod;</td>
  <td>n-ary product</td>
 </tr>
 <tr valign="top">
  <td>&#8720;</td>
  <td>8720</td>
  <td>2210</td>
  <td></td>
  <td>n-ary coproduct</td>
 </tr>
 <tr valign="top">
  <td>&#8721;</td>
  <td>8721</td>
  <td>2211</td>
  <td>&amp;sum;</td>
  <td>n-ary summation</td>
 </tr>
 <tr valign="top">
  <td>&#8722;</td>
  <td>8722</td>
  <td>2212</td>
  <td>&amp;minus;</td>
  <td>minus sign</td>
 </tr>
 <tr valign="top">
  <td>&#8723;</td>
  <td>8723</td>
  <td>2213</td>
  <td></td>
  <td>minus-or-plus sign</td>
 </tr>
 <tr valign="top">
  <td>&#8724;</td>
  <td>8724</td>
  <td>2214</td>
  <td></td>
  <td>dot plus</td>
 </tr>
 <tr valign="top">
  <td>&#8725;</td>
  <td>8725</td>
  <td>2215</td>
  <td></td>
  <td>division slash</td>
 </tr>
 <tr valign="top">
  <td>&#8726;</td>
  <td>8726</td>
  <td>2216</td>
  <td></td>
  <td>set minus</td>
 </tr>
 <tr valign="top">
  <td>&#8727;</td>
```

```
 <td>8727</td>
 <td>2217</td>
 <td>&amp;lowast;</td>
 <td>asterisk operator</td>
</tr>
<tr valign="top">
 <td>&#8728;</td>
 <td>8728</td>
 <td>2218</td>
 <td></td>
 <td>ring operator</td>
</tr>
<tr valign="top">
 <td>&#8729;</td>
 <td>8729</td>
 <td>2219</td>
 <td></td>
 <td>bullet operator</td>
</tr>
<tr valign="top">
 <td>&#8730;</td>
 <td>8730</td>
 <td>221A</td>
 <td>&amp;radic;</td>
 <td>square root</td>
</tr>
<tr valign="top">
 <td>&#8731;</td>
 <td>8731</td>
 <td>221B</td>
 <td></td>
 <td>cube root</td>
</tr>
<tr valign="top">
 <td>&#8732;</td>
 <td>8732</td>
 <td>221C</td>
 <td></td>
 <td>fourth root</td>
</tr>
<tr valign="top">
 <td>&#8733;</td>
 <td>8733</td>
 <td>221D</td>
 <td>&amp;prop;</td>
 <td>proportional to</td>
</tr>
<tr valign="top">
 <td>&#8734;</td>
 <td>8734</td>
 <td>221E</td>
 <td>&amp;infin;</td>
 <td>infinity</td>
</tr>
```

```
<tr valign="top">
 <td>&#8735;</td>
 <td>8735</td>
 <td>221F</td>
 <td></td>
 <td>right angle</td>
</tr>
<tr valign="top">
 <td>&#8736;</td>
 <td>8736</td>
 <td>2220</td>
 <td>&amp;ang;</td>
 <td>angle</td>
</tr>
<tr valign="top">
 <td>&#8737;</td>
 <td>8737</td>
 <td>2221</td>
 <td></td>
 <td>measured angle</td>
</tr>
<tr valign="top">
 <td>&#8738;</td>
 <td>8738</td>
 <td>2222</td>
 <td></td>
 <td>spherical angle</td>
</tr>
<tr valign="top">
 <td>&#8739;</td>
 <td>8739</td>
 <td>2223</td>
 <td></td>
 <td>divides</td>
</tr>
<tr valign="top">
 <td>&#8740;</td>
 <td>8740</td>
 <td>2224</td>
 <td></td>
 <td>does not divide</td>
</tr>
<tr valign="top">
 <td>&#8741;</td>
 <td>8741</td>
 <td>2225</td>
 <td></td>
 <td>parallel to</td>
</tr>
<tr valign="top">
 <td>&#8742;</td>
 <td>8742</td>
 <td>2226</td>
 <td></td>
```

```
 <td>not parallel to</td>
</tr>
<tr valign="top">
 <td>&#8743;</td>
 <td>8743</td>
 <td>2227</td>
 <td>&amp;and;</td>
 <td>logical and</td>
</tr>
<tr valign="top">
 <td>&#8744;</td>
 <td>8744</td>
 <td>2228</td>
 <td>&amp;or;</td>
 <td>logical or</td>
</tr>
<tr valign="top">
 <td>&#8745;</td>
 <td>8745</td>
 <td>2229</td>
 <td>&amp;cap;</td>
 <td>intersection</td>
</tr>
<tr valign="top">
 <td>&#8746;</td>
 <td>8746</td>
 <td>222A</td>
 <td>&amp;cup;</td>
 <td>union</td>
</tr>
<tr valign="top">
 <td>&#8747;</td>
 <td>8747</td>
 <td>222B</td>
 <td>&amp;int;</td>
 <td>integral</td>
</tr>
<tr valign="top">
 <td>&#8748;</td>
 <td>8748</td>
 <td>222C</td>
 <td></td>
 <td>double integral</td>
</tr>
<tr valign="top">
 <td>&#8749;</td>
 <td>8749</td>
 <td>222D</td>
 <td></td>
 <td>triple integral</td>
</tr>
<tr valign="top">
 <td>&#8750;</td>
 <td>8750</td>
```

```
 <td>222E</td>
 <td></td>
 <td>contour integral</td>
</tr>
<tr valign="top">
 <td>&#8751;</td>
 <td>8751</td>
 <td>222F</td>
 <td></td>
 <td>surface integral</td>
</tr>
<tr valign="top">
 <td>&#8752;</td>
 <td>8752</td>
 <td>2230</td>
 <td></td>
 <td>volume integral</td>
</tr>
<tr valign="top">
 <td>&#8753;</td>
 <td>8753</td>
 <td>2231</td>
 <td></td>
 <td>clockwise integral</td>
</tr>
<tr valign="top">
 <td>&#8754;</td>
 <td>8754</td>
 <td>2232</td>
 <td></td>
 <td>clockwise contour integral</td>
</tr>
<tr valign="top">
 <td>&#8755;</td>
 <td>8755</td>
 <td>2233</td>
 <td></td>
 <td>anticlockwise contour integral</td>
</tr>
<tr valign="top">
 <td>&#8756;</td>
 <td>8756</td>
 <td>2234</td>
 <td>&amp;there4;</td>
 <td>therefore</td>
</tr>
<tr valign="top">
 <td>&#8757;</td>
 <td>8757</td>
 <td>2235</td>
 <td></td>
 <td>because</td>
</tr>
<tr valign="top">
```

```
 <td>&#8758;</td>
 <td>8758</td>
 <td>2236</td>
 <td></td>
 <td>ratio</td>
</tr>
<tr valign="top">
 <td>&#8759;</td>
 <td>8759</td>
 <td>2237</td>
 <td></td>
 <td>proportion</td>
</tr>
<tr valign="top">
 <td>&#8760;</td>
 <td>8760</td>
 <td>2238</td>
 <td></td>
 <td>dot minus</td>
</tr>
<tr valign="top">
 <td>&#8761;</td>
 <td>8761</td>
 <td>2239</td>
 <td></td>
 <td>excess</td>
</tr>
<tr valign="top">
 <td>&#8762;</td>
 <td>8762</td>
 <td>223A</td>
 <td></td>
 <td>geometric proportion</td>
</tr>
<tr valign="top">
 <td>&#8763;</td>
 <td>8763</td>
 <td>223B</td>
 <td></td>
 <td>homothetic</td>
</tr>
<tr valign="top">
 <td>&#8764;</td>
 <td>8764</td>
 <td>223C</td>
 <td>&amp;sim;</td>
 <td>tilde operator</td>
</tr>
<tr valign="top">
 <td>&#8765;</td>
 <td>8765</td>
 <td>223D</td>
 <td></td>
 <td>reversed tilde</td>
```

```
</tr>
<tr valign="top">
 <td>&#8766;</td>
 <td>8766</td>
 <td>223E</td>
 <td></td>
 <td>inverted lazy S</td>
</tr>
<tr valign="top">
 <td>&#8767;</td>
 <td>8767</td>
 <td>223F</td>
 <td></td>
 <td>sine wave</td>
</tr>
<tr valign="top">
 <td>&#8768;</td>
 <td>8768</td>
 <td>2240</td>
 <td></td>
 <td>wreath products</td>
</tr>
<tr valign="top">
 <td>&#8769;</td>
 <td>8769</td>
 <td>2241</td>
 <td></td>
 <td>not tilde</td>
</tr>
<tr valign="top">
 <td>&#8770;</td>
 <td>8770</td>
 <td>2242</td>
 <td></td>
 <td>minus tilde</td>
</tr>
<tr valign="top">
 <td>&#8771;</td>
 <td>8771</td>
 <td>2243</td>
 <td></td>
 <td>asymptotically equal to</td>
</tr>
<tr valign="top">
 <td>&#8772;</td>
 <td>8772</td>
 <td>2244</td>
 <td></td>
 <td>not asymptotically equal to</td>
</tr>
<tr valign="top">
 <td>&#8773;</td>
 <td>8773</td>
 <td>2245</td>
```

```
 <td>&amp;cong;</td>
 <td>approximately equal to</td>
</tr>
<tr valign="top">
 <td>&#8774;</td>
 <td>8774</td>
 <td>2246</td>
 <td></td>
 <td>approximately but not actually equal to</td>
</tr>
<tr valign="top">
 <td>&#8775;</td>
 <td>8775</td>
 <td>2247</td>
 <td></td>
 <td>neither approximately nor actually equal to</td>
</tr>
<tr valign="top">
 <td>&#8776;</td>
 <td>8776</td>
 <td>2248</td>
 <td>&amp;asymp;</td>
 <td>almost equal to</td>
</tr>
<tr valign="top">
 <td>&#8777;</td>
 <td>8777</td>
 <td>2249</td>
 <td></td>
 <td>not almost equal to</td>
</tr>
<tr valign="top">
 <td>&#8778;</td>
 <td>8778</td>
 <td>224A</td>
 <td></td>
 <td>almost equal or equal to</td>
</tr>
<tr valign="top">
 <td>&#8779;</td>
 <td>8779</td>
 <td>224B</td>
 <td></td>
 <td>triple tilde</td>
</tr>
<tr valign="top">
 <td>&#8780;</td>
 <td>8780</td>
 <td>224C</td>
 <td></td>
 <td>all equal to</td>
</tr>
<tr valign="top">
 <td>&#8781;</td>
```

```
 <td>8781</td>
 <td>224D</td>
 <td></td>
 <td>equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8782;</td>
 <td>8782</td>
 <td>224E</td>
 <td></td>
 <td>geometrically equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8783;</td>
 <td>8783</td>
 <td>224F</td>
 <td></td>
 <td>difference between</td>
</tr>
<tr valign="top">
 <td>&#8784;</td>
 <td>8784</td>
 <td>2250</td>
 <td></td>
 <td>approaches the limit</td>
</tr>
<tr valign="top">
 <td>&#8785;</td>
 <td>8785</td>
 <td>2251</td>
 <td></td>
 <td>geometrically equal to</td>
</tr>
<tr valign="top">
 <td>&#8786;</td>
 <td>8786</td>
 <td>2252</td>
 <td></td>
 <td>approximately equal to or the image of</td>
</tr>
<tr valign="top">
 <td>&#8787;</td>
 <td>8787</td>
 <td>2253</td>
 <td></td>
 <td>image of or approximately equal to</td>
</tr>
<tr valign="top">
 <td>&#8788;</td>
 <td>8788</td>
 <td>2254</td>
 <td></td>
 <td>colon equals</td>
</tr>
```

```
<tr valign="top">
 <td>&#8789;</td>
 <td>8789</td>
 <td>2255</td>
 <td></td>
 <td>equals colon</td>
</tr>
<tr valign="top">
 <td>&#8790;</td>
 <td>8790</td>
 <td>2256</td>
 <td></td>
 <td>ring in equal to</td>
</tr>
<tr valign="top">
 <td>&#8791;</td>
 <td>8791</td>
 <td>2257</td>
 <td></td>
 <td>ring equal to</td>
</tr>
<tr valign="top">
 <td>&#8792;</td>
 <td>8792</td>
 <td>2258</td>
 <td></td>
 <td>corresponds to</td>
</tr>
<tr valign="top">
 <td>&#8793;</td>
 <td>8793</td>
 <td>2259</td>
 <td></td>
 <td>estimates</td>
</tr>
<tr valign="top">
 <td>&#8794;</td>
 <td>8794</td>
 <td>225A</td>
 <td></td>
 <td>equiangular to</td>
</tr>
<tr valign="top">
 <td>&#8795;</td>
 <td>8795</td>
 <td>225B</td>
 <td></td>
 <td>star equals</td>
</tr>
<tr valign="top">
 <td>&#8796;</td>
 <td>8796</td>
 <td>225C</td>
 <td></td>
```

```
 <td>delta equal to</td>
</tr>
<tr valign="top">
 <td>&#8797;</td>
 <td>8797</td>
 <td>225D</td>
 <td></td>
 <td>equal to by definition</td>
</tr>
<tr valign="top">
 <td>&#8798;</td>
 <td>8798</td>
 <td>225E</td>
 <td></td>
 <td>measured by</td>
</tr>
<tr valign="top">
 <td>&#8799;</td>
 <td>8799</td>
 <td>225F</td>
 <td></td>
 <td>questioned equal to</td>
</tr>
<tr valign="top">
 <td>&#8800;</td>
 <td>8800</td>
 <td>2260</td>
 <td>&amp;ne;</td>
 <td>not equal to</td>
</tr>
<tr valign="top">
 <td>&#8801;</td>
 <td>8801</td>
 <td>2261</td>
 <td>&amp;equiv;</td>
 <td>identical to</td>
</tr>
<tr valign="top">
 <td>&#8802;</td>
 <td>8802</td>
 <td>2262</td>
 <td></td>
 <td>not identical to</td>
</tr>
<tr valign="top">
 <td>&#8803;</td>
 <td>8803</td>
 <td>2263</td>
 <td></td>
 <td>strictly equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8804;</td>
 <td>8804</td>
```

```
 <td>2264</td>
 <td>&amp;le;</td>
 <td>less-than or equal to</td>
</tr>
<tr valign="top">
 <td>&#8805;</td>
 <td>8805</td>
 <td>2265</td>
 <td>&amp;ge;</td>
 <td>greater-than or equal to</td>
</tr>
<tr valign="top">
 <td>&#8806;</td>
 <td>8806</td>
 <td>2266</td>
 <td></td>
 <td>less-than over equal to</td>
</tr>
<tr valign="top">
 <td>&#8807;</td>
 <td>8807</td>
 <td>2267</td>
 <td></td>
 <td>greater-than over equal to</td>
</tr>
<tr valign="top">
 <td>&#8808;</td>
 <td>8808</td>
 <td>2268</td>
 <td></td>
 <td>less-than but not equal to</td>
</tr>
<tr valign="top">
 <td>&#8809;</td>
 <td>8809</td>
 <td>2269</td>
 <td></td>
 <td>greater-than but not equal to</td>
</tr>
<tr valign="top">
 <td>&#8810;</td>
 <td>8810</td>
 <td>226A</td>
 <td></td>
 <td>much less-than</td>
</tr>
<tr valign="top">
 <td>&#8811;</td>
 <td>8811</td>
 <td>226B</td>
 <td></td>
 <td>much greater-than</td>
</tr>
<tr valign="top">
```

```
 <td>&#8812;</td>
 <td>8812</td>
 <td>226C</td>
 <td></td>
 <td>between</td>
</tr>
<tr valign="top">
 <td>&#8813;</td>
 <td>8813</td>
 <td>226D</td>
 <td></td>
 <td>not equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8814;</td>
 <td>8814</td>
 <td>226E</td>
 <td></td>
 <td>not less-than</td>
</tr>
<tr valign="top">
 <td>&#8815;</td>
 <td>8815</td>
 <td>226F</td>
 <td></td>
 <td>not greater-than</td>
</tr>
<tr valign="top">
 <td>&#8816;</td>
 <td>8816</td>
 <td>2270</td>
 <td></td>
 <td>neither less-than nor equal to</td>
</tr>
<tr valign="top">
 <td>&#8817;</td>
 <td>8817</td>
 <td>2271</td>
 <td></td>
 <td>neither greater-than nor equal to</td>
</tr>
<tr valign="top">
 <td>&#8818;</td>
 <td>8818</td>
 <td>2272</td>
 <td></td>
 <td>less-than or equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8819;</td>
 <td>8819</td>
 <td>2273</td>
 <td></td>
 <td>greater-than or equivalent to</td>
```

```
</tr>
<tr valign="top">
 <td>&#8820;</td>
 <td>8820</td>
 <td>2274</td>
 <td></td>
 <td>neither less-than nor equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8821;</td>
 <td>8821</td>
 <td>2275</td>
 <td></td>
 <td>neither greater-than nor equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8822;</td>
 <td>8822</td>
 <td>2276</td>
 <td></td>
 <td>less-than or greater-than</td>
</tr>
<tr valign="top">
 <td>&#8823;</td>
 <td>8823</td>
 <td>2277</td>
 <td></td>
 <td>greater-than or less-than</td>
</tr>
<tr valign="top">
 <td>&#8824;</td>
 <td>8824</td>
 <td>2278</td>
 <td></td>
 <td>neither less-than nor greater-than</td>
</tr>
<tr valign="top">
 <td>&#8825;</td>
 <td>8825</td>
 <td>2279</td>
 <td></td>
 <td>neither greater-than nor less-than</td>
</tr>
<tr valign="top">
 <td>&#8826;</td>
 <td>8826</td>
 <td>227A</td>
 <td></td>
 <td>precedes</td>
</tr>
<tr valign="top">
 <td>&#8827;</td>
 <td>8827</td>
 <td>227B</td>
```

```
 <td></td>
 <td>succeeds</td>
</tr>
<tr valign="top">
 <td>&#8828;</td>
 <td>8828</td>
 <td>227C</td>
 <td></td>
 <td>precedes or equal to</td>
</tr>
<tr valign="top">
 <td>&#8829;</td>
 <td>8829</td>
 <td>227D</td>
 <td></td>
 <td>succeeds or equal to</td>
</tr>
<tr valign="top">
 <td>&#8830;</td>
 <td>8830</td>
 <td>227E</td>
 <td></td>
 <td>precedes or equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8831;</td>
 <td>8831</td>
 <td>227F</td>
 <td></td>
 <td>succeeds or equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8832;</td>
 <td>8832</td>
 <td>2280</td>
 <td></td>
 <td>does not precede</td>
</tr>
<tr valign="top">
 <td>&#8833;</td>
 <td>8833</td>
 <td>2281</td>
 <td></td>
 <td>does not succeed</td>
</tr>
<tr valign="top">
 <td>&#8834;</td>
 <td>8834</td>
 <td>2282</td>
 <td>&amp;sub;</td>
 <td>subset of</td>
</tr>
<tr valign="top">
 <td>&#8835;</td>
```

```
 <td>8835</td>
 <td>2283</td>
 <td>&amp;sup;</td>
 <td>superset of</td>
</tr>
<tr valign="top">
 <td>&#8836;</td>
 <td>8836</td>
 <td>2284</td>
 <td>&amp;nsub;</td>
 <td>not a subset of</td>
</tr>
<tr valign="top">
 <td>&#8837;</td>
 <td>8837</td>
 <td>2285</td>
 <td></td>
 <td>not a superset of</td>
</tr>
<tr valign="top">
 <td>&#8838;</td>
 <td>8838</td>
 <td>2286</td>
 <td>&amp;sube;</td>
 <td>subset of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8839;</td>
 <td>8839</td>
 <td>2287</td>
 <td>&amp;supe;</td>
 <td>superset of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8840;</td>
 <td>8840</td>
 <td>2288</td>
 <td></td>
 <td>neither a subset of nor equal to</td>
</tr>
<tr valign="top">
 <td>&#8841;</td>
 <td>8841</td>
 <td>2289</td>
 <td></td>
 <td>neither a superset of nor equal to</td>
</tr>
<tr valign="top">
 <td>&#8842;</td>
 <td>8842</td>
 <td>228A</td>
 <td></td>
 <td>subset of with not equal to</td>
</tr>
```

```
<tr valign="top">
 <td>&#8843;</td>
 <td>8843</td>
 <td>228B</td>
 <td></td>
 <td>superset of with not equal to</td>
</tr>
<tr valign="top">
 <td>&#8844;</td>
 <td>8844</td>
 <td>228C</td>
 <td></td>
 <td>multiset</td>
</tr>
<tr valign="top">
 <td>&#8845;</td>
 <td>8845</td>
 <td>228D</td>
 <td></td>
 <td>multiset multiplication</td>
</tr>
<tr valign="top">
 <td>&#8846;</td>
 <td>8846</td>
 <td>228E</td>
 <td></td>
 <td>multiset union</td>
</tr>
<tr valign="top">
 <td>&#8847;</td>
 <td>8847</td>
 <td>228F</td>
 <td></td>
 <td>square image of</td>
</tr>
<tr valign="top">
 <td>&#8848;</td>
 <td>8848</td>
 <td>2290</td>
 <td></td>
 <td>square original of</td>
</tr>
<tr valign="top">
 <td>&#8849;</td>
 <td>8849</td>
 <td>2291</td>
 <td></td>
 <td>square image of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8850;</td>
 <td>8850</td>
 <td>2292</td>
 <td></td>
```

```
 <td>square original of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8851;</td>
 <td>8851</td>
 <td>2293</td>
 <td></td>
 <td>square cap</td>
</tr>
<tr valign="top">
 <td>&#8852;</td>
 <td>8852</td>
 <td>2294</td>
 <td></td>
 <td>square cup</td>
</tr>
<tr valign="top">
 <td>&#8853;</td>
 <td>8853</td>
 <td>2295</td>
 <td>&amp;oplus;</td>
 <td>circled plus</td>
</tr>
<tr valign="top">
 <td>&#8854;</td>
 <td>8854</td>
 <td>2296</td>
 <td></td>
 <td>circled minus</td>
</tr>
<tr valign="top">
 <td>&#8855;</td>
 <td>8855</td>
 <td>2297</td>
 <td>&amp;otimes;</td>
 <td>circled times</td>
</tr>
<tr valign="top">
 <td>&#8856;</td>
 <td>8856</td>
 <td>2298</td>
 <td></td>
 <td>circled division slash</td>
</tr>
<tr valign="top">
 <td>&#8857;</td>
 <td>8857</td>
 <td>2299</td>
 <td></td>
 <td>circled dot operator</td>
</tr>
<tr valign="top">
 <td>&#8858;</td>
 <td>8858</td>
```

```
 <td>229A</td>
 <td></td>
 <td>circled ring operator</td>
</tr>
<tr valign="top">
 <td>&#8859;</td>
 <td>8859</td>
 <td>229B</td>
 <td></td>
 <td>circled asterisk operator</td>
</tr>
<tr valign="top">
 <td>&#8860;</td>
 <td>8860</td>
 <td>229C</td>
 <td></td>
 <td>circled equals</td>
</tr>
<tr valign="top">
 <td>&#8861;</td>
 <td>8861</td>
 <td>229D</td>
 <td></td>
 <td>circled dash</td>
</tr>
<tr valign="top">
 <td>&#8862;</td>
 <td>8862</td>
 <td>229E</td>
 <td></td>
 <td>squared plus</td>
</tr>
<tr valign="top">
 <td>&#8863;</td>
 <td>8863</td>
 <td>229F</td>
 <td></td>
 <td>squared minus</td>
</tr>
<tr valign="top">
 <td>&#8864;</td>
 <td>8864</td>
 <td>22A0</td>
 <td></td>
 <td>squared times</td>
</tr>
<tr valign="top">
 <td>&#8865;</td>
 <td>8865</td>
 <td>22A1</td>
 <td></td>
 <td>squared dot operator</td>
</tr>
<tr valign="top">
```

```
  <td>&#8866;</td>
  <td>8866</td>
  <td>22A2</td>
  <td></td>
  <td>right tack</td>
 </tr>
 <tr valign="top">
  <td>&#8867;</td>
  <td>8867</td>
  <td>22A3</td>
  <td></td>
  <td>left tack</td>
 </tr>
 <tr valign="top">
  <td>&#8868;</td>
  <td>8868</td>
  <td>22A4</td>
  <td></td>
  <td>down tack</td>
 </tr>
 <tr valign="top">
  <td>&#8869;</td>
  <td>8869</td>
  <td>22A5</td>
  <td>&amp;perp;</td>
  <td>up tack</td>
 </tr>
 <tr valign="top">
  <td>&#8870;</td>
  <td>8870</td>
  <td>22A6</td>
  <td></td>
  <td>assertion</td>
 </tr>
 <tr valign="top">
  <td>&#8871;</td>
  <td>8871</td>
  <td>22A7</td>
  <td></td>
  <td>models</td>
 </tr>
 <tr valign="top">
  <td>&#8872;</td>
  <td>8872</td>
  <td>22A8</td>
  <td></td>
  <td>true</td>
 </tr>
 <tr valign="top">
  <td>&#8873;</td>
  <td>8873</td>
  <td>22A9</td>
  <td></td>
  <td>forces</td>
```

```
</tr>
<tr valign="top">
 <td>&#8874;</td>
 <td>8874</td>
 <td>22AA</td>
 <td></td>
 <td>triple vertical bar right turnstile</td>
</tr>
<tr valign="top">
 <td>&#8875;</td>
 <td>8875</td>
 <td>22AB</td>
 <td></td>
 <td>double vertical bar double right turnstile</td>
</tr>
<tr valign="top">
 <td>&#8876;</td>
 <td>8876</td>
 <td>22AC</td>
 <td></td>
 <td>does not prove</td>
</tr>
<tr valign="top">
 <td>&#8877;</td>
 <td>8877</td>
 <td>22AD</td>
 <td></td>
 <td>not true</td>
</tr>
<tr valign="top">
 <td>&#8878;</td>
 <td>8878</td>
 <td>22AE</td>
 <td></td>
 <td>does not force</td>
</tr>
<tr valign="top">
 <td>&#8879;</td>
 <td>8879</td>
 <td>22AF</td>
 <td></td>
 <td>negated double vertical bar double right turnstile</td>
</tr>
<tr valign="top">
 <td>&#8880;</td>
 <td>8880</td>
 <td>22B0</td>
 <td></td>
 <td>precedes under relation</td>
</tr>
<tr valign="top">
 <td>&#8881;</td>
 <td>8881</td>
 <td>22B1</td>
```

```
 <td></td>
 <td>succeeds under relation</td>
</tr>
<tr valign="top">
 <td>&#8882;</td>
 <td>8882</td>
 <td>22B2</td>
 <td></td>
 <td>normal subgroup of</td>
</tr>
<tr valign="top">
 <td>&#8883;</td>
 <td>8883</td>
 <td>22B3</td>
 <td></td>
 <td>contains as normal subgroup</td>
</tr>
<tr valign="top">
 <td>&#8884;</td>
 <td>8884</td>
 <td>22B4</td>
 <td></td>
 <td>normal subgroup of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8885;</td>
 <td>8885</td>
 <td>22B5</td>
 <td></td>
 <td>contains as normal subgroup or equal to</td>
</tr>
<tr valign="top">
 <td>&#8886;</td>
 <td>8886</td>
 <td>22B6</td>
 <td></td>
 <td>original of</td>
</tr>
<tr valign="top">
 <td>&#8887;</td>
 <td>8887</td>
 <td>22B7</td>
 <td></td>
 <td>image of</td>
</tr>
<tr valign="top">
 <td>&#8888;</td>
 <td>8888</td>
 <td>22B8</td>
 <td></td>
 <td>multimap</td>
</tr>
<tr valign="top">
 <td>&#8889;</td>
```

```
  <td>8889</td>
  <td>22B9</td>
  <td></td>
  <td>hermitian conjugate matrix</td>
 </tr>
 <tr valign="top">
  <td>&#8890;</td>
  <td>8890</td>
  <td>22BA</td>
  <td></td>
  <td>intercalate</td>
 </tr>
 <tr valign="top">
  <td>&#8891;</td>
  <td>8891</td>
  <td>22BB</td>
  <td></td>
  <td>xor</td>
 </tr>
 <tr valign="top">
  <td>&#8892;</td>
  <td>8892</td>
  <td>22BC</td>
  <td></td>
  <td>nand</td>
 </tr>
 <tr valign="top">
  <td>&#8893;</td>
  <td>8893</td>
  <td>22BD</td>
  <td></td>
  <td>nor</td>
 </tr>
 <tr valign="top">
  <td>&#8894;</td>
  <td>8894</td>
  <td>22BE</td>
  <td></td>
  <td>right angle with arc</td>
 </tr>
 <tr valign="top">
  <td>&#8895;</td>
  <td>8895</td>
  <td>22BF</td>
  <td></td>
  <td>right triangle</td>
 </tr>
 <tr valign="top">
  <td>&#8896;</td>
  <td>8896</td>
  <td>22C0</td>
  <td></td>
  <td>n-ary logical and</td>
 </tr>
```

```
<tr valign="top">
 <td>&#8897;</td>
 <td>8897</td>
 <td>22C1</td>
 <td></td>
 <td>n-ary logical or</td>
</tr>
<tr valign="top">
 <td>&#8898;</td>
 <td>8898</td>
 <td>22C2</td>
 <td></td>
 <td>n-ary intersection</td>
</tr>
<tr valign="top">
 <td>&#8899;</td>
 <td>8899</td>
 <td>22C3</td>
 <td></td>
 <td>n-ary union</td>
</tr>
<tr valign="top">
 <td>&#8900;</td>
 <td>8900</td>
 <td>22C4</td>
 <td></td>
 <td>diamond operator</td>
</tr>
<tr valign="top">
 <td>&#8901;</td>
 <td>8901</td>
 <td>22C5</td>
 <td>&amp;sdot;</td>
 <td>dot operator</td>
</tr>
<tr valign="top">
 <td>&#8902;</td>
 <td>8902</td>
 <td>22C6</td>
 <td></td>
 <td>star operator</td>
</tr>
<tr valign="top">
 <td>&#8903;</td>
 <td>8903</td>
 <td>22C7</td>
 <td></td>
 <td>division times</td>
</tr>
<tr valign="top">
 <td>&#8904;</td>
 <td>8904</td>
 <td>22C8</td>
 <td></td>
```

```
 <td>bowtie</td>
</tr>
<tr valign="top">
 <td>&#8905;</td>
 <td>8905</td>
 <td>22C9</td>
 <td></td>
 <td>left normal factor semidirect product</td>
</tr>
<tr valign="top">
 <td>&#8906;</td>
 <td>8906</td>
 <td>22CA</td>
 <td></td>
 <td>right normal factor semidirect product</td>
</tr>
<tr valign="top">
 <td>&#8907;</td>
 <td>8907</td>
 <td>22CB</td>
 <td></td>
 <td>left semidirect product</td>
</tr>
<tr valign="top">
 <td>&#8908;</td>
 <td>8908</td>
 <td>22CC</td>
 <td></td>
 <td>right semidirect product</td>
</tr>
<tr valign="top">
 <td>&#8909;</td>
 <td>8909</td>
 <td>22CD</td>
 <td></td>
 <td>reversed tilde equals</td>
</tr>
<tr valign="top">
 <td>&#8910;</td>
 <td>8910</td>
 <td>22CE</td>
 <td></td>
 <td>curly logical or</td>
</tr>
<tr valign="top">
 <td>&#8911;</td>
 <td>8911</td>
 <td>22CF</td>
 <td></td>
 <td>curly logical and</td>
</tr>
<tr valign="top">
 <td>&#8912;</td>
 <td>8912</td>
```

```
 <td>22D0</td>
 <td></td>
 <td>double subset</td>
</tr>
<tr valign="top">
 <td>&#8913;</td>
 <td>8913</td>
 <td>22D1</td>
 <td></td>
 <td>double superset</td>
</tr>
<tr valign="top">
 <td>&#8914;</td>
 <td>8914</td>
 <td>22D2</td>
 <td></td>
 <td>double intersection</td>
</tr>
<tr valign="top">
 <td>&#8915;</td>
 <td>8915</td>
 <td>22D3</td>
 <td></td>
 <td>double union</td>
</tr>
<tr valign="top">
 <td>&#8916;</td>
 <td>8916</td>
 <td>22D4</td>
 <td></td>
 <td>pitchfork</td>
</tr>
<tr valign="top">
 <td>&#8917;</td>
 <td>8917</td>
 <td>22D5</td>
 <td></td>
 <td>equal and parallel to</td>
</tr>
<tr valign="top">
 <td>&#8918;</td>
 <td>8918</td>
 <td>22D6</td>
 <td></td>
 <td>less-than with dot</td>
</tr>
<tr valign="top">
 <td>&#8919;</td>
 <td>8919</td>
 <td>22D7</td>
 <td></td>
 <td>greater-than with dot</td>
</tr>
<tr valign="top">
```

```
 <td>&#8920;</td>
 <td>8920</td>
 <td>22D8</td>
 <td></td>
 <td>very much less-than</td>
</tr>
<tr valign="top">
 <td>&#8921;</td>
 <td>8921</td>
 <td>22D9</td>
 <td></td>
 <td>very much greater-than</td>
</tr>
<tr valign="top">
 <td>&#8922;</td>
 <td>8922</td>
 <td>22DA</td>
 <td></td>
 <td>less-than equal to or greater-than</td>
</tr>
<tr valign="top">
 <td>&#8923;</td>
 <td>8923</td>
 <td>22DB</td>
 <td></td>
 <td>greater-than equal to or less-than</td>
</tr>
<tr valign="top">
 <td>&#8924;</td>
 <td>8924</td>
 <td>22DC</td>
 <td></td>
 <td>equal to or less-than</td>
</tr>
<tr valign="top">
 <td>&#8925;</td>
 <td>8925</td>
 <td>22DD</td>
 <td></td>
 <td>equal to or greater-than</td>
</tr>
<tr valign="top">
 <td>&#8926;</td>
 <td>8926</td>
 <td>22DE</td>
 <td></td>
 <td>equal to or precedes</td>
</tr>
<tr valign="top">
 <td>&#8927;</td>
 <td>8927</td>
 <td>22DF</td>
 <td></td>
 <td>equal to or succeeds</td>
```

```
</tr>
<tr valign="top">
 <td>&#8928;</td>
 <td>8928</td>
 <td>22E0</td>
 <td></td>
 <td>does not precede or equal</td>
</tr>
<tr valign="top">
 <td>&#8929;</td>
 <td>8929</td>
 <td>22E1</td>
 <td></td>
 <td>does not succeed or equal</td>
</tr>
<tr valign="top">
 <td>&#8930;</td>
 <td>8930</td>
 <td>22E2</td>
 <td></td>
 <td>not square image of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8931;</td>
 <td>8931</td>
 <td>22E3</td>
 <td></td>
 <td>not square original of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8932;</td>
 <td>8932</td>
 <td>22E4</td>
 <td></td>
 <td>square image of or not equal to</td>
</tr>
<tr valign="top">
 <td>&#8933;</td>
 <td>8933</td>
 <td>22E5</td>
 <td></td>
 <td>square original of or not equal to</td>
</tr>
<tr valign="top">
 <td>&#8934;</td>
 <td>8934</td>
 <td>22E6</td>
 <td></td>
 <td>less-than but not equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8935;</td>
 <td>8935</td>
 <td>22E7</td>
```

```
 <td></td>
 <td>greater-than but not equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8936;</td>
 <td>8936</td>
 <td>22E8</td>
 <td></td>
 <td>precedes but not equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8937;</td>
 <td>8937</td>
 <td>22E9</td>
 <td></td>
 <td>succeeds but not equivalent to</td>
</tr>
<tr valign="top">
 <td>&#8938;</td>
 <td>8938</td>
 <td>22EA</td>
 <td></td>
 <td>not normal subgroup of</td>
</tr>
<tr valign="top">
 <td>&#8939;</td>
 <td>8939</td>
 <td>22EB</td>
 <td></td>
 <td>does not contain as normal subgroup</td>
</tr>
<tr valign="top">
 <td>&#8940;</td>
 <td>8940</td>
 <td>22EC</td>
 <td></td>
 <td>not normal subgroup of or equal to</td>
</tr>
<tr valign="top">
 <td>&#8941;</td>
 <td>8941</td>
 <td>22ED</td>
 <td></td>
 <td>does not contain as normal subgroup or equal</td>
</tr>
<tr valign="top">
 <td>&#8942;</td>
 <td>8942</td>
 <td>22EE</td>
 <td></td>
 <td>vertical ellipsis</td>
</tr>
<tr valign="top">
 <td>&#8943;</td>
```

```
 <td>8943</td>
 <td>22EF</td>
 <td></td>
 <td>midline horizontal ellipsis</td>
</tr>
<tr valign="top">
 <td>&#8944;</td>
 <td>8944</td>
 <td>22F0</td>
 <td></td>
 <td>up right diagonal ellipsis</td>
</tr>
<tr valign="top">
 <td>&#8945;</td>
 <td>8945</td>
 <td>22F1</td>
 <td></td>
 <td>down right diagonal ellipsis</td>
</tr>
<tr valign="top">
 <td>&#8946;</td>
 <td>8946</td>
 <td>22F2</td>
 <td></td>
 <td>element of with long horizontal stroke</td>
</tr>
<tr valign="top">
 <td>&#8947;</td>
 <td>8947</td>
 <td>22F3</td>
 <td></td>
 <td>element of with vertical bar at end of horizontal stroke</td>
</tr>
<tr valign="top">
 <td>&#8948;</td>
 <td>8948</td>
 <td>22F4</td>
 <td></td>
 <td>small element of with vertical bar at end of horizontal stroke</td>
</tr>
<tr valign="top">
 <td>&#8949;</td>
 <td>8949</td>
 <td>22F5</td>
 <td></td>
 <td>element of with dot above</td>
</tr>
<tr valign="top">
 <td>&#8950;</td>
 <td>8950</td>
 <td>22F6</td>
 <td></td>
 <td>element of with overbar</td>
</tr>
```

```
<tr valign="top">
 <td>&#8951;</td>
 <td>8951</td>
 <td>22F7</td>
 <td></td>
 <td>small element of with overbar</td>
</tr>
<tr valign="top">
 <td>&#8952;</td>
 <td>8952</td>
 <td>22F8</td>
 <td></td>
 <td>element of with underbar</td>
</tr>
<tr valign="top">
 <td>&#8953;</td>
 <td>8953</td>
 <td>22F9</td>
 <td></td>
 <td>element of with two horizontal strokes</td>
</tr>
<tr valign="top">
 <td>&#8954;</td>
 <td>8954</td>
 <td>22FA</td>
 <td></td>
 <td>contains with long horizontal stroke</td>
</tr>
<tr valign="top">
 <td>&#8955;</td>
 <td>8955</td>
 <td>22FB</td>
 <td></td>
 <td>contains with vertical bar at end of horizontal stroke</td>
</tr>
<tr valign="top">
 <td>&#8956;</td>
 <td>8956</td>
 <td>22FC</td>
 <td></td>
 <td>small contains with vertical bar at end of horizontal stroke</td>
</tr>
<tr valign="top">
 <td>&#8957;</td>
 <td>8957</td>
 <td>22FD</td>
 <td></td>
 <td>contains with overbar</td>
</tr>
<tr valign="top">
 <td>&#8958;</td>
 <td>8958</td>
 <td>22FE</td>
 <td></td>
```

```
  <td>small contains with overbar</td>
 </tr>
 <tr valign="top">
  <td>&#8959;</td>
  <td>8959</td>
  <td>22FF</td>
  <td></td>
  <td>z notation bag membership</td>
 </tr>
</table>
\getchunk{page foot}
```

———-

algebrapage.xhtml

— algebrapage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
Axiom provides various facilities for treating topics in
abstract algebra
  <table>
   <tr>
    <td>
     <a href="algnumbertheory.xhtml">Number Theory</a>
    </td>
    <td>
     Topics in algebraic number theory
    </td>
   </tr>
   <tr>
    <td>
     <a href="alggrouptheory.xhtml">Group Theory</a>
    </td>
    <td>
     Permuation groups; representation theory
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

———-

alggrouptheory.xhtml

— alggrouptheory.xhtml —

```
\getchunk{standard head}
 </head>
```

```
 <body>
\getchunk{page head}
  <div align="center">Group Theory</div>
  <hr/>
Axiom can work with individual permutations, permutation groups and
do representation theory.
<ul>
 <li> <a href="alggrouptheorygroup.xhtml">Info on Group Theory</a></li>
 <li> <a href="alggrouptheoryreptheory.xhtml">
      Info on Representation Theory</a></li>
 <li> <a href="alggrouptheoryrepa6.xhtml">Representations of A6</a><br/>
      The irreducible representations of the alternating group A6 over
      fields of characteristic 2.</li>
</ul>
\getchunk{page foot}
```

alggrouptheorygroup.xhtml

— alggrouptheorygroup.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Group Theory</div>
  <hr/>
A group is a set G together with an associative operation * satisfying
the axioms of existence of a unit element and an inverse of every
element of the group. The Axiom category <a href="db.xhtml?Group">Group</a>
represents this setting. Many data structures in Axiom are groups and
therefore there is a large variety of examples as fields and polynomials,
although the main interest there is not a group structure.
<br/>
<br/>
To work with and in groups in a concrete manner some way of representing
groups has to be chosen. A group can be given as a list of generators and
as a set of relations. If there are no relations, then we have a free group,
realized in the domain <a href="db.xhtml?FreeMonoid">FreeMonoid</a> which
won't be discussed here. We consider permutation groups, where a group is
realized as a subgroup of the symmetric group of a set, i.e., the group of
all bijections of a set, the operation being the composition of maps. Indeed,
every groupo can be realized this way, although this may not be practical.
<br/>
<br/>
Furthermore group elements can be given as invertible matrices. The group
operation is reflected by matrix multiplication. More precise in
representation theory group homomorphisms from a group to general linear
groups are constructed. Some algorithms are implemented in Axiom.
\getchunk{page foot}
```

alggrouptheoryrepa6.xhtml

— alggrouptheoryrepa6.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Representations of A6</div>
  <hr/>
In what follows you'll see how to use Axiom to get all the irreducible
representations of the alternating group A6 over the field with two
elements (GF 2). First, we generate A6 by a three-cycle: x=(1,2,3) and a
5-cycle: y=(2,3,4,5,6). Next we have Axiom calculate the permutation
representation over the integers and over GF 2:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="genA6:LIST PERM INT:=[cycle [1,2,3],cycle [2,3,4,5,6]]" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="pRA6:=permutationRepresentation(genA6,6)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Now we apply Parker's 'Meat-Axe' and split it:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="sp0:=meatAxe(pRA6::(LIST MATRIX PF 2))" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
We have found the trivial module as a quotient module and a
5-dimensional sub-module. Try to split again:
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="sp1:=meatAxe sp0.1" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
and we find a 4-dimensional sub-module and the trivial one again. Now we
```

```
can test if this representation is absolutely irreducible.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’]);"
    value="isAbsolutelyIrreducible? sp1.2" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
and we see that this 4-dimensional representation is absolutely irreducible.
So, we have found a second irreducible representation. Now, we construct a
representation by reducing an irreducible one of the symmetric group S_6
over the integers mod 2. We take the one labelled by the partition
[2,2,1,1] and restrict it to A6:
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree([’p1’,’p6’]);"
    value="d2211:=irreducibleRepresentation([2,2,1,1],genA6)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree([’p1’,’p6’,’p7’]);"
    value="d2211m2:=d2211::(LIST MATRIX PF 2); sp2:=meatAxe d2211m2" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
This gave both a five and a four dimensional representation. Now we take the
4-dimensional one and we shall see that it is absolutely irreducible:
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree([’p1’,’p6’,’p7’,’p8’]);"
    value="isAbsolutelyIrreducible? sp2.1" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
The two 4-dimensional representations are not equivalent:
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree([’p1’,’p6’,’p7’,’p9’]);"
    value="areEquivalent?(sp1.2,sp2.1)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
So we have found a third irreducible representation. Now we construct a new
representation using the tensor product and try to split it:
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree([’p1’,’p6’,’p7’,’p10’]);"
    value="dA6d16:=tensorProduct(sp1.2,sp2.1); meatAxe dA6d16" />
```

```
  <div id="ansp10"><div></div></div>
 </li>
</ul>
The representation is irreducible, but it may be not absolutely irreducible.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p11']);"
    value="isAbsolutelyIrreducible? dA6d16" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
So let's try the same procedure over the field with 4 elements:
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p12']);"
    value="sp3:=meatAxe(dA6d16::(LIST MATRIX FF(2,2)))" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
Now we find two 8-dimensional representations, dA6d8a and dA6d8b. Both
are absolutely irreducible.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p12','p13']);"
    value="isAbsolutelyIrreducible? sp3.1" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p12','p14']);"
    value="isAbsolutelyIrreducible? sp3.2" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
and they are not equivalent.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p12','p15']);"
    value="areEquivalent?(sp3.1,sp3.2)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
So we have found five absolutely irreducible representations of A6 in
characteristic 2. General theory now tells us that there are no more
irreducible ones. Here, for future reference are all the absolutely
irreducible 2-module representations of A6.
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p1','p2','p3','p16']);"
```

```
    value="sp0.2" />
  <div id="ansp16"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p17']);"
    value="sp1.2" />
  <div id="ansp17"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="handleFree(['p1','p6','p7','p18']);"
    value="sp2.1" />
  <div id="ansp18"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p12','p19']);"
    value="sp3.1" />
  <div id="ansp19"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p12','p20']);"
    value="sp3.2" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
And here again is the irreducible, but not absolutely irreducible
representations of A6 over GF 2.
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="handleFree(['p1','p6','p7','p10','p21']);"
    value="dA6d16" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

alggrouptheoryreptheory.xhtml

— alggrouptheoryreptheory.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
Representation theory for finite groups studies finite groups by embedding
them in a general linear group over a field or an integral domain. Hence,
we are representing each element of the group by an invertible matrix.
```

```
Two marix representations of a given group are equivalent, if, by changing
the basis of the underlying space, you can go from one to the other. When
you change bases, you transform the matrices that are the images of elements
by conjugating them by an invertible matrix.
<br/>
<br/>
If we can find a subspace which is fixed under the image of the group, then
there exists a 'base change' after which all the representing matrices are
in upper triangular block form. The block matrices on the main diagonal give
a new representation of the group of lower degree. Such a representation is
said to be 'reducible'.
\getchunk{page foot}
```

———-

algnumbertheory.xhtml

— algnumbertheory.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Number Theory</div>
  <hr/>
Here are some sample computations using Axiom's algebraic number facilities.
<ul>
 <li> <a href="algnumbertheorygalois.xhtml">Galois Groups</a><br/>
      Computation of Galois groups using factorizations over number fields
 </li>
 <li> <a href="numfunctions.xhtml">Number Theory Functions</a><br/>
      Some functions of interest to number theorists
 </li>
</ul>
\getchunk{page foot}
```

———-

algnumbertheorygalois.xhtml

— algnumbertheorygalois.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Computation of Galois Groups</div>
  <hr/>
```

```
As a sample use of Axiom's algebraic number facilities, we compute the
Galois group of the polynomial p(x)=x**5-5*x+12
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="p:=x**5-5*x+12" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
We would like to construct a polynomial f(x) such that the splitting field
of p(x) is generated by one root of f(x). First we construct a polynomial
r=r(x) such that one root of r(x) generates the filed generatedy by two
roots of the polynomial p(x). As it will turn out, the field generate by
two roots of p(x) is, in fact, the splitting field of p(x).

From the proof of the primitive element theorem we know that if a and b
are algebraic numbers, then the field Q(a,b) is equal to Q(a+k*b) for an
appropriately chosen integer k. In our case, we construct the minimal
polynomial of a[i]-a[j], where a[i] and a[j] are two roots of p(x). We
construct this polynomial using <a href="dbopresultant.xhtml">resultant</a>.
The main result we need is that if f(x) is a polynomial with roots a[1]...a[m]
and g(x) is a polynomial with roots b[1]...b[n], then the polynomial
h(x)=resultant(f(y),g(x-y),y) is a polynomial of degree m*n with roots
a[i]+b[j], 1 &lt;= i &lt;= m, 1 &lt;= j &lt;= n.

For f(x) we use the polynomial p(x). For g(x) we use the polynomial -p(-x).
Thus, the polynomial we first construct is resultant(p(y),-p(y-x),y).
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="q:=resultant(eval(p,x,y),-eval(p,x,y-x),y)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The roots of q(x) are a[i]-a[j], 1 &lt;= i,j &lt;= 5. Of course, there are
five pairs (i,j) with i=j, so 0 is a 5-fold root of q(x). Let's get rid of
this factor.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="q1:=exquo(q,x^5)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Factor the polynomial q1.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="factoredQ:=factor q1" />
  <div id="ansp4"><div></div></div>
```

```
 </li>
</ul>
We see that q1 has two irreducible factors, each of degree 10. (The fact
that the polynomial q1 has two factors of degree 10 is enough to show that
the Galois group of p(x) is the diheral group of order 10. (ref:
McKay, Soicher, Computing Galois Groups over the Rationals, Journal of
Number Theory 20, 273-281 (1983). We do not assume the results of this
paper and we continue with the computation.) Note that the type of
factoredQ is
<a href="db.xhtml?Factored(Polnomial(Integer))">
Factored Polynomial Integer</a>.
This is a special data type for
recording factorizations of polynomials with integer coefficients (see
<a href="factored.xhtml">Factored</a>). We can access the individual
factors using the operation <a href="dbopnthfactor.xhtml">nthFactor</a>.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5']);"
    value="r:=nthFactor(factoredQ,1)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Consider the polynomial r=r(x). This is the minimal polynomial of the
difference of two roots of p(x). Thus, the splitting field of p(x)
contains a subfield of degree 10. We show that this subfield is the
splitting field of p(x) by showing that p(x) factors completely over this
field. First we create a symbolic root of the polynomial r(x). (We
replaced x by b in the polynomial r so that our symbolic root would
be printed as b.)
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6']);"
    value="beta:AN:=rootOf(eval(r,x,b))" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
We next tell Axiom to view p(x) as a univariate polynomial in x with
algebraic number coefficients. This is accomplished with this type
declaration:
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7']);"
    value="p:=p::UP(x,INT)::UP(x,AN)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Factor p(x) over the field Q(beta). (This computation will take some time).
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8']);"
```

```
    value="algFactors:=factor(p,[beta])" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
When factoring over number fields, it is important to specify the field
over which the polynomial is to be factored, as polynomials have different
factorizations over different fields. When you use the operation
<a href="dbopfactor.xhtml">factor</a>, the field over which the polynomial
is factored is the field generated by
<ol>
 <li> the algebraic numbers that appear in the coefficients of the polynomial
 </li>
 <li> the algebraic numbers that appear in a list passed as an optional
      second argument of the operation
 </li>
</ol>
In our case, the coefficients of p are all rational numbers and only beta
appears in the list, so the field is simply Q(beta). It was necessary to
give the list [beta] as a second argument of the operations because
otherwise the polynomial would have to be factored over the field generated
by its coefficients, namely the rational numbers.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree([’p1’,’p9’]);"
    value="factor(p)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
We have shown that the splitting field of p(x) has degree 10. Since the
symmetric group of degree 5 has only one transitive subgroup of order 10,
we know that the Galois group of p(x) must be this group, the dihedral
group of order 10. Rather than stop here, we explicitly compute the action
of the Galois group on the roots of p(x).

First we assign the roots of p(x) as the values of five variables. We
can obtain an individual root by negating the constant coefficient of one
of the factors of p(x).
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p10’]);"
    value="factor1:=nthFactor(algFactors,1)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
   onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p10’,’p11’]);"
   value="root1:=-coefficient(factor1,0)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
```

```
We can obtain a list of all of the roots in this way.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
   onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12']);"
    value="roots:=[-coefficient(nthFactor(algFactors,i),0) for i in 1..5]" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
The expression -coefficient(nthFactor(algFactors,i),0) is the ith root of p(x)
and the elements of roots are the ith roots of p(x) as i ranges from 1 to 5.
Assign the roots as the values of the variables a1..a5.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
   onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13']);"
    value="(a1,a2,a3,a4,a5):=(roots.1,roots.2,roots.3,roots.4,roots.5)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
Next we express the roots of r(x) as polynomials in beta. We could obtain
these roots by calling the operation
<a href="dbopfactor.xhtml">factor</a>. factor(r,[beta]) factors r(x) over
Q(beta). However, this is a length computation and we can obtain the roots
of r(x) as differences of the roots a1,...,a5 of p(x). Only ten of these
differences are roots of r(x) and the other ten are roots of the other
irreducible factor of q1. We can determine if a given value is a root
of r(x) by evaluating r(x) at that particular value. (Of course, the order
in which factors are returned by the operation
<a href="dbopfactor.xhtml">factor</a> is unimportant and may change with
different implementations of the operation. Therefore, we cannot predict
in advance which differences are roots of r(x) and which are not.) Let's
look at four examples (two are roots of r(x) and two are not).
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
   onclick=
    "handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p14']);"
    value="eval(r,x,a1-a2)" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
   onclick=
    "handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p15']);"
    value="eval(r,x,a1-a3)" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
   onclick=
    "handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p16']);"
    value="eval(r,x,a1-a4)" />
  <div id="ansp16"><div></div></div>
```

```
 </li>
 <li>
  <input type="submit" id="p17" class="subbut"
   onclick=
    "handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p12’,’p13’,’p17’]);"
    value="eval(r,x,a1-a5)" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
Take one of the differences that was a root of r(x) and assign it to the
variable bb. For example, if eval(r,x,a1-a4) returned 0, you would enter this.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
   onclick=
    "handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p12’,’p13’,’p18’]);"
    value="bb:=a1-a4" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
Of course, if the difference is equal to the root beta, you should choose
another root of r(x).

Automorphisms of the splitting field are given by mapping a generator of
the field, namely beta, to other roots of its minimal polynomial. Let’s
see what happens when beta is mapped to bb. We compute the images of the
roots a1,...,a5 under this automorphism.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
   onclick=
    "handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p12’,’p13’,’p19’]);"
    value="aa1:=subst(a1,beta=bb)" />
  <div id="ansp19"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p20" class="subbut"
   onclick=
    "handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p12’,’p13’,’p20’]);"
    value="aa2:=subst(a2,beta=bb)" />
  <div id="ansp20"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p21" class="subbut"
   onclick=
    "handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p12’,’p13’,’p21’]);"
    value="aa3:=subst(a3,beta=bb)" />
  <div id="ansp21"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p22" class="subbut"
   onclick=
    "handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p7’,’p8’,’p12’,’p13’,’p22’]);"
    value="aa4:=subst(a4,beta=bb)" />
```

```
  <div id="ansp22"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p23" class="subbut"
   onclick=
    "handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p23']);"
    value="aa5:=subst(a5,beta=bb)" />
  <div id="ansp23"><div></div></div>
 </li>
</ul>
Of course, the values aa1,...,aa5 are simply a permutation of the values
a1,...,a5. Let's find the value of aa1 (execute as many of the following
five commands as necessary).
<ul>
 <li>
  <input type="submit" id="p24" class="subbut"
   onclick=
"handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p19','p24']);"
    value="(aa1=a1)::Boolean" />
  <div id="ansp24"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p25" class="subbut"
   onclick=
"handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p20','p25']);"
    value="(aa2=a2)::Boolean" />
  <div id="ansp25"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p26" class="subbut"
   onclick=
"handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p21','p26']);"
    value="(aa3=a3)::Boolean" />
  <div id="ansp26"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p27" class="subbut"
   onclick=
"handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p22','p27']);"
    value="(aa4=a4)::Boolean" />
  <div id="ansp27"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p28" class="subbut"
   onclick=
"handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p12','p13','p23','p28']);"
    value="(aa5=a5)::Boolean" />
  <div id="ansp28"><div></div></div>
 </li>
</ul>
Proceeding in this fashion, you can find the values of aa2..aa5.
You have represented the automorphism beta -> bb as a permutation of the
roots a1,...,a5. If you wish, you can repeat this computation for all the
roots of r(x) and represent the Galois group of p(x) as a subgroup of the
```

```
symmetric group on five letters.

Here are two other problems that you may attack in a similar fashion:
<ol>
 <li> Show that the Galois group of p(x)=x**4+2*x**3-2*x**2-2*x+1 is the
      dihedral group of order eight. (The splitting field of this polynomial
      is the Hilbert class field of the quadratic field Q(sqrt(145)).)
 </li>
 <li> Show that the Galois group of p(x)=x**6+108 has order 6 and is
      isomorphic to the symmetric group on three letters. (The splitting
      field of this polynomial is the spliting field of x**3-2.)
 </li>
</ol>
\getchunk{page foot}
```

basiccommand.xhtml

— basiccommand.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
     <a href="calculus.xhtml">
      <b>Calculus</b>
     </a>
    </td>
    <td>Compute integrals, derivatives, or limits</td>
   </tr>
   <tr>
    <td>
     <a href="bcmatrix.xhtml">
      <b>Matrix</b>
     </a>
    </td>
    <td>Create a matrix</td>
   </tr>
   <tr>
    <td><a href="bcexpand.xhtml"><b>Operations</b></a></td>
    <td>Expand, factor, simplify, substitute, etc.</td>
   </tr>
   <tr>
    <td>
     <a href="draw.xhtml">
      <b>Draw</b>
     </a>
    </td>
    <td>Create 2D or 3D plots.</td>
```

```
   </tr>
   <tr>
    <td>
     <a href="series.xhtml">
      <b>Series</b>
     </a>
    </td>
    <td>Create a power series</td>
   </tr>
   <tr>
    <td>
     <a href="solve.xhtml">
      <b>Solve</b>
     </a>
    </td>
    <td>Solve an equation or system of equations</td>
   </tr>
  </table>
\getchunk{page foot}
```

basiclimit.xhtml

— **basiclimit.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
   What kind of limit do you want to compute?:<br/>
   <a href="reallimit.xhtml">
    <b>A real limit</b>
   </a><br/>
   The limit as the variable approaches a real value along the real axis
   <br/><br/>
   <a href="complexlimit.xhtml">
    <b>A complex limit</b>
   </a><br/>
   The limit as the variable approaches a complex value along any path in
   the complex plane.
\getchunk{page foot}
```

bcexpand.xhtml

— **bcexpand.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
```

```
\getchunk{page head}
Simplification
<ul>
<li>Simplify Expressions</li>
<li>Simplify Radicals</li>
<li>Factor Expressions</li>
<li>Factor Complex</li>
<li>Expand Expressions</li>
<li>Expand Logarithms</li>
<li>Contract Logarithms</li>
<li>Simpify Trigonometrics</li>
<li>Reduce Trigonometrics</li>
<li>Expand Trigonometrics</li>
<li>Canonical Trigonometrics</li>
<li>Complex to rectangular</li>
<li>Complex to polar</li>
<li>Complex to exponentials</li>
<li>Exponentials to complex</li>
</ul>
Calculus
<ul>
<li>Integrate</li>
<li>Risch Integrate</li>
<li>Change Variable</li>
<li>Differentiate</li>
<li>Find Limit</li>
<li>Get Series</li>
<li>Pade Approximation</li>
<li>Calculate Sum</li>
<li>Calculate Product</li>
<li>Laplace Transform</li>
<li>Inverse Laplace Transform</li>
<li>Greatest Common Divisor</li>
<li>Least Common Multiple</li>
<li>Divide Polynomials</li>
<li>Partial Fractions</li>
<li>Continued Fractions</li>
</ul>
Algebra
<ul>
<li>Generate Matrix</li>
<li>Enter Matrix</li>
<li>Invert Matrix</li>
<li>Characteristic Polynomial</li>
<li>Determinant</li>
<li>Eigenvalues</li>
<li>Eigenvectors</li>
<li>Adjoint Matrix</li>
<li>Transpose Matrix</li>
</ul>
Equations
<ul>
<li>Solve</li>
<li>Solve Numerically</li>
```

```
<li>Roots of Polynomials</li>
<li>Real Roots of Polynomials</li>
<li>Solve Linear Systems</li>
<li>Solve Algebraic System</li>
<li>Eliminate Variable</li>
</ul>
Ordinary Differential Equations
<ul>
<li>Solve ODE</li>
<li>Solve Initial Value Problem</li>
<li>Solve Boundary Value Problem</li>
<li>Solve ODE with Laplace</li>
</ul>
Data Structures
<ul>
<li>Record</li>
<li>List</li>
<li>Set</li>
</ul>
\getchunk{page foot}
```

bcmatrix.xhtml

— bcmatrix.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
<![CDATA[
   function byformula() {
      // find out how many rows and columns, must be positive and nonzero
    var rcnt = parseInt(document.getElementById('rowcnt').value);
     if (rcnt <= 0) {
      alert("Rows must be positive and non-zero -- defaulting to 1");
      rcnt = 1;
      document.getElementById('rowcnt').value=1;
      return(false);
     }
    var ccnt = parseInt(document.getElementById('colcnt').value);
     if (ccnt <= 0) {
      alert("Columns must be positive and non-zero -- defaulting to 1");
      ccnt = 1;
      document.getElementById('colcnt').value=1;
      return(false);
     }
      // remove the question and the buttons
    var quest = document.getElementById('question');
    var clicks = document.getElementById('clicks');
    quest.removeChild(clicks);
    var tbl = document.getElementById('form2');
    var tblsize = tbl.rows.length;
      // make the row variable question
```

```
  // row variable left cell
var row = tbl.insertRow(tblsize);
var cell = row.insertCell(0);
var tnode = document.createTextNode("Enter the row variable");
cell.appendChild(tnode);
  // row variable right cell
cell = row.insertCell(1);
tnode = document.createElement('input');
tnode.type = 'text';
tnode.name = 'rowvar';
tnode.id = 'rowvar';
tnode.size=10;
tnode.value='i';
tnode.tabindex=21;
cell.appendChild(tnode);
  // make the column variable question
  // column variable left cell
tblsize = tblsize + 1;
row = tbl.insertRow(tblsize);
cell = row.insertCell(0);
tnode = document.createTextNode("Enter the column variable");
cell.appendChild(tnode);
  // column variable right cell
cell = row.insertCell(1);
tnode = document.createElement('input');
tnode.type = 'text';
tnode.name = 'colvar';
tnode.id = 'colvar';
tnode.size=10;
tnode.tabindex=22;
tnode.value='j';
cell.appendChild(tnode);
  // make the formula question
  // column variable left cell
tblsize = tblsize + 1;
row = tbl.insertRow(tblsize);
cell = row.insertCell(0);
tnode = document.createTextNode("Enter the formulas for the elements");
cell.appendChild(tnode);
  // formula input field
tblsize = tblsize + 1;
row = tbl.insertRow(tblsize);
cell = row.insertCell(0);
tnode = document.createElement('input');
tnode.type = 'text';
tnode.name = 'formula1';
tnode.id = 'formula1';
tnode.size=50;
tnode.value = '1/(x-i-j-1)';
tnode.tabindex=23;
cell.appendChild(tnode);
  // insert the continue button
tblsize = tblsize + 1;
row = tbl.insertRow(tblsize);
```

```
 cell = row.insertCell(0);
 tnode = document.createElement('input');
 tnode.type = 'button';
 tnode.id = 'contbutton';
 tnode.value = 'Continue';
 tnode.setAttribute("onclick","makeRequest('formula');");
 tnode.tabindex=24;
 cell.appendChild(tnode);
 return(false);
}
function byelement() {
   // find out how many rows and columns, must be positive and nonzero
 var rcnt = parseInt(document.getElementById('rowcnt').value);
  if (rcnt <= 0) {
   alert("Rows must be positive and non-zero -- defaulting to 1");
   rcnt = 1;
   document.getElementById('rowcnt').value=1;
   return(false);
  }
 var ccnt = parseInt(document.getElementById('colcnt').value);
  if (ccnt <= 0) {
   alert("Columns must be positive and non-zero -- defaulting to 1");
   ccnt = 1;
   document.getElementById('colcnt').value=1;
   return(false);
  }
   // remove the question and the buttons
 var quest = document.getElementById('question');
 var clicks = document.getElementById('clicks');
 quest.removeChild(clicks);
   // write "Elements"
 var tbl = document.getElementById('form2');
 var tblsize = tbl.rows.length;
 var row = tbl.insertRow(tblsize);
 var thecell = row.insertCell(0);
 var tnode = document.createTextNode("Elements");
 thecell.appendChild(tnode);
   // create input boxes for the matrix values
 tblsize = tblsize + 1;
 for (var i = 0 ; i < rcnt ; i++) {
  row = tbl.insertRow(tblsize);
  for (var j = 0 ; j < ccnt ; j++) {
   thecell = row.insertCell(j);
   tnode = document.createElement('input');
   tnode.type = 'text';
   tnode.name = 'a'+i+'c'+j;
   tnode.id = 'a'+i+'c'+j;
   tnode.size=10;
   tnode.tabindex=20+(i*10)+j;
   thecell.appendChild(tnode);
  }
  tblsize = tblsize + 1;
 }
   // insert the continue button
```

```
    row = tbl.insertRow(tblsize);
    thecell = row.insertCell(0);
    tnode = document.createElement('input');
    tnode.type = 'button';
    tnode.id = 'contbutton';
    tnode.value = 'Continue';
    tnode.setAttribute("onclick","makeRequest('element');");
    thecell.appendChild(tnode);
    return(false);
   }
   function commandline(arg) {
    if (arg == 'element') {
     var rcnt = parseInt(document.getElementById('rowcnt').value);
     var ccnt = parseInt(document.getElementById('colcnt').value);
     var cmdhead = 'matrix([';
     var cmdtail = '])';
      for (var i = 0 ; i < rcnt ; i++) {
      var listbody = '[';
      for (var j = 0 ; j < ccnt ; j++) {
       var aij = document.getElementById('a'+i+'c'+j).value;
       listbody = listbody+aij;
       if (j != (ccnt - 1)) {
        listbody = listbody+',';
       }
      }
      listbody = listbody+']';
      if (i != (rcnt - 1)) {
       listbody = listbody+',';
      }
      cmdhead = cmdhead+listbody;
     }
     cmd = cmdhead+cmdtail;
     return(cmd);
    } else {
     var rcnt = parseInt(document.getElementById('rowcnt').value);
     var ccnt = parseInt(document.getElementById('colcnt').value);
     var cmdhead = 'matrix([';
     var cmdtail = '])';
     var formula = document.getElementById('formula1').value;
     var rowv = document.getElementById('rowvar').value;
     var colv = document.getElementById('colvar').value;
     var cmd = cmdhead+formula+' for '+colv+' in 1..'+ccnt+']'+
                              ' for '+rowv+' in 1..'+rcnt+cmdtail;
     return(cmd);
    }
   }
]]>
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
Enter the size of the matrix:
```

```
<table id="form2">
 <tr>
  <td size="10">Rows</td>
  <td><input type="text" id="rowcnt" tabindex="10" size="10" value="2"/></td>
 </tr>
 <tr>
  <td>Columns</td>
  <td><input type="text" id="colcnt" tabindex="20" size="10" value="3"/></td>
 </tr>
</table>
<div id="question">
 <div id="clicks">
  How would you like to enter the matrix elements?
  <center>
   <input type="button" value="By Formula" onclick="byformula();"/>
   <input type="button" value="By Element" onclick="byelement();"/>
  </center>
 </div>
</div>
\getchunk{answer field}
\getchunk{page foot}
```

calculus.xhtml

— calculus.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
     <a href="differentiate.xhtml">
      <b>Differentiate</b>
     </a>
    </td>
   </tr>
   <tr>
    <td><a href="indefiniteintegral.xhtml">
      <b>Do an Indefinite Integral</b></a></td>
   </tr>
   <tr>
    <td><a href="definiteintegral.xhtml">
     <b>Do a Definite Integral</b></a></td>
   </tr>
   <tr>
    <td><a href="basiclimit.xhtml"><b>Find a limit</b></a></td>
   </tr>
   <tr>
    <td><a href="summation.xhtml">
```

```
      <b>Do a summation</b>
     </a>
    </td>
   </tr>
   <tr>
    <td><a href="(|bcProduct|).xhtml"><b>Compute a product</b></a></td>
   </tr>
  </table>
\getchunk{page foot}
```

———

calculuspage.xhtml

— calculuspage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
     <a href="callimits.xhtml">Limits</a>
    </td>
    <td>
     Compute limits of functional expressions
    </td>
   </tr>
   <tr>
    <td>
     <a href="calderivatives.xhtml">Derivatives</a>
    </td>
    <td>
     Compute derivatives and partial derivatives
    </td>
   </tr>
   <tr>
    <td>
     <a href="calintegrals.xhtml">Integrals</a>
    </td>
    <td>
     Introduction to Axiom's symbolic integration
    </td>
   </tr>
   <tr>
    <td>
     <a href="calmoreintegrals.xhtml">More Integrals</a>
    </td>
    <td>
     More information about symbolic integration
    </td>
   </tr>
```

```
   <tr>
    <td>
     <a href="callaplace.xhtml">Laplace</a>
    </td>
    <td>
     Computing Laplace transforms
    </td>
   </tr>
   <tr>
    <td>
     <a href="calseries.xhtml">Series</a>
    </td>
    <td>
     Compute series expansions of expressions
    </td>
   </tr>
   <tr>
    <td>
     <a href="equdifferential.xhtml">Differential Equations</a>
    </td>
    <td>
     Solve differential equations
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

calderivatives.xhtml

— calderivatives.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Derivatives</div>
  <hr/>
Use the Axiom function <a href="dbopd.xhtml">D</a> to differentiate an
expression.

To find the derivative of an expression f with respect to a variable x,
enter D(f,x).
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="f:=exp exp x" />
```

```
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree([’p1’,’p2’]);"
    value="D(f,x)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
An optional third argument n in <a href="dbopd.xhtml">D</a> asks Axiom for
the nth derivative of f. This finds the fourth derivative of f with
respect to x.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree([’p1’,’p3’]);"
    value="D(f,x,4)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
You can also compute partial derivatives by specifying the order of
differentiation.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest(’p4’);"
    value="g:=sin(x^2+y)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree([’p4’,’p5’]);"
    value="D(g,y)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree([’p4’,’p6’]);"
    value="D(g,[y,y,x,x])" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Axiom can manipulate the derivatives (partial or iterated) of expressions
involving formal operators. All the dependencies must be explicit. This
returns 0 since F (so far) does not explicitly depend on x.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest(’p7’);"
    value="D(F,x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Suppose that we have F a function of x, y, and z, where x and y are
```

```
themselves functions of z. Start by declaring that F, x, and y are
operators.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="F:=operator 'F; x:=operator 'x; y:=operator 'y" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
You can use F, x, and y in expressions.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p8','p9']);"
    value="a:=F(x z, y z, z^2)+x y(z+1)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Differentiate formally with respect to z. The formal derivatives appearing
in dadz are not just formal symbols, but do represent derivatives of x, y, and
F.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p8','p9','p10']);"
    value="dadz:=D(a,z)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
You can evaluate the above for particular functional values of F, x, and y.
If x(z) is exp(z) and y(z) is log(z+1), then this evaluates dadz.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p8','p9','p10','p11']);"
    value="eval(eval(dadz,'x,z+->exp z),'y,z+->log(z+1))" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
You obtain the same result by first evaluating a and then differentiating.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p8','p9','p10','p12']);"
    value="m:=eval(eval(a,'x,z+->exp z),'y,z+->log(z+1))" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p8','p9','p10','p12','p13']);"
    value="D(m,z)" />
```

```
  <div id="ansp13"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———-

calintegrals.xhtml

— calintegrals.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Integration</div>
  <hr/>
Axiom has extensive library facilities for integration.

The first example is the integration of a fraction with a denominator that
factors into a quadratic and a quartic irreducible polynomial. The usual
partial fraction approach used by most other computer algebra systems either
fails or introduces expensive unneeded algebraic numbers.

We use a factorization-free algorithm.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="integrate((x^2+2*x+1)/((x+1)^6+1),x)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
When real parameters are present, the form of the integral can depend on the
signs of some expressions.

Rather than query the user or make sign assumptions, Axiom returns all
possible answers.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="integrate(1/(x^2+a),x)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The <a href="dbopintegrate.xhtml">integrate</a> operation generally assumes
that all parameters are real. The only exception is when the integrand has
complex valued quantities.
```

```
If the parameter is complex instead of real, then the notion of sign is
undefined and there is a unique answer. You can request this answer by
"prepending" the word "complex" to the command name.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="complexIntegrate(1/(x^2+a),x)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The following two examples illustrate the limitations of table-based
approaches. The two integrands are very similar, but the answer to one of
them requires the addition of two new algebraic numbers.

This is the easy one. The next one looks very similar but the answer is
much more complicated.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="integrate(x^3/(a+b*x)^(1/3),x)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Only an algorithmic approach is guaranteed to find what new constants must
be added in order to find a solution.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="integrate(1/(x^3*(a+b*x)^(1/3)),x)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Some computer algebra systems use heuristics or table-driven approaches to
integration. When these systems cannot determine the answer to an
integration problem, they reply "I don't know". Axiom uses an algorithm
for integration that conclusively proves that an integral cannot be expressed
in terms of elementary functions.

When Axiom returns an integral sign, it has proved that no answer exists as
an elementary function.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="integrate(log(1+sqrt(a*x+b))/x,x)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Axiom can handle complicated mixed functions much beyond what you can find
in tables. Whenever possible, Axiom tries to express the answer using the
```

```
functions present in the integrand.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="integrate((sinh(1+sqrt(x+b))+2*sqrt(x+b))/(sqrt(x+b)*(x+cosh(1+sqrt(x+b)))),x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
A strong structure-checking algorithm in Axiom finds hidden algebraic
relationships between functions.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="integrate(tan(atan(x)/3),x)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
The discovery of this algebraic relationship is necessary for correct
integration of this function. Here are the details:
<ol>
 <li>
  If x=tan(t) and g=tan(t/3) then the following algebraic relationship is true:
<pre>
      g^3 - 3xg^2 - 3g + x = 0
</pre>
 </li>
 <li>
  Integrate g using this algebraic relation; this produces:
<pre>
((24g^2-8)log(3g^2-1) + (81x^2+24)g^2 + 72xg - 27x^2 - 16) / (54g^2 - 18)
</pre>
 </li>
 <li>
  Rationalize the denominator, producing:
<pre>
     (8log(3g^2-1) - 3g^2 + 18xg + 16)/18
</pre>
  Replace g by the initial definition g=tan(arctan(x)/3) to produce the
final result.
 </li>
</ol>
This is an example of a mixed function where the algebraic layer is over
the transcendental one.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="integrate((x+1)/(x*(x+log x)^(3/2)),x)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
While incomplete for non-elementary functions, Axiom can handle some of them.
```

```
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="integrate(exp(-x^2)*erf(x)/(erf(x)^3-erf(x)^2-erf(x)+1),x)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
More examples of Axiom's integration capabilities are discussed in
<a href="axbook/section-8.8.xhtml">Integration</a>.
\getchunk{page foot}
```

callaplace.xhtml

— callaplace.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Laplace Transforms</div>
  <hr/>
Axiom can compute some forward Laplace transforms, mostly of elementary
functions not involving logarithms, although some cases of special functions
are handled. To compute the forward Laplace transform of F(t) with respect
to t and express the result as f(s), issue the command laplace(F(t),t,s).
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="laplace(sin(a*t)*cosh(a*t)-cos(a*t)*sinh(a*t),t,s)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Here are some other non-trivial examples.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="laplace((exp(a*t)-exp(b*t))/t,t,s)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="laplace(2/t*(1-cos(a*t)),t,s)" />
  <div id="ansp3"><div></div></div>
```

```
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="laplace(exp(-a*t)*sin(b*t)/b^2,t,s)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="laplace((cos(a*t)-cos(b*t))/t,t,s)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Axiom also knows about a few special functions.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="laplace(exp(a*t+b)*Ei(c*t),t,s)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="laplace(a*Ci(b*t)+c*Si(d*t),t,s)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
When Axiom does not know about a particular transform, it keeps it as a
formal transform in the answer.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="laplace(sin(a*t)-a*t*cos(a*t)+exp(t^2),t,s)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

callimits.xhtml

— callimits.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
```

```
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Limits</div>
  <hr/>
To compute a limit, you must specify a functional expression, a variable,
and a limiting value for that variable. If you do not specify a direction,
Axiom attempts to compute a two-sided limit.

Issue this to compute the limit of (x^2-2*x+2)/(x^2-1) as x approaches 1.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="limit((x^2-3*x+2)/(x^2-1),x=1)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Sometimes the limit when approached from the left is different from the
limit from the right and, in this case, you may wish to ask for a one-sided
limit. Also, if you have a function that is only defined on one side of a
particular value, you can compute a one-sided limit.

The function log(x) is only defined to the right of zero, that is, for
x>0. Thus, when computing limits of functions involving log(x), you probably
want a "right-hand" limit.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value='limit(x*log(x),x=0,"right")' />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
When you do not specify "right" or "left" as the optional fourth argument,
<a href="dboplimit.xhtml">limit</a> tries to compute a two-sided limit.
Here the limit from the left does not exist, as Axiom indicates when you
try to take a two-sided limit.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="limit(x*log(x),x=0)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
A function can be defined on both sides of a particular value, but tend to
different limits as its variable approaches that value from the left and
from the right. We can construct an example of this as follows: Since
sqrt(y^2) is simply the absolute value of y, the function sqrt(y^2)/y is
simply the sign (+1 or -1) of the nonzero real number y. Therefore,
sqrt(y^2)/y=-1 for y&#60;0 and sqrt(y^2)/y=+1 for y>0. This is what happens
when we take the limit at y=0. The answer returned by Axiom gives both a
"left-handed" and a "right-handed" limit.
<ul>
```

```
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="limit(sqrt(y^2)/y,y=0)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Here is another example, this time using a more complicated function.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="limit(sqrt(1-cos(t))/t,t=0)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
You can compute limits at infinity by passing either "plus infinity" or
"minus infinity" as the third argument of <a href="dboplimit.xhtml">limit</a>.
To do this, use the constants %plusInfinity and %minusInfinity.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="limit(sqrt(3*x^2+1)/(5*x),x=%plusInfinity)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="limit(sqrt(3*x^2+1)/(5*x),x=%minusInfinity)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
You can take limits of functions with parameters. As you can see, the limit
is expressed in terms of the parameters.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="limit(sinh(a*x)/tan(b*x),x=0)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
When you use <a href="dboplimit.xhtml">limit</a>, you are taking the limit
of a real function of a real variable. When you compute this, Axiom returns
0 because, as a function of a real variable, sin(1/z) is always between -1
and 1, so z*sin(1/z) tends to 0 as z tends to 0.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="limit(z*sin(1/z),z=0)" />
  <div id="ansp9"><div></div></div>
 </li>
```

```
</ul>
However, as a function of a complex variable, sin(1/z) is badly behaved
near 0 (one says that sin(1/z) has an essential singularlity at z=0). When
viewed as a function of a complex variable, z*sin(1/z) does not approach any
limit as z tends to 0 in the complex plane. Axiom indicates this when we
call <a href="dbopcomplexlimit.xhtml">complexLimit</a>.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="complexLimit(z*sin(1/z),z=0)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
You can also take complex limits at infinity, that is, limits of a function
of z as z approaches infinity on the Riemann sphere. Use the symbol
%infinity to denote "complex infinity". As above, to compute complex limits
rather than real limits, use <a href="dbopcomplexlimit.xhtml">complexLimit</a>.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="makeRequest('p11');"
    value="complexLimit((2+z)/(1-z),z=%infinity)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
In many cases, a limit of a real function of a real variable exists when
the corresponding complex limit does not. This limit exists.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="makeRequest('p12');"
    value="limit(sin(x)/x,x=%plusInfinity)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
But this limit does not.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="complexLimit(sin(x)/x,x=%infinity)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

calmoreintegrals.xhtml

— calmoreintegrals.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Integration</div>
  <hr/>
Integration is the reverse process of differentiation, that is, an integral
of a function f with respect to a variable x is any function g such that
D(g,x) is equal to f. The package
<a href="db.xhtml?FunctionSpaceIntegration">FunctionSpaceIntegration</a>
provides the top-level integration operation
<a href="dbopintegrate.xhtml">integrate</a>, for integrating real-valued
elementary functions.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="integrate(cosh(a*x)*sinh(a*x),x)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Unfortunately, antiderivatives of most functions cannot be expressed in
terms of elementary functions.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="integrate(log(1+sqrt(a*x+b)),x)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Given an elementary function to integrate, Axiom returns a formal integral
as above only when it can prove that the integral is not elementary and
not when it cannot determine the integral. In this rare case it prints a
message that it cannot determine if an elementary integral exists. Similar
functions may have antiderivatives that look quite different because the
form of the antiderivative depends on the sign of a constant that appears
in the function.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="integrate(1/(x^2-2),x)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="integrate(1/(x^2+2),x)" />
  <div id="ansp4"><div></div></div>
```

```
 </li>
</ul>
If the integrand contains parameters, then there may be several possible
antiderivatives, depending on the signs of expressions of the parameters.
In this case Axiom returns a list of answers that cover all possible cases.
Here you use the answer involving the square root of a when a>0 and the
answer involving the square root of -a when a&#60;0.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest(’p5’);"
    value="integrate(x^2/(x^4-a^2),x)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
If the parameters and the variables of integration can be complex numbers
rather than real, then the notion of sign is not defined. In this case all
the possible answers can be expressed as one complex function. To get that
function, rather than a list of real functions, use
<a href="dbopcomplexintegrate.xhtml">complexIntegrate</a>, which is provided
by the package
<a href="db.xhtml?FunctionSpaceComplexIntegration">
FunctionSpaceComplexIntegration</a>.

This operation is used for integrating complex-valued elementary functions.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest(’p6’);"
    value="complexIntegrate(x^2/(x^4-a^2),x)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
As with the real case, antiderivatives for most complex-valued functions
cannot be expressed in terms of elementary functions.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest(’p7’);"
    value="complexIntegrate(log(1+sqrt(a*x+b))/x,x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Sometimes <a href="dbopintegrate.xhtml">integrate</a> can involve
symbolic algebraic numbers such as those returned by
<a href="dboprootof.xhtml">rootOf</a>. To see how to work with these
strange generated symbols (such as %%a0), see
<a href="axbook/section-8.3.xhtml#subsec-8.3.2">
Using All Roots of a Polynomial</a>.

Definite integration is the process of computing the area between the x-axis
and the curve of a function f(x). The fundamental theorem of calculus
states that if f is continuous on an interval a..b and such that D(g,x) is
equal to f, then the definite integral of f for x in the interval a..b is
```

```
equal to g(b)-g(a).

The package
<a href="db.xhtml?RationalFunctionDefiniteIntegration">
RationalFunctionDefiniteIntegration</a>
provides the top-level definite integration operation,
<a href="dbopintegrate.xhtml">integrate</a>,
for integrating real-valued rational functions.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="integrate((x^4-3*x^2+6)/(x^6-5*x^4+5*x^2+4),x=1..2)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Axiom checks beforehand that the function you are integrating is defined on
the interval a..b, and prints an error message if it finds that this is not
the case, as in the following example:
<pre>
 integrate(1/(x^2-2),x=1..2)

 Error detected within library code:
 Pole in path of integration
</pre>
When parameters are present in the function, the function may or may not be
defined on the interval of integration.

If this is the case, Axiom issues a warning that a pole might lie in the
path of integration, and does not compute the integral.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="integrate(1/(x^2-a),x=1..2)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
If you know that you are using values of the parameter for which the
function has no pole in the interval of integration, use the string
"noPole" as a third argument to <a href="dbopintegrate.xhtml">integrate</a>.

The value here is, of course, incorrect if sqrt(a) is between 1 and 2.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value='integrate(1/(x^2-a),x=1..2,"noPole")' />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

calseries.xhtml

— calseries.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Working with Power Series</div>
  <hr/>
Axiom has very sophisticated facilities for working with power series.
Infinite series are represented by a list of the coefficients that have
already been determined, together with a function for computing the
additional coefficients if needed. The system command that determines how
many terms of a series is displayed is
<pre>
  )set streams calculate
</pre>
By default Axiom will display ten terms. Series can be created from
expressions, from functions for the series coefficients, and from applications
of operations on existing series. The most general function for creating
a series is called <a href="dbopseries.xhtml">series</a>, although you can
also use
<a href="dboptaylor.xhtml">taylor</a>,
<a href="dboplaurent.xhtml">laurent</a>, and
<a href="dboppuiseux.xhtml">puiseux</a> in situations where you know what
kind of exponents are involved.

For information about solving differential equations in terms of power
series see
<a href="axbook/section-8.10.xhtml#subsec-8.10.3">
Power Series Solutions of Differential Equations</a>
<ul>
 <li>
  <a href="calseries1.xhtml">
   Creation of Power Series
  </a>
 </li>
 <li>
  <a href="calseries2.xhtml">
   Coefficients of Power Series
  </a>
 </li>
 <li>
  <a href="calseries3.xhtml">
   Power Series Arithmetic
  </a>
 </li>
 <li>
  <a href="calseries4.xhtml">
   Functions on Power Series
  </a>
 </li>
```

```
 <li>
  <a href="calseries5.xhtml">
   Converting to Power Series
  </a>
 </li>
 <li>
  <a href="calseries6.xhtml">
   Power Series from Formulas
  </a>
 </li>
 <li>
  <a href="calseries7.xhtml">
   Substituting Numerical Values in Power Series
  </a>
 </li>
 <li>
  <a href="calseries8.xhtml">
   Example: Bernoulli Polynomials and Sums of Powers
  </a>
 </li>
</ul>
\getchunk{page foot}
```

calseries1.xhtml

— calseries1.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Creation of Power Series</div>
  <hr/>
This is the easiest way to create a power series. This tells Axiom that x
is to be treated as a power series, so funcitons of x are again power series.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="x:=series 'x" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
We didn't say anything about the coefficients of the power series, so the
coefficients are general expressions over the integers. This allows us to
introduce denominators, symbolic constants, and other variables as needed.
Here the coefficents are integers (note that the coefficients are the
```

```
Fibonacci numbers).
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="1/(1-x-x^2)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
This series has coefficients that are rational numbers.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="sin(x)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
When you enter this expression you introduce the symbolic constants sin(1)
and cos(1).
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="sin(1+x)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
When you enter the expression the variable a appears in the resulting
series expansion.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="sin(a*x)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
You can also convert an expression into a series expansion. This expression
creates the series expansion of 1/log(v) about v=1. For details and more
examples see
<a href="axbook/section-8.9.xhtml#subsec-8.9.5">
Converting to Power Series</a>
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="series(1/log(v),v=1)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
You can create power series with more general coefficients. You normally
accomplish this via a type declaration, see
<a href="axbook/section-2.3.xhtml">Declarations</a>. See
```

```
<a href="axbook/section-8.9.xhtml#subsec-8.9.4">
Functions on Power Series</a> for some warnings about working with
declared series.

We delcare that y is a one-variable Taylor series
(<a href="db.xhtml?UnivariateTaylorSeries">UTS</a> is the abbreviation for
<a href="db.xhtml?UnivariateTaylorSeries">UnivariateTaylorSeries</a> in the
variable z with <a href="db.xhtml?Float">FLOAT</a> (that is, floating-point)
coefficients, centered about 0. Then, by assignment, we obtain the Taylor
expansion of exp(z) with floating-point coefficients.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="y:UTS(FLOAT,'z,0):=exp(z)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
You can also create a power series by giving an explicit formula for the
nth coefficient. For details and more examples see
<a href="axbook/section-8.9.xhtml#subsec-8.9.6">
Power Series from Formulas</a>

To create a series about w=0 whose nth Taylor coefficient is 1/n!, you can
evaluate this expression. This is the Taylor expansion of exp(w) at w=0.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="series(1/factorial(n),n,w=0)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

calseries2.xhtml

— calseries2.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Coefficients of Power Series</div>
  <hr/>
You can extract any coefficient from a power series -- even on that
hasn't been computed yet. This is possible because in Axiom, infinite
```

```
series are represented by a list of the coefficients that have already
been determined, together with a function for computing additional
coefficients. (This is known as lazy evaluation.) When you ask for a
coefficient that hasn't yet been computed, Axiom computes whatever
additional coefficients it needs and then stores them in the representation
of the power series.

Here's an example of how to extract the coefficients of a power series.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="x:=series('x)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="y:=exp(x)*sin(x)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
This coefficient is readily available
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="coefficient(y,6)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
But let's get the fifteenth coefficient of y
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p4']);"
    value="coefficient(y,15)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
If you look at y then you see that the coefficients up to order 15 have
all been computed.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p4','p5']);"
    value="y" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

calseries3.xhtml

— calseries3.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Power Series Arithmetic</div>
  <hr/>
You can manipulate power series using the usual arithmetic operations
<a href="dbopplus.xhtml">+</a>,
<a href="dbopminus.xhtml">-</a>,
<a href="dbopstar.xhtml">*</a>, and
<a href="dbopdivide.xhtml">/</a>.

The results of these operations are also power series.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="x:=series 'x" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="(3+x)/(1+7*x)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
You can also compute f(x)^g(x), where f(x) and g(x) are two power series.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="base:=1/(1-x)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p3','p4']);"
    value="expon:=x*base" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p3','p4','p5']);"
    value="base^expon" />
```

```
  <div id="ansp5"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

calseries4.xhtml

— calseries4.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Functions on Power Series</div>
  <hr/>
Once you have created a power series, you can apply transcendental
functions (for example,
<a href="dbopexp.xhtml">exp</a>,
<a href="dboplog.xhtml">log</a>,
<a href="dbopsin.xhtml">sin</a>,
<a href="dboptan.xhtml">tan</a>,
<a href="dbopcosh.xhtml">cosh</a>, etc.) to it.

To demonstrate this, we first create the power series expansion of the
rational function x^2/(1-6*x+x^2) about x=0.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="x:=series 'x" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="rat:=x^2/(1-6*x+x^2)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
If you want to compute the series expansion of
sin(x^2/1-6*x+x^2) you simply compute the sine of rat.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="sin(rat)" />
  <div id="ansp3"><div></div></div>
```

```
 </li>
</ul>
<hr/>
<b>Warning:</b> the type of the coefficients of a power series may affect
the kind of computations that you can do with that series. This can only
happen when you have made a declaration to specify a series domain with a
certain type of coefficient.
<hr/>
If you evaluate then you have declared that y is a one variable Taylor
series (<a href="db.xhtml?UnivariateTaylorSeries">UTS</a> is the abbreviation
for <a href="db.xhtml?UnivariateTaylorSeries">UnivariateTaylorSeries</a>) in
the variable y with <a href="dbfractioninteger.xhtml">FRAC INT</a> (that is,
fractions of integers) coefficients, centered about 0.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="y:UTS(FRAC INT,'y,0):='y" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
You can now compute certain power series in y, provided that these series
have rational coefficients.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="exp(y)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
You can get examples of such series by applying transcendental functions
to series in y that have no constant terms.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p4','p5','p6']);"
    value="tan(y^2)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p4','p5','p7']);"
    value="cos(y+y^5)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Similarly, you can compute the logarithm of a power series with rational
coefficients if the constant coefficient is 1.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p4','p5','p8']);"
    value="log(1+sin(y))" />
```

```
  <div id="ansp8"><div></div></div>
 </li>
</ul>
If you wanted to apply, say, the operation <a href="dbopexp.xhtml">exp</a> to
a power series with a nonzero constant coefficient a0, then the constant
coefficient of the result would be exp(a0), which is not a rationa number.
Therefore, evaluating exp(2+tan(y)) would generate an error message.

If you want to compute the Taylor expansion of exp(2+tan(y)), you must
ensure that the coefficient domain has an operation
<a href="dbopexp.xhtml">exp</a> defined for it. An example of such a domain
is <a href="dbexpressioninteger.xhtml">Expression Integer</a>, the type of
formal functional expressions over the integers. When working with
coefficients of this type
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="z:UTS(EXPR INT,'z,0):='z" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
this presents no problems.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p9','p10']);"
    value="exp(2+tan(z))" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
Another way to create Taylor series whose coefficients are expressions over
the integers is to use <a href="dboptaylor.xhtml">taylor</a> which works
similarly to <a href="dbopseries.xhtml">series</a>. This is equivalent to
the previous computation, except that now we are using the variable w
instead of z.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="makeRequest('p11');"
    value="w:=taylor 'w" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p11','p12']);"
    value="exp(2+tan(w))" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

calseries5.xhtml

— calseries5.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Converting to Power Series</div>
  <hr/>
The <a href="db.xhtml?ExpressionToUnivariatePowerSeries">
ExpressionToUnivariatePowerSeries</a> package provides operations for
computing series expansions of functions.

Evaluate this to compute the Taylor expansion of sin x about x=0. The first
argument, sin(x), specifies the function whose series expansion is to be
computed and the second argument, x=0, specifies that the series is to be
expanded in powers of (x-0), that is, in powers of x.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="taylor(sin(x),x=0)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Here is the Taylor expansion of sin x about x=%pi/6:
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="taylor(sin(x),x=%pi/6)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The function to be expanded into a series may have variables other than the
series variable. For example, we may expand tan(x*y) as a Taylor series in x.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="taylor(tan(x*y),x=0)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
or as a Taylor series in y.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
```

```
    onclick="makeRequest('p4');"
    value="taylor(tan(x*y),y=0)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
A more interesting function it (t*%e^(x*t))/(%e^t-1).
When we expand this function as a Taylor series in t the nth order
coefficient is the nth Bernoulli polynomial divided by n!.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="bern:=taylor(t*exp(x*t)/(exp(t)-1),t=0)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Therefore, this and the next expression produce the same result.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p5','p6']);"
    value="factorial(6)*coefficient(bern,6)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value="bernoulliB(6,x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Technically, a series with terms of negative degree is not considered to
be a Taylor series, but rather a Laurent series. If you try to compute a
Taylor series expansion of x/log(x) at x=1 via taylor(x/log(x),x=1) you
get an error message. The reason is that the function has a pole at x=1,
meaning that its series expansion about this point has terms of negative
degree. A series with finitely many terms of negative degree is called a
Laurent series. You get the desired series expansion by issuing this.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="laurent(x/log(x),x=1)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Similarly, a series with terms of fractional degree is neither a Taylor
series nor a Laurent series. Such a series is called a Puiseux series. The
expression laurent(sqrt(sec(x)),x=3*%pi/2) results in an error message
because the series expansion about this point has terms of fractional degree.
However, this command produces what you want.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
```

```
    onclick="makeRequest('p9');"
    value="puiseux(sqrt(sec(x)),x=3*%pi/2)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Finally, consider the case of functions that do not have Puiseux expansions
about certain points. An example of this is x^x about x=0. puiseux(x^x,x=0)
produces an error message because of the type of singularity of the
function at x=0. The general function <a href="dbopseries.xhtml">series</a>
can be used in this case. Notice that the series returned is not, strictly
speaking, a power series because of the log(x) in the expansion.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="series(x^x,x=0)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
<hr/>
The operation <a href="dbopseries.xhtml">series</a> returns the most general
type of infinite series. The user who is not interested in distinguishing
between various types of infinite series may wish to use this operation
exclusively.
<hr/>
\getchunk{page foot}
```

calseries6.xhtml

— calseries6.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Power Series from Formulas</div>
  <hr/>
The <a href="db.xhtml?GenerateUnivariatePowerSeries">
GenerateUnivariatePowerSeries</a> package enables you to create power series
from explicit formulas for their nth coefficients. In what follows, we
construct series expansions for certain transcendental functions by giving
forumulas for their coefficients. You can also compute such series
expansions directly by simply specifying the function and the point about
which the series is to be expanded. See
<a href="axbook/section-8.9.xhtml#subsec-8.9.5">
Converting to Power Series</a> for more information.
```

```
Consider the Taylor expansion of %e^x about x=0:
<pre>
  %e^x = 1 + x + x^2/2 + x^3/6 + ...
       = sum from n=0 to n=%infinity of x^n/n!
</pre>
The nth Taylor coefficient is 1/n!. This is how to create this series in
Axiom.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="series(n+->1/factorial(n),x=0)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The first argument specifies the formula for the nth coefficient by giving
a function that maps n to 1/n!. The second argument specifies that the
series is to be expanded in powers of (x-0), that is, in powers of x. Since
we did not specify an initial degress, the first term in the series was the
term of degree 0 (the constant term). Note that the formula was given as
an anonymous function. These are discussed in
<a href="axbook/section-6.17.xhtml">Anonymous Functions</a>

Consider the Taylor expansion of log x about x=1:
<pre>
 log x = (x-1) - (x-1)^2/2 + (x-1)^3/3 - ...
       = sum from n=1 to n=%infinity of (-1_^(n-1) (x-1)^n/n
</pre>
If you were to evaluate the expression series(n+->(-1)^(n-1)/n,x=1) you
would get an error message because Axiom would try to calculate a term of
degree n=1,... are to be computed.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="series(n+->(-1)^(n-1)/n,x=1,1..)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Next consider the Taylor expansion of an odd function, say, sin(x):
<pre>
  sin x = x = x^2/3! + x^5/5! - ...
</pre>
Here every other coefficient is zero and we would like to give an explicit
formula onloy for the odd Taylor coefficients. This is one way to do it.
The third argument, 1.., specifies that the first term to be computed is
the term of degree 1. The fourth argument, 2, specifies that we increment
by 2 to find the degrees of subsequent terms, that is, the next term is of
degree 1+2, the next of degree 1+2+2, etc.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="series(n+->(-1)^((n-1)/2)/factorial(n),x=0,1..,2)" />
```

```
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The initial degree and the increment do not have to be integers. For
example, this expression produces a series expansion of sin(x^(1/3)).
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest(’p4’);"
    value="series(n+->(-1)^((3*n-1)/2)/factorial(3*n),x=0,1/3..,2/3)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
While the increment must be positive, the initial degree may be negative.
This yields the Laurent expansion of csc(x) at x=0.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest(’p5’);"
    value="cscx:=series(n+->(-1)^((n-1)/2)*2*(2^n-1)*bernoulli(numer(n+1))/factorial(n+1),x=0,-1..,2)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Of course, the reciprocal of this power series is the Taylor expansion of
sin(x).
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree([’p5’,’p6’]);"
    value="1/cscx" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
As a final example, here is the Taylor expansion of asin(x) about x=0.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest(’p7’);"
    value="asinx:=series(n+->binomial(n-1,(n-1)/2)/(n*2^(n-1)),x=0,1..,2)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
When we compute the sine of this series, we get x (in the sense that all
higher terms computed so far are zero).
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree([’p7’,’p8’]);"
    value="sin(asinx)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
As we discussed in
<a href="calseries5.xhtml">Converting to Power Series</a>, you can also use
```

```
the operations
<a href="dboptaylor.xhtml">taylor</a>,
<a href="dboplaurent.xhtml">laurent</a>, and
<a href="dboppuiseux.xhtml">puiseux</a>, instead of
<a href="dbopseries.xhtml">series</a> if you know ahead of time what
kind of exponents a series has. You can't go wrong with
<a href="dbopseries.xhtml">series</a> though.
\getchunk{page foot}
```

——–

calseries7.xhtml

— calseries7.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Substituting Numerical Values in Power Series</div>
  <hr/>
Use <a href="dbopeval.xhtml">eval</a> to substitute a numerical value for a
variable in a power series. For example, here's a way to obtain numerical
approximations of %e from the Taylor series expansion of exp(x).

First you create the desired Taylor expansion.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="f:=taylor(exp(x))" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Then you evaluate the series at the value 1.0. The result is a sequence
of the partial sums.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="eval(f,1.0)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

——–

calseries8.xhtml

— calseries8.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Example: Bernoulli Polynomials and Sums of Powers</div>
  <hr/>
Axiom provides operations for computing definite and indefinite sums.

You can compute the sum of the first ten fourth powers by evaluating this.
This creates a list whose entries are m^4 as m ranges from 1 to 10, and then
computes the sum of the entries of that list.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="reduce(+,[m^4 for m in 1..10])" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can also compute a formula for the sum of the first k fourth powers,
where k is an unspecified positive integer.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="sum4:=sum(m^4,m=1..k)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
This formula is valid for any positive integer k. For instance, if we
replace k by 10, we obtain the number we computed earlier.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="eval(sum4,k=10)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
You can compute a formula for the sum of the first k nth powers in a
similar fashion. Just replace the 4 in the definition of sum4 by any
expression not involving k. Axiom computes these formulas using Bernoulli
polynomials; we use the rest of this section to describe this method.

First consider this function of t and x.
```

```
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="f:=t*exp(x*t)/(exp(t)-1)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Since the expressions involved get quite large, we tell Axiom to show us only
terms of degree up to 5.
<ul>
 <li>
  <input type="submit" id="p5" class="noresult"
    onclick="makeRequest('p5');"
    value=")set streams calculate 5" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
If we look at the Taylor expansion of f(x,t) about t=0, we see that the
coefficients of the powers of t are polynomials in x.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p4','p5','p6']);"
    value="ff:=taylor(f,t=0)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
In fact, the nth coefficient in this series is essentiall the nth Bernoulli
polynomial: the nth coefficient of the series is 1/n!*Bn(x), where Bn(x) is
the nth Bernoulli polynomial. Thus, to obtain the nth Bernoulli polynomial,
we multiply the nth coefficient of the series ff by n!. For example, the
sixth Bernoulli polynomial is this.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p4','p5','p6','p7']);"
    value="factorial(6)*coefficient(ff,6)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
We derive some properties of the function f(x,t). First we compute
f(x+1,t)-f(x-t).
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p4','p8']);"
    value="g:=eval(f,x=x+1)-f" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
If we normalize g, we see that it has a particularly simple form.
<ul>
 <li>
```

```
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p4','p8','p9']);"
    value="normalize(g)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
From this it follows that the nth coefficient in the Taylor expansion of
g(x,t) at t=0 is 1/(n-1)!*x^(n-1). If you want to check this, evaluate the
next expression.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p4','p5','p8','p9','p10']);"
    value="taylor(g,t=0)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
However, since
<pre>
  g(x,t)=f(x+1,t)-f(x,t)
</pre>
it follows that the nth coefficient
is
<pre>
   1/n! * (Bn(x+1) - Bn(x))
</pre>
Equating coefficients, we see that
<pre>
   1/(n-1)! * x^(n-1) = 1/n! * (Bn(x+1) - Bn(x))
</pre>
and, therefore
<pre>
   x^(n-1) = 1/n * (Bn(x+1) - Bn(x))
</pre>
Let's apply this formula repeatedly, letting x vary between two integers
a and b, with a&#60;b:
<pre>
       a^(n-1) = 1/n * (Bn(a+1) - Bn(a))
   (a+1)^(n-1) = 1/n * (Bn(a+2) - Bn(a+1))
   (a+2)^(n-1) = 1/n * (Bn(a+3) - Bn(a+2))
               .
               .
   (b-1)^(n-1) = 1/n * (Bn(b)   - Bn(b-1))
       b^(n-1) = 1/n * (Bn(b+1) - Bn(b))
</pre>
When we add these equations we find that the sum of the left-hand sides is
<pre>
   sum(m=a..b,m^(n-1))
</pre>
the sum of the (n-1)-st powers from a to b. The sum
of the right-hand sides is a "telescoping series". After cancellation, the
sum is simply
<pre>
   1/n*(Bn(b+1)-Bn(a))
```

```
</pre>

Replacing n by n+1, we have shown that
<pre>
   sum(m=a..b,m^n) = 1/(n+1)*(B&#60;n+1>(b+1)-B&#60;n+1>(a))
</pre>

Let's use this to obtain the formula for the sum of fourth powers.
First we obtain the Bernoulli polynomial B5.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p4','p5','p6','p11']);"
    value="B5:=factorial(5)*coefficient(ff,5)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
To find the sum of the first k 4th powers, we multiply 1/5 by
B5(k+1)-B5(1)
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p4','p5','p6','p11','p12']);"
    value="1/5*(eval(B5,x=k+1)-eval(B5,x=1))" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
This is the same formula that we obtained via sum(m^4,m=1..k)
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p2','p13']);"
    value="sum4" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
At this point you may want to do the same computation, but with an exponent
other than 4. For example, you might try to find a formula for the sum of
the first k 20th powers.
\getchunk{page foot}
```

cats.xhtml

— cats.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
    CATS -- Computer Algebra Test Suite
```

```
  </div>
<hr/>
The Computer Algebra Test Suite is intended to show that Axiom conforms
to various published standards. Axiom implementations of these functions
are tested against reference publications.

In order to show standards compliance we need to examine Axiom's behavior
against known good results. Where possible, these results are also tested
against other available computer algebra systems.

The available test suites are:
<ol>
 <li><a href="dlmf.xhtml">Gamma Function</a></li>
</ol>
\getchunk{page foot}
```

commandline.xhtml

— commandline.xhtml —

```
\getchunk{standard head}
\getchunk{menu style}
  <script type="text/javascript">
   function commandline(arg) {
    return(document.getElementById('comm').value);
   }
   function popUp(url,widthsize,heightsize) {
     var widthsize=640;
     var heightsize=480;
     var p1 = "toolbar=no";
     var p2 = ",location=no";
     var p3 = ",directories=no";
     var p4 = ",status=no";
     var p5 = ",menubar=no";
     var p6 = ",scrollbars=no";
     var p7 = ",resizable=no";
     var p8 = ",top="+(window.screen.height-heightsize)/2;
     var p9 = ",left="+(window.screen.width-widthsize)/2;
     var pa = ",width="+widthsize;
     var pb = ",height="+heightsize;
     var winProps=p1+p2+p3+p4+p5+p6+p7+p8+p9+pa+pb;
     var popUp = window.open(url,"popup",winProps);
     popUp.focus();
   }
   var card = '{is:"card",'+
               'title:"",'+
               'content:"",'+
               'type:"",'+
               'visible:"false"}';
   function newcard(title,content,type,visible) {
      var c = eval("("+card+")");
```

```
      c.title=title;
      c.content=content;
      c.type=type;
      c.visible=visible;
      return(c);
   }
   var pamphlet = '{is:"pamphlet",'+
                  'head:"",'+
                  'body:[],'+
                  'tail:"",'+
                  'visible:"false"}';
   function newpamphlet(title,content,type,visible) {
      var p = eval("("+pamphlet+")");
      p.title=title;
      p.content=content;
      p.type=type;
      p.visible=visible;
      return(p);
   }
   function dump(d) {
     if (d.is == "card") {
      return("CARD"+"\\n"+
             "title:  "+d.title+"\\n"+
             "content:"+d.content+"\\n"+
             "visible:"+d.visible+"\\n"+
             "type:   "+d.type);
     }
     if (d.is == "pamphlet") {
      return("PAMPHLET"+"\\n"+
             "title:  "+d.title+"\\n"+
             "content:"+d.content+"\\n"+
             "visible:"+d.visible+"\\n"+
             "type:   "+d.type);
     }
   }
   function makeone() {
     var p = newpamphlet("makeone",card,"text","true");
     alert(dumppamphlet(p));
     alert(dump(p));
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <!--main menu-->
   <form method="get" action="foo.xhtml">
    <table class="mainmenu" border="0" cellspacing="0" cellpadding="0">
     <tr>
      <td align="center" valign="middle" nowrap="nowrap">
       <div class="menu">
        <ul>
```

```
<!-- Begin File Menu -->
<li>
 <a class="toplevel" href="/">
  File<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop" href="javascript:popUp('menufileopen.xhtml')">
    Open
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menufileread.xhtml')">
    Read file
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menufilesave.xhtml')">
    Save
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menufilesaveas.xhtml')">
    Save as
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menufileloadlibrary.xhtml')">
    Load library
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menufileinputfile.xhtml')">
    Input file
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menufiletogglespool.xhtml')">
    Toggle spool
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menufileprint.xhtml')">
    Print
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menufileexit.xhtml')">
    Exit
   </a>
  </li>
 </ul>
```

```
</li>
<!-- End File Menu -->
<!-- Start Edit Menu -->
<li>
 <a class="toplevel" href="/">
  Edit<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop" href="javascript:popUp('menueditcopy.xhtml')">
    Copy
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menueditcopytext.xhtml')">
    Copy text
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menueditcopytex.xhtml')">
    Copy TeX
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menueditdeleteselection.xhtml')">
     Delete selection
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menueditcopyasimage.xhtml')">
    Copy as image
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menueditselectiontoimage.xhtml')">
    Selection to image
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menueditselectiontoinput.xhtml')">
    Selection to input
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menueditcut.xhtml')">
    Cut
   </a>
  </li>
  <li>
   <a class="drop" href="javascript:popUp('menueditpaste.xhtml')">
```

```
    Paste
   </a>
  </li>
 </ul>
</li>
<!-- End Edit Menu -->
<!-- Start Axiom Menu -->
<li>
 <a class="toplevel" href="/">
  Axiom<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiominterrupt.xhtml')">
    Interrupt
   </a>
  </li>
  <li>
   <a class="drop"
     href="javascript:popUp('menuaxiomrestart.xhtml')">
    Restart
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomclearmemory.xhtml')">
    Clear Memory
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomaddtopath.xhtml')">
     Add to path
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomshowfunctions.xhtml')">
    Show functions
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomshowdefinition.xhtml')">
    Show definition
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomshowvariables.xhtml')">
    Show variables
   </a>
  </li>
```

```
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomdeletefunction.xhtml')">
    Delete function
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomdeletevariable.xhtml')">
    Delete variable
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomtoggletimedisplay.xhtml')">
    Toggle time display
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomset.xhtml')">
    Set ...
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuaxiomdisplay.xhtml')">
    Display ...
   </a>
  </li>
 </ul>
</li>
<!-- End Axiom Menu -->
<!-- Start Equations Menu -->
<li>
 <a class="toplevel" href="/">
  Equations<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop"
      href="javascript:popUp('menuequationssolve.xhtml')">
    Solve
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuequationssolvenumerically.xhtml')">
    Solve numerically
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuequationsrootsofpolynomial.xhtml')">
```

```
  Roots of polynomial
 </a>
</li>
<li>
 <a class="drop" href=
     "javascript:popUp('menuequationsrealrootsofpolynmial.xhtml')">
   Real roots of polynomial
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menuequationssolvelinearsystem.xhtml')">
  Solve linear system
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menuequationssolvealgebraicsystem.xhtml')">
  Solve algebraic system
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menuequationseliminatevariable.xhtml')">
  Eliminate variable
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menuequationssolveode.xhtml')">
  Solve ODE
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menuequationsinitialvalueproblem1.xhtml')">
  Initial value problem (1)
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menuequationsinitialvalueproblem2.xhtml')">
  Initial value problem (2)
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menuequationsboundaryvalueproblem.xhtml')">
  Boundary value problem
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menuequationssolveodewithlaplace.xhtml')">
```

```
    Solve ODE with Laplace
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menuequationsatvalue.xhtml')">
    At value
   </a>
  </li>
 </ul>
</li>
<!-- End Equations Menu -->
<!-- Start Algebra Menu -->
<li>
 <a class="toplevel" href="/">
  Algebra<!--span class="nabla">&nabla;</span-->
 </a>
 <ul>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebrageneratematrix.xhtml')">
    Generate matrix
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebraentermatrix.xhtml')">
    Enter matrix
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebrainvertmatrix.xhtml')">
    Invert matrix
   </a>
  </li>
  <li>
   <a class="drop" href=
     "javascript:popUp('menualgebracharacteristicpolynomial.xhtml')">
    Characteristic polynomial
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebradeterminant.xhtml')">
    Determinant
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebraeigenvalues.xhtml')">
    Eigenvalues
   </a>
  </li>
```

```
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebraeigenvectors.xhtml')">
    Eigenvectors
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebraadjointmatrix.xhtml')">
    Adjoint matrix
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebratransposematrix.xhtml')">
    Transpose matrix
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebramakelist.xhtml')">
    Make list
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebraapplytolist.xhtml')">
    Apply to list
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebramaptolist.xhtml')">
    Map to list
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebrareducelist.xhtml')">
    Reduce list
   </a>
  </li>
  <li>
   <a class="drop"
      href="javascript:popUp('menualgebramaptomatrix.xhtml')">
    Map to matrix
   </a>
  </li>
 </ul>
</li>
<!-- End Algebra Menu -->
<!-- Start Calculus Menu -->
<li>
 <a class="toplevel" href="/">
```

```
 Calculus<!--span class="nabla">&nabla;</span-->
</a>
<ul>
 <li>
  <a class="drop"
     href="javascript:popUp('menucalculuslevel3.xhtml')">
   Level 3
  </a>
  <ul>
   <li>
    <a class="drop"
     href="javascript:popUp('menucalculuslevel3a.xhtml')">
     Level 3 A
    </a>
   </li>
   <li>
    <a class="drop"
     href="javascript:popUp('menucalculuslevel3b.xhtml')">
     Level 3 B
    </a>
   </li>
   <li>
    <a class="drop"
     href="javascript:popUp('menucalculuslevel3c.xhtml')">
     Level 3 C
    </a>
   </li>
  </ul>
 </li>
 <li>
  <a class="drop"
     href="javascript:popUp('menucalculusintegrate.xhtml')">
   Integrate
  </a>
 </li>
 <li>
  <a class="drop"
     href="javascript:popUp('menucalculusrischintegrate.xhtml')">
   Risch integrate
  </a>
 </li>
 <li>
  <a class="drop"
     href="javascript:popUp('menucalculuschangevariable.xhtml')">
   Change variable
  </a>
 </li>
 <li>
  <a class="drop"
     href="javascript:popUp('menucalculusdifferentiate.xhtml')">
   Differentiate
  </a>
 </li>
 <li>
```

```
 <a class="drop"
    href="javascript:popUp('menucalculusfindlimit.xhtml')">
  Find limit
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menucalculusgetseries.xhtml')">
  Get series
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menucalculuspadeapproximation.xhtml')">
  Pade approximation
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menucalculuscalculussum.xhtml')">
  Calculus sum
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menucalculuscalculusproduct.xhtml')">
  Calculus product
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menucalculuslaplacetransform.xhtml')">
  Laplace transform
 </a>
</li>
<li>
 <a class="drop" href=
   "javascript:popUp('menucalculusinverselaplacetransform.xhtml')">
  Inverse Laplace transform
 </a>
</li>
<li>
 <a class="drop" href=
   "javascript:popUp('menucalculusgreatestcommondivisor.xhtml')">
  Greatest common divisor
 </a>
</li>
<li>
 <a class="drop" href=
   "javascript:popUp('menucalculusleastcommonmultiple.xhtml')">
  Least common multiple
 </a>
</li>
<li>
```

```
    <a class="drop"
       href="javascript:popUp('menucalculusdividepolynomials.xhtml')">
     Divide polynomials
    </a>
   </li>
   <li>
    <a class="drop"
       href="javascript:popUp('menucalculuspartialfractions.xhtml')">
     Partial fractions
    </a>
   </li>
   <li>
    <a class="drop"
      href="javascript:popUp('menucalculuscontinuedfractions.xhtml')">
     Continued fractions
    </a>
   </li>
  </ul>
 </li>
 <!-- End Calculus Menu -->
 <!-- Start Simplify Menu -->
 <li>
  <a class="toplevel" href="/">
   Simplify<!--span class="nabla">&nabla;</span-->
  </a>
  <ul>
   <li>
    <a class="drop" href=
      "javascript:popUp('menusimplifysimplifyexpression.xhtml')">
     Simplify expression
    </a>
   </li>
   <li>
    <a class="drop"
       href="javascript:popUp('menusimplifysimplifyradicals.xhtml')">
     Simplify radicals
    </a>
   </li>
   <li>
    <a class="drop"
       href="javascript:popUp('menusimplifyfactorexpression.xhtml')">
     Factor expression
    </a>
   </li>
   <li>
    <a class="drop"
       href="javascript:popUp('menusimplifyfactorcomplex.xhtml')">
     Factor complex
    </a>
   </li>
   <li>
    <a class="drop"
       href="javascript:popUp('menusimplifyexpandexpression.xhtml')">
     Expand expression
```

```
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menusimplifyexpandlogarithms.xhtml')">
  Expand logarithms
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menusimplifycontractlogarithms.xhtml')">
  Contract logarithms
 </a>
</li>
<li>
 <a class="drop"
   href="javascript:popUp('menusimplifyfactorialsandgamma.xhtml')">
  Factorials and Gamma
 </a>
</li>
<li>
 <a class="drop" href=
   "javascript:popUp('menusimplifytrigsimplification.xhtml')">
  Trigonometric simplification
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menusimplifycomplexsimplification.xhtml')">
  Complex simplification
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menusimplifysubtitute.xhtml')">
  Substitute
 </a>
</li>
<li>
 <a class="drop"
    href="javascript:popUp('menusimplifyevaluatenounform.xhtml')">
  Evaluate noun form
 </a>
</li>
<li>
 <a class="drop" href=
    "javascript:popUp('menusimplifytogglealgebraicflag.xhtml')">
  Toggle algebraic flag
 </a>
</li>
<li>
 <a class="drop" href=
   "javascript:popUp('menusimplifyaddalgebraicequality.xhtml')">
  Add algebraic equality
```

```
            </a>
           </li>
           <li>
            <a class="drop"
               href="javascript:popUp('menusimplifymoduluscomputation.xhtml')">
             Modulus computation
            </a>
           </li>
          </ul>
         </li>
         <!-- End Simplify Menu -->
         <!-- Start Numeric Menu -->
         <li>
          <a class="toplevel" href="/">
           Numeric<!--span class="nabla">&nabla;</span-->
          </a>
          <ul>
           <li>
            <a class="drop"
              href="javascript:popUp('menunumerictogglenumericoutput.xhtml')">
             Toggle numeric output
            </a>
           </li>
           <li>
            <a class="drop"
               href="javascript:popUp('menunumerictofloat.xhtml')">
             To float
            </a>
           </li>
           <li>
            <a class="drop"
               href="javascript:popUp('menunumerictobigfloat.xhtml')">
             To bigfloat
            </a>
           </li>
           <li>
            <a class="drop"
               href="javascript:popUp('menunumericsetprecision.xhtml')">
             Set precision
            </a>
           </li>
          </ul>
         </li>
         <!-- End Simplify Menu -->
        </ul>
       </div>
      </td>
     </tr>
    </table>
   </form>
  </div>
<hr/>
<a href="javascript:makeone();">makeone</a>
  <form id="commreq">
```

```
    <p>
      Type an input command line to Axiom:<br/>
      <input type="text" id="p1"
       onclick="interpcall('p1');"
       value="integrate(sin(x),x)"
       size="80" />
    </p>
  </form>
  <center>
    <input type="button" value="Continue" name="continue"
      onclick="intercall('p1');"/>
  </center>
  <div id="mathAns"><div></div></div>
\getchunk{page foot}
```

complexlimit.xhtml

— complexlimit.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    var myform = document.getElementById("form2");
    var myfunct = myform.expr.value;
    var myvar = myform.vars.value;
    var ans = "";
    // decide what the limit point should be
    var finite = document.getElementById('finite').checked;
    if (finite == true) {
      var myreal = document.getElementById('fpreal').value;
      var mycomplex = document.getElementById('fpcomplex').value;
      if (myreal == 0) {
       if (mycomplex == 0) {
         ans = 'complexLimit('+myfunct+','+myvar+'=0)';
       } else {
         ans = 'complexLimit('+myfunct+','+myvar+'='+mycomplex+'*%i)';
       }
      } else {
       if (mycomplex == 0) {
         ans = 'complexLimit('+myfunct+','+myvar+'='+myreal+')';
       } else {
         ans =
          'complexLimit('+myfunct+','+myvar+'='+myreal+'+'+mycomplex+'*%i)';
       }
      }
    } else {
     ans = 'complexLimit('+myfunct+','+myvar+'=%infinity)';
    }
    return(ans);
   }
\getchunk{showfullanswer}
```

```
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <form id="form2">
   Enter the function you want to compute the limit of:<br/>
   <input type="text" id="expr" tabindex="10" size="50"
     value="sin(a*x)/tan(b*x)"/><br/>
   Enter the name of the variable:<br/>
   <input type="text" id="vars" tabindex="20" value="x"/><br/>
   <input type="radio" id="finite" tabindex="30" checked="checked"
     name="point"/>
    A finite point: Real part:
    <input type="text" id="fpreal" tabindex="40" value="0"/>
    Complex part:
    <input type="text" id="fpcomplex" tabindex="50" value="0"/><br/>
   <input type="radio" id="plus" tabindex="60" name="point"/>
    %infinity<br/>
  </form>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

conversionfunctions.xhtml

— conversionfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Conversion Functions</div>
  <hr/>
You can use conversion (see
<a href="axbook/section-9.27.xhtml#subsec-9.27.2">Jenks section 9.27.2</a>)
to go back and forth between
<a href="db.xhtml?Integer">Integer</a>,
<a href="db.xhtml?Fraction(Integer)">Fraction(Integer)</a> and
<a href="db.xhtml?Float">Float</a>, as appropriate.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="i:=3::Float" />
  <div id="ansp1"><div></div></div>
 </li>
```

```
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="i::Integer" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="i::Fraction Integer" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Since you are explicitly asking for a conversion, you must take
responsibility for any loss of exactness.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="r:=3/7::Float" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="r::Fraction Integer" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
This conversion cannot be performed: use
<a href="dboptruncate.xhtml">truncate</a> or
<a href="dboprounds.xhtml">round</a> if that is what you intend.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p4','p6']);"
    value="r::Integer" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The operations
<a href="dboptruncate.xhtml">truncate</a> and
<a href="dbopround.xhtml">round</a> truncate ...
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="truncate 3.6" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
and round to the nearest integral <a href="db.xhtml?Float">Float</a>
respectively.
<ul>
```

```
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="round 3.6" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="truncate(-3.6)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="round(-3.6)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
The operation <a href="dbopfractionpart.xhtml">fractionPart</a>
computes the fractional part of x, that is, x-truncate x.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="makeRequest('p11');"
    value="fractionPart 3.6" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
The operation <a href="dbopdigits.xhtml">digits</a> allows the user to
set the precision. It returns the previous value it was using.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="makeRequest('p12');"
    value="digits 40" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="sqrt 0.2" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="makeRequest('p14');"
    value="pi()$Float" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
The precision is only limited by the computer memory available.
Calculations at 500 or more digits of precision are not difficult.
<ul>
```

```
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="makeRequest('p15');"
    value="digits 500" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="makeRequest('p16');"
    value="pi()$Float" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Reset <a href="dbopdigits.xhtml">digits</a> to its default value.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="makeRequest('p17');"
    value="digits 20" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
Numbers of type <a href="db.xhtml?Float">Float</a> are represented as a record
of two integers, namely, the mantissa and the exponent where the base
of the exponent is binary. That is, the floating point number of the
binary. That is, the floating point number (m,e) represents the number
m*2**e. A consequence of using a binary base is that decimal numbers
can not, in general, be represented exactly.
\getchunk{page foot}
```

crytopage.xhtml

— **cryptopage.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
</center>
<hr/>
<ol>
 <li> <a href="cryptoclass1.xhtml">
       Laboratory Class 1: Introduction to Axiom
      </a>
 </li>
 <li> <a href="cryptoclass2.xhtml">
       Laboratory Class 2: Strings and Values
      </a>
 </li>
```

```
 <li> <a href="cryptoclass3.xhtml">
       Laboratory Class 3: Number Theory
      </a>
 </li>
 <li> <a href="cryptoclass4.xhtml">
       Laboratory Class 4: Simple Cryptosystems
      </a>
 </li>
 <li> <a href="cryptoclass5.xhtml">
       Laboratory Class 5: RSA and public-key cryptosystems
      </a>
 </li>
 <li> <a href="cryptoclass6.xhtml">
       Laboratory Class 6: Digital Signatures
      </a>
 </li>
 <li> <a href="cryptoclass7.xhtml">
       Laboratory Class 7: Knapsack cryptosystems
      </a>
 </li>
 <li> <a href="cryptoclass8.xhtml">
       Laboratory Class 8: Modes of Encryption
      </a>
 </li>
 <li> <a href="cryptoclass9.xhtml">
       Laboratory Class 9: Hash Functions
      </a>
 </li>
</ol>
\getchunk{page foot}
```

crytoclass1.xhtml

— cryptoclass1.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
  <h2>RCM3720 Cryptography, Network and Computer Security</h2>
  <h3>Laboratory Class 1: Introduction to Axiom</h3>
</center>
<hr/>

<b>Numbers and arithmetic</b>

<ul>
 <li> You can treat Axiom like a glorified calculator.  Enter the following:
  <ul>
   <li> <span class="cmd">3+5</span></li>
```

```
   <li> <span class="cmd">5*7</span></li>
   <li> <span class="cmd">2^3/3^5</span></li>
   <li> <span class="cmd">(3^4)^5</span></li>
   <li> <span class="cmd">3^(4^5)</span></li>
  </ul>
 </li>
 <li> What happens if you enter the last command without the brackets?</li>

 <li> To obtain the factorial <tt>n!</tt>, use the Axiom command
      <tt>factorial</tt>:
  <ul>
   <li> <span class="cmd">factorial(10)</span></li>
  </ul>
 </li>
 <li> By trial and error, find the smallest number whose factorial
      ends in six zeros.
 </li>
</ul>

<b>Lists</b>

<ul>
 <li> Assignment is done using "<tt>:=</tt>"
      where the <i>colon-equals</i> symbols are
      used for assigning a particular object to a variable.
  <ul>
   <li> <span class="cmd">var:=3</span></li>
  </ul>
 </li>
 <li> Lists are created using square brackets;
  <ul>
   <li> <span class="cmd">mylist1:=[k^2 for k in 1..10]</span></li>
  </ul>
 </li>
 <li> We can operate on all elements of a list using the
      <tt>reduce</tt> command:
  <ul>
   <li> <span class="cmd">reduce(+,mylist1)</span></li>
   <li> <span class="cmd">reduce(*,mylist2)</span></li>
  </ul>
 </li>
 <li> Of course, these could be done as single commands:
  <ul>
   <li> <span class="cmd">reduce(+,[k^2 for k in 1..10])</span></li>
   <li> <span class="cmd">reduce(*,[1/j for j in 5..15])</span></li>
  </ul>
 </li>
 <li> Notice how the last result is given as a single large fraction.  To
      obtain a decimal result we can do either of two things:
  <ol>
   <li> Convert the output to be of type ``Float'':
    <ul>
     <li> <span class="cmd">reduce(*,[1/j for j in 5..15])::Float</span></li>
     <li> Two colons can be used to change the type of an object.</li>
```

```
    </ul>
   </li>
   <li> Use floats in the initial command:
    <ul>
     <li> <span class="cmd">reduce(*,[1.0/j for j in 5..15])</span></li>
    </ul>
   </li>
  </ol>
 </li>
 <li> Using lists, add up the first 1000 integers.</li>

 <li> By trial and error, find the smallest number <i>n</i> for which
      the sum of the first <i>n</i> reciprocals is bigger than 8.
 </li>
 <li> We can also add numbers by using the <tt>sum</tt> function; here's how
      to add the first 100 reciprocals:
  <ul>
   <li> <span class="cmd">sum(1.0/k,k = 1..100)</span></li>
  </ul>
 </li>
</ul>

<b>Functions and maps</b>

<ul>
 <li> We shall create a simple function, and apply it to <tt>mylist1</tt> from
      above:
  <ul>
   <li> <span class="cmd">f(x) == x-2</span></li>
   <li> <span class="cmd">map(f,mylist1)</span></li>
  </ul>
 </li>
 <li> Supposing we want to subtract 2 from every element of a list without
      having to create a function first.  In this case we can use the
      "mapping" symbols:
  <ul>
   <li> <span class="cmd">map(x +-> x-2,mylist1)</span></li>
  </ul>
 </li>
 <li> Create a list called <tt>nums</tt> containing all the integers from 1
      to 100.  Now we shall create a simple function <tt>f(x)</tt> which
      returns <tt>x</tt> if it is prime, and 0 otherwise.  The Axiom
      function <tt>prime?</tt> tests for primality:
  <ul>
   <li> <span class="cmd">f(x)==if prime?(x) then x else 0</span></li>
  </ul>
 </li>
 <li> Now apply this function <tt>f</tt> to <tt>nums</tt>.
      Remove all the zeros:
      <i>(% refers to the output of the last command.)</i>
  <ul>
   <li> <span class="cmd">remove(0,%)</span></li>
  </ul>
 </li>
```

```
 <li> and determine how many primes there are, using the hash symbol #
      which can be used to count the number of elements in a list:
  <ul>
   <li> <span class="cmd">#%</span></li>
  </ul>
 </li>
 <li> These last commands can be done as a single command:
  <ul>
   <li> <span class="cmd">#remove(0,map(f,nums))</span></li>
  </ul>
 </li>
 <li> Use the last command to create a function called <tt>numprimes</tt>
      which will count the number of primes below any given integer.
 </li>
 <li> How many primes are there less than 1000?  Less than 10000?</li>

 <li> Alternatively, we can list all the primes below 100 by creating our
      list using the "such that" operator---a vertical stroke:
  <ul>
   <li> <span class="cmd">[k for k in 1..100 | prime?(k)]</span></li>
  </ul>
 </li>
 <li> or we could just return the length of the list:
  <ul>
   <li> <span class="cmd">#[k for k in 1..100 | prime?(k)]</span></li>
  </ul>
 </li>
 <li> Use this approach to create a function called <tt>numprimes2</tt>
      which will count the number of primes below any given integer.
 </li>
 <li>How many primes are there less than 2000?  Less than 15000?</li>
</ul>

<b>Housekeeping</b><br/>

Axiom contains many commands for managing your workspace and your
environment; such commands are all prefixed with a right parenthesis.

<ul>
 <li> Sometimes you need to clear a variable, say a variable <tt>x</tt>:
  <ul>
   <li> <span class="cmd">)clear properties x</span></li>
  </ul>
 </li>
 <li> Most commands of this sort can be abbreviated using their
      first two letters:
  <ul>
   <li> <span class="cmd">)cl pr x</span></li>
  </ul>
 </li>
 <li> To clean out everything:
  <ul>
   <li> <span class="cmd">)cl all</span></li>
  </ul>
```

```
 </li>
 <li> To see what variables you've accumulated over your work:
  <ul>
   <li> <span class="cmd">)display names</span></li>
   <li> <i>or abbreviated as</i> )d n</li>
  </ul>
 </li>
 <li> You may have noticed earlier that Axiom poured out lots
      of messages when it first "got going".  These can be turned off:
  <ul>
   <li> <span class="cmd">)set messages autoload off</span></li>
  </ul>
 </li>
 <li> Note here that if you just type in "<tt>)set</tt>" or its abbreviation
      "<tt>)se</tt>", you'll be presented with the list of all the possible
      options.  Likewise "<tt>)se me</tt>" lists all possible options for
      messages, and so on.
 </li>
 <li> Can you find the command which turns on a time function,
      so gives the time to compute each command?
 </li>
 <li> The command "<tt>)summary</tt> gives a quick summary of these
      commands.
 </li>
 <li> To quit Axiom, type
  <ul>
   <li> <span class="cmd">)quit</span></li>
  </ul>
 </li>
 <li> or its one letter abbreviation "<tt>)q</tt>", followed by <tt>y</tt> to
      confirm.
 </li>
</ul>
\getchunk{page foot}
```

crytoclass2.xhtml

— cryptoclass2.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 2: Strings and Values</h3>
</center>
<hr/>

<b>Characters and Strings</b>
```

```
<ul>
 <li> All printable characters have a fixed ASCII value; some of which are:
<br/>
<pre>
      Character |  A   B   Y   Z   a   b   y   z
    ------------+-------------------------------
    ASCII Value | 65  66  89  90  97  98  121 122
                |
      Character |  0   1   8   9   ,   -   .   /
    ------------+-------------------------------
    ASCII Value | 48  49  56  57  44  45  46  47
</pre>
 </li>
 <li> To obtain values 0 to 25 for A to Z, we need to subtract 65 from
      the ASCII values.
 </li>
 <li> In Axiom, the <tt>ord</tt> command gives the ASCII value of a
      character.  Create a string such as:
  <ul>
   <li> <span class="cmd">str:="THISISASTRING"</span></li>
  </ul>
 </li>
 <li> A string can be turned into a list of characters using <tt>members</tt>:
  <ul>
   <li> <span class="cmd">members(str)</span></li>
  </ul>
 </li>
 <li> This means a string can be turned into a list of ASCII values by
      mapping the <tt>ord</tt> function onto the list of members:
  <ul>
   <li> <span class="cmd">map(ord,members(str))</span></li>
  </ul>
 </li>
 <li> To obtain values in the 0--25 range, try using an unnamed function:
  <ul>
   <li> <span class="cmd">strn:=map(x +-> ord(x)-65,members(str))</span></li>
  </ul>
 </li>
 <li> Use this last command to create a function <tt>str2lst</tt> which will
      take a string (assumed to be of capital letters, with no spaces or
      punctuation), and return a list of values between 0 and 25.
 </li>
 <li> To go the other way, we first need to add 65 to all elements of
      <tt>strn</tt>:
  <ul>
   <li> <span class="cmd">map(x +-> x+65,strn)</span></li>
  </ul>
 </li>
 <li> Turn this into characters with <tt>char</tt>:
  <ul>
   <li> <span class="cmd">map(char,%)</span></li>
  </ul>
 </li>
 <li> These can be done as a single command:
```

```
 <ul>
  <li> <span class="cmd">map(x +-> char(x+65),strn)</span></li>
 </ul>
</li>
<li> To put them all together as a single string we can concatenate them
     with the <tt>concat</tt> function from the <tt>String</tt> domain:
 <ul>
  <li> <span class="cmd">concat(%)$String</span></li>
 </ul>
</li>
<li> In one line:
 <ul>
  <li> <span class="cmd">concat(map(x +-> char(x+65),strn))$String</span></li>
 </ul>
</li>
<li> Alternatively, we could convert the characters to type <tt>String</tt>
     before concatenation:
 <ul>
  <li>
   <span class="cmd">
    concat(map(x +-> char(x+65)::String,strn))
   </span>
  </li>
 </ul>
</li>
<li> Use either version of this last command to create a function
     <tt>lst2str</tt> which will take a list of values between 0 and 25 and
     return a string.
</li>
<li> Create a text file in one of your private directories called
     <tt>my3720.input</tt> and copy your <tt>str2lst</tt> and
     <tt>lst2str</tt> functions to it.
</li>
<li> You can read command line input from a file with the extension
     <tt>.input</tt> using the <tt>)read</tt> command:
 <ul>
  <li> <span class="cmd">)read my3720</span></li>
 </ul>
</li>
<li> The Caesar cipher can be implemented by the following three steps:
 <ol>
  <li> Turn the string into a list,</li>
  <li> Add 3 to every number in the list,</li>
  <li> Turn this new list back into a string.</li>
 </ol>
</li>
<li> To ensure that step (2) remains in the 0--25 range, we need to use the
     <tt>rem</tt> function.  These can all be put together as:
 <ul>
  <li>
   <span class="cmd">
    caesar(str) == lst2str(map(x +-> (x+3) rem 26, str2lst(str)))
   </span>
  </li>
```

```
  </ul>
 </li>
 <li> Try this out on a few strings of your choice.</li>

 <li> By replacing the "<tt>+3</tt>" in the <tt>caesar</tt> function with
  "<tt>+n</tt>" create a new function called <tt>trans(str,n)</tt> which
  implements a general translation cipher.
 </li>
 <li> Test it out; these two commands should produce the same results.
  <ul>
   <li> <span class="cmd">caesar("MYSTRING")</span></li>
   <li> <span class="cmd">trans("MYSTRING",3)</span></li>
  </ul>
 </li>
 <li> If you like, add the <tt>caesar</tt> and <tt>trans</tt> functions to
      your <tt>my3720.input</tt> file.
 </li>
 <li> Test your <tt>trans</tt> function out on a few other strings and
      translation values.
 </li>
 <li> The <tt>ROT13</tt> cipher is used in Usenet postings to hide information
      which might be considered offensive.  It is a translation cipher with a
      shift of 13.  Since 13 is half of 26, this means that encrytion and
      decryption are exactly the same.  Apply <tt>ROT13</tt> to:
  <ul>
   <li> GUVFVFNIRELFREVBHFOHFVARFF</li>
  </ul>
 </li>
 <li> Consider this string which has been produced with a translation cipher.
      To decrypt it, simply apply all possible shifts until you obtain
      understandable text.
  <ul>
   <li> IUDTCUQBBOEKHCEDUO</li>
  </ul>
 </li>
 <li> To apply all the possible shifts do:
  <ol>
   <li> <span class="cmd">ct:="IUDTCUQBBOEKHCEDUO"</span></li>
   <li> <span class="cmd">for i in 1..26 repeat output trans(ct,i)</span></li>
  </ol>
 </li>
 <li> What is the plaintext?</li>
</ul>
\getchunk{page foot}
```

———-

crytoclass3.xhtml

— cryptoclass3.xhtml —

```
\getchunk{standard head}
 </head>
```

```
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 3: Number Theory</h3>
</center>
<hr/>

<ul>

 <li> Check out the commands <tt>gcd</tt> and <tt>factor</tt>, and test them
  on different numbers, small and large.
 </li>
 <li> Axiom provides a few useful commands for taking apart the factors of an
  object:
  <ul>
   <li> <span class="cmd">n:=5040</span></li>
   <li> <span class="cmd">f:=factor(n)</span></li>
   <li> <span class="cmd">numf:=numberOfFactors(f)</span></li>
   <li> <span class="cmd">fs:=[nthFactor(f,i) for i in 1..numf]</span></li>
   <li> <span class="cmd">es:=[nthExponent(f,i) for i in 1..numf]</span></li>
   <li> <span class="cmd">reduce(*,[fs.i^es.i for i in 1..numf])</span></li>
  </ul>
 </li>
 <li> The last command simply multiplies all the factors to their powers.</li>

 <li> Check out the commands <tt>prime?</tt>, <tt>nextPrime</tt> and
      <tt>prevPrime</tt>.
 </li>
 <li> To compute the <tt>i</tt>-th prime, we can construct a <i>stream</i>
      (an infinite list) in Axiom:
  <ul>
   <li>
    <span class="cmd">
     primes:Stream Integer:=[i for i in 2.. | prime? i]
    </span>
   </li>
  </ul>
 </li>
 <li> Now we can find, for example, the 100-th prime, and the 2500-th prime:
  <ul>
   <li> <span class="cmd">primes.100</span></li>
   <li> <span class="cmd">primes.2500</span></li>
  </ul>
 </li>
 <li> Create random 10 digit primes:
  <ul>
   <li> <span class="cmd">p := nextPrime(random(10^10))</span></li>
   <li> <span class="cmd">q := nextPrime(random(10^10))</span></li>
  </ul>
 </li>
 <li> Now multiply them and factor the product.  How long did it take?</li>

 <li> Try the same thing with 12 digit primes and 15 digit primes.</li>
```

```
<li> The extended Euclidean algorithm is implemented by the command
 <tt>extendedEuclidean</tt>.  Here's how to use it:
 <ul>
  <li> <span class="cmd">a:=1149</span></li>
  <li> <span class="cmd">b:=3137</span></li>
  <li> <span class="cmd">g:=extendedEuclidean(a,b)</span></li>
  <li> <span class="cmd">s:=g.coef1</span></li>
  <li> <span class="cmd">t:=g.coef2</span></li>
 </ul>
</li>
<li> and now test them:
 <ul>
  <li> <span class="cmd">s*a+t*b</span></li>
 </ul>
</li>
<li> Try this on a few other numbers.</li>

<li> Axiom uses the command <tt>positiveRemainder</tt> instead of
     <tt>mod</tt> command, so let's define <tt>mod</tt> to be a renaming
     of the <tt>positiveRemainder</tt> function:
 <ul>
  <li> <span class="cmd">mod ==> positiveRemainder</span></li>
 </ul>
</li>
<li> Now the commands <tt>addmod</tt>, <tt>submod</tt>, <tt>mulmod</tt>, and
     <tt>invmod</tt> can be used to perform modular arithmetic.  Here's a few
     examples; first a simple modulus calculation:
 <ul>
  <li> <span class="cmd">-10 mod 3</span></li>
 </ul>
</li>
<li> Addition, subtraction and multiplication mod 14:
 <ul>
  <li> <span class="cmd">addmod(10,13,14)</span></li>
  <li> <span class="cmd">submod(17,23,14)</span></li>
  <li> <span class="cmd">mulmod(13,27,14)</span></li>
 </ul>
</li>
<li> Powers and inverses:
 <ul>
  <li> <span class="cmd">powmod(19,237,14)</span></li>
  <li> <span class="cmd">invmod(11,14)</span></li>
 </ul>
</li>
<li> Find out what happens if you try to take an inverse of a number not
     relatively prime to the modulus:
 <ul>
  <li> <span class="cmd">invmod(12,14)</span></li>
 </ul>
</li>
<li> Try these command with a few other numbers, and test out the examples in
 the notes.
</li>
```

```
 <li> The second method, which can be more powerful, is to treat all numbers
  as elements of the residue values 0 to <tt>n-1</tt>.  This can be done with
  the <tt>IntegerMod</tt> construction, or its abbreviation <tt>ZMOD</tt>.
  Here's a few examples:
  <ul>
   <li> <span class="cmd">a:=11::ZMOD 14</span></li>
  </ul>
 </li>
 <li> This declares the variable <tt>a</tt> to be a member of the residue
      class modulo 14.  Now all arithmetic including <tt>a</tt> will be
      reduced to this same class of values:
  <ul>
   <li> <span class="cmd">a+25</span></li>
   <li> <span class="cmd">a*39</span></li>
   <li> <span class="cmd">a^537</span></li>
  </ul>
 </li>
 <li> Inversion can be done with the <tt>recip</tt> command:
  <ul>
   <li> <span class="cmd">recip(a)</span></li>
  </ul>
 </li>
 <li> We don't have to define a variable first.  All the above commands could
      be equivalently written as:
  <ul>
   <li> <span class="cmd">(11::ZMOD 14)+25</span></li>
   <li> <span class="cmd">11::ZMOD 14*39</span></li>
   <li> <span class="cmd">11::ZMOD 14^537</span></li>
   <li> <span class="cmd">recip(11::ZMOD 14)</span></li>
  </ul>
 </li>
 <li> If the modulus is a prime, then division (by non-zero values) is also
      possible.  Axiom provides the alternative construction
      <tt>PrimeField</tt> or more simply <tt>PF</tt>.  For example:
  <ul>
   <li> <span class="cmd">a:=7::PF 11</span></li>
  </ul>
 </li>
 <li> All the above arithmetic operations of addition, subtraction,
      multiplication and powers work, but now we also have inversion:
  <ul>
   <li> <span class="cmd">1/a</span></li>
  </ul>
 </li>
 <li> Using any of the methods you like, test out Fermat's theorem for a large
      prime <tt>p</tt> and an integer <tt>a</tt>.
 </li>
 <li> Euler's totient function is implemented with <tt>eulerPhi</tt>.  Choose
      a large integer <tt>n</tt>, a random <tt>a</tt> with
      <tt>gcd(a,n)=1</tt> , and test Euler's theorem
 </li>
</ul>
\getchunk{page foot}
```

crytoclass4.xhtml

— cryptoclass4.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 4: Simple Cryptosystems</h3>
</center>
<hr/>

We have experimented with the Caesar cipher and the more general
translation cipher.  We shall start looking at the Vigen&#x0E8;re cipher.
The trick is to add the correct letter of the code to the letter of
the key:
<pre>
 Index of plain text i: 1  2  3  4  5  6  7  8
             Plaintext: W  I  T  H  D  R  A  W
                   Key: C  O  D  E  C  O  D  E
        Index of key j: 1  2  3  4  1  2  3  4
</pre>

The indices of the key repeat 1, 2, 3, 4.  We can get a repetition of
length four by using a modulus of 4:
<pre>
                     i: 1  2  3  4  5  6  7  8
             i (mod 4): 1  2  3  0  1  2  3  0
</pre>

What we need to do is to subtract one before the modulus, and add one after:
<pre>
                     i: 1  2  3  4  5  6  7  8
                   i-1: 0  1  2  3  4  5  6  7
           i-1 (mod 4): 0  1  2  3  0  1  2  3
       i-1 (mod 4) + 1: 1  2  3  4  1  2  3  4
</pre>

This means that in the Vigen&#x0E8;re cipher, we add the <i>i</i>-th
character of the plaintext, and the <i>j</i>-th character of the key, where
<pre>
   j=i-1 (mod n) + 1
</pre>
with <i>n</i> being the length of the key.

<ul>

 <li> First read in the <tt>rcm3720.input</tt> file you have created:
  <ul>
```

```
  <li> <span class="cmd">)read rcm3720</span></li>
 </ul>

 You may have to include the full path here.
</li>
<li> Enter a plaintext
 <ul>
  <li> <span class="cmd">plaintext:="WITHDRAWONEHUNDREDDOLLARS"</span></li>
 </ul>
</li>
<li> and a keyword:
 <ul>
  <li> <span class="cmd">key := "CODE"</span></li>
 </ul>
</li>
<li> Now we can obtain the lengths of the plaintext and key with the hash
     symbol:
 <ul>
  <li> <span class="cmd">pn:=#plaintext</span></li>
  <li> <span class="cmd">kn:=#key</span></li>
 </ul>
</li>
<li> Turn both plaintext and key into lists of numbers:
 <ul>
  <li> <span class="cmd">pl:=str2lst(plaintext)</span></li>
  <li> <span class="cmd">kl:=str2lst(key)</span></li>
 </ul>
</li>
<li> Now we can add them using the formula for <tt>j</tt> above to obtain
     the list corresponding to the ciphertext:
 <ul>
  <li>
   <span class="cmd">
    cl:=[(pl.i+kl.((i-1) rem kn+1))::ZMOD 26 for i in 1..pn]
   </span>
  </li>
 </ul>
</li>
<li> And obtain the ciphertext (we need to convert our list to a list of
     integers first):
 <ul>
  <li> <span class="cmd">ciphertext:=lst2str(cl::List INT)</span></li>
 </ul>
</li>
<li> Try a few other Vigen&#x0E8;re encryptions.</li>

<li> To decrypt, we just <i>subtract</i> the key value from the ciphertext
 value:
 <ul>
  <li>
   <span class="cmd">
    pl:=[(cl.i+kl.((i-1) rem kn+1))::ZMOD 26 for i in 1..pn]
   </span>
   </li>
```

```
  <li> <span class="cmd">lst2str(pl::List INT)</span></li>
 </ul>
</li>
<li> Now for the Hill (matrix) cipher.  We shall use <tt>3 x 3</tt>
 matrices, so first create a plaintext whose length is a multiple of 3:
 <ul>
  <li> <span class="cmd">plaintext:="WITHDRAWONEHUNDREDDOLLARSXX"</span></li>
  <li> <span class="cmd">pl:=str2lst(plaintext)</span></li>
  <li> <span class="cmd">r:=3</span></li>
  <li> <span class="cmd">c:INT:=#pl/r</span></li>
 </ul>
</li>
<li> The values <tt>r</tt> and <tt>c</tt> are the row and column numbers
     of the plaintext matrix.
</li>
<li> Now put all the plaintext values into a <tt>r x c</tt> matrix:
 <ul>
  <li>
   <span class="cmd">
    S:=matrix([[pl.(r*(i-1)+j) for i in 1..c] for j in 1..r])
   </span>
  </li>
 </ul>
</li>
<li> Create the key matrix:
 <ul>
  <li>
   <span class="cmd">
    M:Matrix ZMOD 26:=matrix([[22,11,19],[15,20,24],[25,21,16]])
   </span>
  </li>
 </ul>
</li>
<li> Multiply the two matrices:
 <ul>
  <li> <span class="cmd">C:=M*S</span></li>
 </ul>
</li>
<li> Notice how the results are automatically reduced modulo 26,
     because that is how the matrix <tt>M</tt> was defined.
</li>
<li> Now we have to read off the elements of <tt>C</tt> into a single list;
 this can be done by transposing the matrix, and reading off the rows as
 lists:
 <ul>
  <li>
   <span class="cmd">
    CL:=concat(transpose(C)::List List ZMOD 26)
   </span>
  </li>
 </ul>
</li>
<li> And finally turn into ciphertext:
 <ul>
```

```
   <li> <span class="cmd">lst2str(CL::List INT)</span></li>
  </ul>
 </li>
 <li> Finally, here's how we can invert our matrix <tt>M</tt> modulo 26:
  <ul>
   <li> <span class="cmd">adj:=adjoint(M).adjMat</span></li>
   <li> <span class="cmd">invdet:=recip(determinant(M))</span></li>
   <li> <span class="cmd">MI:=invdet*adj</span></li>
  </ul>
 </li>
 <li> Or alternatively, as one command:
  <ul>
   <li>
    <span class="cmd">
     MI:=recip(determinant(M))*adjoint(M).adjMat
    </span>
   </li>
  </ul>
 </li>
 <li> Check the result:
  <ul>
   <li> <span class="cmd">M*MI</span></li>
  </ul>
 </li>
</ul>
\getchunk{page foot}
```

crytoclass5.xhtml

— cryptoclass5.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 5: RSA and public-key cryptosystems</h3>
</center>
<hr/>
<ul>
 <li> Read in this file:
  <ul>
   <li>
    <span class="cmd">
     )read "S:/Samples/RCM3720/rcm3720.input" )quiet
    </span>
   </li>
  </ul>
 </li>
 <li> You can leave the "<tt>)quiet</tt>" off if you like.  The file
```

```
 is also available <a href="rcm3720.input">here</a>.
 If you obtain it from the
 website, save it to a place of your choice, and <tt>read</tt> it
 into your Axiom session using the full path, as shown above.
</li>
<li> Now create some large primes and their product:
 <ul>
  <li> <span class="cmd">r() == rand(2^100)</span></li>
  <li> <span class="cmd">p:=nextPrime(r())</span></li>
  <li> <span class="cmd">q:=nextPrime(r())</span></li>
  <li> <span class="cmd">n:=p*q</span></li>
 </ul>
</li>
<li> Choose a value <tt>e</tt> and ensure that it is relatively prime
     to your <tt>(p-1)(q-1)</tt>, and determine
     <tt>d=e^-1 mod (p-1)(q-1)</tt>.  (Use the <tt>invmod</tt> function here).
</li>
<li> Create a plaintext:
 <ul>
  <li> <span class="cmd">pl:="This is my plaintext."</span></li>
 </ul>
</li>
<li> (or any plaintext you like), and convert it to a number using the
     <tt>str2num</tt> procedure from the file above:
 <ul>
   <li> <span class="cmd">pln:=str2num(pl)</span></li>
 </ul>
</li>
<li> Encrypt this number using the RSA method:
 <ul>
  <li> <span class="cmd">ct:=powmod(pln,e,n)</span></li>
 </ul>
</li>
<li> and decrypt the result:
 <ul>
  <li> <span class="cmd">decrypt:=powmod(ct,d,n)</span></li>
  <li> <span class="cmd">num2str(decrypt)</span></li>
 </ul>
</li>
<li> With a friend, swap your public keys and use them to send
     each other a ciphertext encrypted with your friend's public key.
</li>
<li> Now decrypt the ciphertext you have received using your private key.</li>

<li> Now try Rabin: create two large primes <tt>p</tt> and <tt>q</tt> and
     ensure that each is equal to 3 mod 4.  (You might have to run the
     <tt>nextPrime</tt> command a few times until you get primes which work.)
</li>
<li> Create <tt>N=pq</tt> and create a plaintext <tt>pl</tt>, and its
     numerical equivalent.
</li>
<li> Determine the ciphertext <tt>c</tt> by squaring your
     number mod <tt>N</tt>.
</li>
```

```
<li> Determine the <tt>s</tt> and <tt>t</tt> for which <tt>sp+tq=1</tt>
     by using the <tt>extendedEuclidean</tt> function.
</li>
<li> Now follow through the Rabin decryption:
 <ul>
  <li> <span class="cmd">cp:=powmod(c,(p+1)/4,N) </span></li>
  <li> <span class="cmd">cq:=powmod(c,(q+1)/4,N)</span></li>
  <li>
   <span class="cmd">
    c1:=(s*p*cq+t*q*cp)::ZMOD N,num2str(c1::INT)
   </span>
  </li>
  <li>
   <span class="cmd">
    c2:=(s*p*cq-t*q*cp)::ZMOD N,num2str(c2::INT)
   </span>
  </li>
  <li>
   <span class="cmd">
    c3:=(-s*p*cq-t*q*cp)::ZMOD N,num2str(c3::INT)
   </span>
  </li>
  <li>
   <span class="cmd">
    c4:=(-s*p*cq+t*q*cp)::ZMOD N,num2str(c4::INT)
   </span>
  </li>
 </ul>
</li>
<li> One of the outputs <tt>c1</tt>, <tt>c2</tt>, <tt>c3</tt> and
     <tt>c4</tt> should produce the correct plaintext; the others should be
     gibberish.
</li>
<li> As above, swap public keys with a friend, and use those public
     keys to encrypt a message to him or her.  Now decrypt the ciphertext
     you have been given.
</li>
<li> For the el Gamal system, you need a large prime and a primitive
     root. Create a large prime <tt>p</tt> and find a primitive root
     <tt>a</tt> using.
 <ul>
  <li> <span class="cmd">a:=primitiveElement()$PF p</span></li>
 </ul>
</li>
<li> The <tt>primitiveElement</tt> command is not very efficient, so
     if it seems to be taking a long time, abort the computation and try
     with another prime.
</li>
<li> Do this in pairs with a friend, so that you each agree on a
     large prime and a primitive root.
</li>
<li> Now choose a random value <tt>A</tt>:
 <ul>
  <li> <span class="cmd">A:=random(p-1)</span></li>
```

```
  </ul>
 </li>
 <li> and create your public key <tt>A1=a^A (mod p)</tt>:
  <ul>
   <li> <span class="cmd">A1:=a^A</span></li>
  </ul>
 </li>
 <li> Swap public keys with your friend.</li>

 <li> Create a plaintext <tt>pl</tt> and its number <tt>pln</tt>, and create
      the ciphertext as follows (where <tt>A1</tt> is your friend's
      public key):
  <ul>
   <li> <span class="cmd">k:=random(p-1)</span></li>
   <li> <span class="cmd">K:=A1^k</span></li>
   <li> <span class="cmd">C:=[a^k, K*pln]</span></li>
  </ul>
 </li>
 <li> This pair <tt>C</tt> is the ciphertext you send to your friend.</li>

 <li> Now decrypt the ciphertext you have been sent:
  <ul>
   <li> <span class="cmd">K:=C.1 ^ A</span></li>
   <li> <span class="cmd">m:=C.2/K</span></li>
   <li> <span class="cmd">num2str(m::INT)</span></li>
  </ul>
 </li>
</ul>
\getchunk{page foot}
```

———-

crytoclass6.xhtml

— cryptoclass6.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 6: Digital Signatures</h3>
</center>
<hr/>

You will need to read in the <a href="rcm3720.input">rcm3720.input</a>
file for the <tt>str2num</tt> and <tt>num2str</tt> procedures.
<br/>
<b>NOTE:</b> To save typing in all the messages and long signature
numbers, just copy them from <a href="signatures.txt">signatures.txt</a>

<ul>
```

```
 <li> For an RSA signature scheme, I provide the public key <i>(n,e)</i>, where
<pre>
      137
   n=2   -1,  e=17
</pre>
 </li>
 <li> This value <tt>n</tt> has two large prime factors.
      Use my public key to verify my signature of the following message:
<pre>
   This is my text.
   687670274656715771910731284950827957007 68
</pre>
 </li>
 <li> Now try with the public key
<pre>
      67
  n=(6   - 1)/5,  e=17
</pre>
 </li>
 <li> to verify my signature:
<pre>
   Please feed my dog!
   1703215098456351993605104919259566435843590978852633
</pre>
 </li>
 <li> For a Rabin signature scheme, I provide the public key
<pre>
       74
   N=(7  -1)/6,
</pre>
  which I know can be factorized into two large primes.
 </li>
 <li> Check the following message and signature:
<pre>
   Arrive Thursday.
   1894797231225344140197834472714118955 09
</pre>
 </li>

 <li> For an El Gamal signature scheme, I choose the next prime after
<pre>
     150
    2
</pre>
  which has a primitive root <tt>a=2</tt>.  My public key is
<pre>
    B=1369851585774063312693119161120024351761244461
</pre>
 </li>
 <li> Verify the signature
<pre>
    Leave AT ONCE!,
    13890805253057543921119767153610694253535781 98
```

```
    114132646807016822998297613380172143030600 4477
</pre>
 </li>
 <li> For a DSS signature, choose <tt>p</tt> to be the next prime after
<pre>
     170
    2     and q=1434415054685906962 09
</pre>
 </li>
 <li> Verify that <tt>q</tt> is a divisor of <tt>p-1</tt>.

  A primitive root of <tt>p</tt> is <tt>a=3</tt>.
  Use this primitive root to determine
<pre>
         (p-1)/q
    g = a        mod p
</pre>
 </li>
 <li> The public key value is
<pre>
    B=1394256880659595564848116770226045673904445792389839.
</pre>
 </li>
 <li> Now using these values, verify this signature:
<pre>
    Now's your chance!
    646092094646383558 01
    138248087412004933 30
</pre>
 </li>
 <li> Now exchange some public keys with a friend, and sign messages to each
      other. Then verify the signatures you have been sent. Make sure you try
      each of
  <ul>
   <li> RSA signatures,</li>
   <li> Rabin signatures,</li>
   <li> El Gamal signatures,</li>
   <li> DSS.</li>
  </ul>
 </li>
</ul>
\getchunk{page foot}
```

crytoclass7.xhtml

— cryptoclass7.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
```

```
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 7: Knapsack cryptosystems</h3>
</center>
<hr/>

You will need to read in the <a href="rcm3720.input">rcm3720.input</a>
file for various necessary procedures.
<br/><br/>
<b>The subset sum problem</b>
<br/><br/>

We will first experiment with this problem; creating random lists and adding
up elements from them.

<ul>
 <li> Start with a list of eight elements:
  <ul>
   <li> <span class="cmd">ln:=8</span></li>
   <li> <span class="cmd">lst:=[random(10^6) for i in 1..ln]</span></li>
   <li> <span class="cmd">m:=[random(2) for i in 1..ln]</span></li>
   <li> <span class="cmd">c:=reduce(+,[m.i*lst.i for i in 1..ln])</span></li>
   <li> <span class="cmd">subsetsum(lst,c)</span></li>
  </ul>
 </li>
 <li> The <tt>subsetsum</tt> command implements a fairly non-efficient
      command for attemping to solve the subset sum problem for an
      arbitrary list.
 </li>
 <li> Try the above commands, but starting with a length <tt>ln</tt> of
      12. You should find the command is a bit slower this time.
      Use this command to time it:
  <ul>
   <li> <span class="cmd">)set messages time on</span></li>
  </ul>
 </li>
 <li> Experiment with lengths of 16 and 20.  How long does the
      <tt>subsetsum</tt> command take for each of these values?
 </li>
</ul>
<br/><br/>
<b>Superincreasing sequences</b>

<ul>
 <li> Create a superincreasing sequence with
  <ul>
   <li> <span class="cmd">ln:=8</span></li>
   <li> <span class="cmd">lst:=[random(10^6) for i in 1..ln]</span></li>
   <li>
    <span class="cmd">
     for i in 2..ln repeat lst.i:=reduce(+,[lst.j for j in 1..i-1])+random(10)+1
    </span>
   </li>
  </ul>
```

```
 </li>
 <li> Now create <tt>m</tt> and <tt>c</tt> as above.  This time, solve the
      problem with
  <ul>
   <li> <span class="cmd">siSolve(lst,c)</span></li>
  </ul>
 </li>
 <li> Now try with larger lengths: 12, 16 and 20, and time the commands each
      time.
 </li>
 <li> What can you say about solving the subset sum problem for general and
      superincreasing lists?
 </li>
</ul>
<br/><br/>
<b>The Merkle-Hellman additive knapsack system</b>

<ul>
 <li> Create a superincreasing list of length <tt>ln</tt> 10, and call it
  <tt>a</tt>.  Create a new number <tt>N</tt> greater than the sum of all
  values of <tt>a</tt>.  Check with
  <ul>
   <li> <span class="cmd">N>reduce(+,[a.i for i in 1..ln])</span></li>
  </ul>
 </li>
 <li> Now choose (randomly) a value <b>wN</b> and which is
      relatively prime to <b>N</b>.  Then construct your public key:
  <ul>
   <li> <span class="cmd">b:=map(x +-> x*w rem N,a)</span></li>
  </ul>
 </li>
 <li> Now for an encryption and decryption. Create a random message <tt>m</tt>
  as above, and encrypt it to a ciphertext <tt>c</tt> using the public key
  <tt>b</tt>.
 </li>
 <li> Decrypt it as follows:
  <ul>
   <li> <span class="cmd">c1:=inv_mod(w,N)*c rem N</span></li>
   <li> <span class="cmd">siSolve(a,c1)</span></li>
  </ul>
 </li>
 <li>
  Experiment with longer lists and messages: 12, 16, 20 or even larger.
 </li>
</ul>
<br/><br/>
<b>The Merkle-Hellman multiplicative knapsack system</b>

<ul>
 <li> Choose <tt>a</tt> to be the first ten primes,
      and a large prime <tt>p</tt>:
  <ul>
   <li> <span class="cmd">a:=[2,3,5,7,11,13,17,19,23,29]</span></li>
   <li> <span class="cmd">p:=6469785001</span></li>
```

```
  </ul>
 </li>
 <li> Check that <tt>p</tt> is greater than the product of all elements of
      <tt>a</tt>:
  <ul>
   <li> <span class="cmd">p>reduce(*,[a.i for i in 1..10])</span></li>
  </ul>
 </li>
 <li> and that <tt>p-1</tt> has only small factors:
  <ul>
   <li> <span class="cmd">factor(p-1)</span></li>
  </ul>
 </li>
 <li> Choose as a primitive root the value 34:
  <ul>
   <li> <span class="cmd">r:=34</span></li>
   <li> <span class="cmd">primitive?(r)$PF(p)</span></li>
  </ul>
 </li>
 <li> and compute the public key:
  <ul>
   <li> <span class="cmd">b:=map(x +-> discreteLog(r,x)$PF(p),a)</span></li>
  </ul>
 </li>
 <li> Create a message of length 10, and encrypt it using the public key
  <tt>b</tt>:
  <ul>
   <li>
    <span class="cmd">
     c:=reduce(+,[m.i*b.i::INT for i in 1..ln])
    </span>
   </li>
  </ul>
 </li>
 <li> Decryption is now done with:
  <ul>
   <li> <span class="cmd">c1:=powmod(r,c,p)</span></li>
   <li> <span class="cmd">factor(c1)</span></li>
  </ul>
 </li>
</ul>
\getchunk{page foot}
```

crytoclass8.xhtml

— cryptoclass8.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
```

```
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 8: Modes of Encryption</h3>
</center>
<hr/>

We will investigate the different modes of encryption using the Hill
(matrix) cryptosystem.  Start off by entering some matrices:
  <ul>
   <li>
    <span class="cmd">
     M:=matrix([[15,9,21],[2,10,7],[16,11,12]])::Matrix ZMOD 26
    </span>
   </li>
   <li>
    <span class="cmd">
     MI:=matrix([[7,17,19],[24,0,23],[12,25,10]])::Matrix ZMOD 26
    </span>
   </li>
  </ul>

Check that you've entered everything correctly with
  <ul>
   <li> <span class="cmd">M*MI</span></li>
  </ul>

Note that because the matrices were defined in terms of numbers mod 26,
their product is automatically reduced mod 26.

Now enter the following column vector:
  <ul>
   <li>
    <span class="cmd">
     zero31:=matrix([[0],[0],[0]])::Matrix ZMOD 26
    </span>
   </li>
  </ul>

For this lab, rather than fiddling about with translations between
letters and numbers, all our work will be done with numbers alone
(in the range 0..25).

<br/><br/>
<b>ECB</b>
<br/><br/>
For electronic codebook mode, encryption is performed by multiplying each
plaintext block by the matrix, and decryption by multiplying each ciphertext
block by the inverse matrix:
<pre>
                  -1
    C =M.P ,  P =M  C
     i    i    i     i
</pre>
where all arithmetic is performed mod 26.
```

```
<ul>
 <li> Start by entering a plaintext, which will be a list of column vectors:
  <ul>
   <li>
    <span class="cmd">
     P:=[matrix([[3*i],[3*i+1],[3*i+2]]) for i in 0..7]
    </span>
   </li>
  </ul>
 </li>
 <li> and a list which will receive the ciphertext:
  <ul>
   <li> <span class="cmd">C:=[zero31 for i in 1..8]</span></li>
  </ul>
 </li>
 <li> and encrypt it:
  <ul>
   <li> <span class="cmd">for i in 1..8 repeat C.i:=M*P.i</span></li>
  </ul>
 </li>
 <li> Now decrypt (first make an empty list <tt>D</tt>):
  <ul>
   <li> <span class="cmd">D:=[zero31 for i in 1..8]</span></li>
   <li> <span class="cmd">for i in 1..8 repeat D.i:=MI*C.i</span></li>
  </ul>
 </li>
 <li> If all has worked out, the list <tt>D</tt> should be the same
      plaintext you obtained earlier.
 </li>
 <li> Now change one value in the plaintext:
  <ul>
   <li> <span class="cmd">Q:=P</span></li>
   <li> <span class="cmd">Q.3:=matrix([[6],[19],[8]])</span></li>
  </ul>
 </li>
 <li> Now encrypt the new plaintext <tt>Q</tt> to a ciphertext <tt>E</tt>. How
  does this ciphertext differ from the ciphertext <tt>C</tt> obtained from
  <tt>P</tt>?
 </li>
 <li> Check that you can decrypt <tt>E</tt> to obtain <tt>Q</tt>.</li>
</ul>
<br/><br/>
<b>CBC</b>
<br/><br/>
For cipherblock chaining mode, the encryption formula for the Hill
cryptosystem is
<pre>
   C =M(P +C   )
    i    i  i-1
</pre>
and decryption is
<pre>
       -1
```

```
   P =M  C -C
    i     i  i-1
</pre>

<ul>
 <li> To enable us to use these formulas, we shall first add an extra column
  to the front of <tt>P</tt> and <tt>C</tt>:
  <ul>
   <li> <span class="cmd">P:=append([zero31],P)</span></li>
   <li> <span class="cmd">C:=append([zero31],C)</span></li>
  </ul>
 </li>
 <li> And we need to create a initialization vector:
  <ul>
   <li> <span class="cmd">IV:=matrix([[random(26)] for i in 1..3])</span></li>
  </ul>
 </li>
 <li> Now for encryption:
  <ul>
   <li> <span class="cmd">C.1:=IV</span></li>
   <li>
    <span class="cmd">
     for i in 2..9 repeat C.i:=M*(P.i+C.(i-1))
    </span>
   </li>
  </ul>
 </li>
 <li> Let’s try to decrypt the ciphertext, using the CBC formula:
  <ul>
   <li> <span class="cmd">D:=[zero31 for i in 1..9]</span></li>
   <li>
    <span class="cmd">
     for i in 2..9 repeat D.i:=MI*(C.i)-C.(i-1)
    </span>
   </li>
  </ul>
 </li>
 <li> Did it work out?</li>

 <li> As before, change one value in the plaintext:
  <ul>
   <li> <span class="cmd">Q:=P</span></li>
   <li> <span class="cmd">Q.4:=matrix([[6],[19],[8]])</span></li>
  </ul>
 </li>
 <li> Now encrypt <tt>Q</tt> to <tt>E</tt> following the procedure outlined
      above.  Compare <tt>E</tt> with <tt>C</tt>---
      how much difference is there?
      How does this difference compare with the differences of ciphertexts
      obtained with ECB?
 </li>
 <li> Just to make sure you can do it, decrypt <tt>E</tt> and make sure you
  end up with a list equal to <tt>Q</tt>.
 </li>
```

```
</ul>
<br/><br/>
<b>OFB</b>
<br/><br/>
Output feedback mode works by creating a <i>key stream</i>, and then adding
it to the plaintext to obtain the ciphertext.  With the Hill system, and an
initialization vector <tt>IV</tt>:
<pre>
   k =IV,   k =Mk
    1        i   i-1
</pre>
and then
<pre>
   c =p +k
    i  i  i
</pre>

<ul>
 <li> First, the key stream:
  <ul>
   <li> <span class="cmd">K:=[zero31 for i in 1..9]</span></li>
   <li> <span class="cmd">K.1:=IV</span></li>
   <li> <span class="cmd">for i in 2..9 repeat K.i:=M*K.(i-1)</span></li>
  </ul>
 </li>
 <li> and next the encryption:
  <ul>
   <li> <span class="cmd">for i in 2..9 repeat C.i:=K.i+P.i</span></li>
  </ul>
 </li>
 <li> What is the formula for decryption?
      Apply it to your ciphertext <tt>C</tt>.
 </li>
</ul>
\getchunk{page foot}
```

crytoclass9.xhtml

— cryptoclass9.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 9: Hash Functions</h3>
</center>
<hr/>
<br/><br/>
<b>A simple hash</b>
```

```
<br/><br/>
Given two prime numbers <tt>p</tt> and <tt>q</tt>, and their product
<tt>N</tt>, we can define a hash of a number <tt>n</tt> to be
<pre>
           n
   hash = g  (mod N)
</pre>

This is provably collision resistant, because if we want to find two hashes
which are equal, then we need to find <tt>m</tt> and <tt>n</tt> for which
<pre>
    m    n
   g  = g  (mod N)
</pre>
or that
<pre>
    m-n
   g    = 1 (mod N)
</pre>

By Euler's theorem, we know that
<pre>
    &#x3D5;(N)
   g         = 1 (mod N)
</pre>

This means that finding a collision requires finding two numbers
<tt>m</tt> and <tt>n</tt> for which
<pre>
   m = n (mod &#x3D5;(N))
</pre>

Since computing &#x3D5;(N) requires a knowledge of the factorization of
<tt>N</tt>, this will be hard if <tt>p</tt> and <tt>q</tt> are large.

<ul>
 <li> Enter the following commands:
  <ul>
   <li> <span class="cmd">p:=nextPrime(87654321)</span></li>
   <li> <span class="cmd">q:=nextPrime(98765432)</span></li>
   <li> <span class="cmd">N:=p*q</span></li>
   <li> <span class="cmd">g:=17</span></li>
  </ul>
 </li>
 <li> Read in the utility file <a href="rcm3720.input">rcm3720.input</a></li>

 <li> Now experiment with the following hashes:
  <ul>
   <li> <span class="cmd">n:=str2num("A cat")</span></li>
   <li> <span class="cmd">h:=powmod(g,n,N)</span></li>
   <li> <span class="cmd">n:=str2num("A bat")</span></li>
   <li> <span class="cmd">h:=powmod(g,n,N)</span></li>
  </ul>
 </li>
```

```
 <li> Even though the strings are very similar,
      how similar are the hash values?
 </li>
 <li> Experiment with hashing some other strings---some short, some long.</li>

 <li> Read in a text file (any text file, of any length) as follows:</li>
  <ul>
   <li>
    <span class="cmd">
     f:TextFile:=open("\\full\\path\\to\\file","input")
    </span>
   </li>
   <li> <span class="cmd">str:=""</span></li>
   <li>
    <span class="cmd">
     while not endOfFile?(f) repeat str:=concat(str,readLine(f));
    </span>
   </li>
  </ul>

 <li> Now the variable <tt>str</tt> will contain the file as one long string.
      Hash this string, by converting it to a number first.
 </li>
 <li> Try this with a few different text files,
      of different lengths---some short, some long.
 </li>
</ul>

<br/><br/>
<b>A simplified version of MASH</b>
<br/><br/>
We shall experiment with a simplified version of the MASH hash function:

 <ol>
  <li> Start with two prime numbers <tt>p</tt> and <tt>q</tt>,
      and their product <tt>N</tt>.
  </li>
  <li> Turn the data to be hashed into a single integer <tt>n</tt>.</li>

  <li> Express <tt>n</tt> as ``digits'' in base <tt>N</tt>:</li>
<pre>
                    2      3            q
   n = a + a N + a N  + a N  + ... + a N
        0   1     2      3            q
</pre>

  <li> Start with <tt>H</tt> being the largest prime less than <tt>N</tt>.</li>

  <li> For <tt>i</tt> from 0 to <tt>q</tt></li>
<pre>
                     2
      H &#60;-- (H + a_i)  +H (mod N)
</pre>
```

```
  <li> The final value of <tt>H</tt> is the hash.</li>
 </ol>

<ul>
 <li> With <tt>p</tt>, <tt>q</tt> and <tt>N</tt> as before, pick a long
      string (or the string from a text file) to be hashed, and turn it
      into a number <tt>n</tt>.
 </li>
 <li> Determine the ‘‘digits’’ in base <tt>N</tt>:
  <ul>
   <li>
    <span class="cmd">
     a:=wholeRagits(n::RadixExpansion(N))::List ZMOD N
    </span>
   </li>
  </ul>
 </li>
 <li> Now create the hash:
  <ul>
   <li> <span class="cmd">H:=prevPrime(N)</span></li>
   <li> <span class="cmd">for i in 1..#a repeat H:=(H+a.i)^2+H</span></li>
  </ul>
 </li>
 <li> Note that since the elements of the list <tt>a</tt> are already
      defined as being modulo <tt>N</tt>, we don’t have to use a mod
      function in this last step.
 </li>
 <li> Create the hashes of a few other strings and files.  What happens if you
      try to hash a really long text file?
 </li>
 <li> Experiment with hashing using some other (large) primes.</li>
</ul>
\getchunk{page foot}
```

crytoclass10.xhtml

— cryptoclass10.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 10: The Data Encryption Standard</h3>
</center>
<hr/>

The object of this lab will be to build up the necessary functions and
tools to implement simplified DES (sDES).  All operations will be done on
```

```
binary lists.  Since the definitions of the sDES functions require lists to be
indexed starting at 0, but in Axiom lists are indexed starting at 1, many of
the operations will have extra ones added at some stage.

<ul>
 <li> Save the file <tt>des.input</tt> to a directory in which you
      have write access.  Read the file into Axiom, and open up the file
      with a text editor.
 </li>

 <li> Compare the first command <tt>perm(b)</tt> with the initial
      permutation for sDES defined in page 94 of the notes.  How do the
      indices in the Axiom command relate to the indices of the
      permutation in the notes?
 </li>

 <li> Now using the above procedure as a guide, write a procedure called
      <tt>invperm</tt> to perform the inverse permutation.
 </li>

 <li> Test this procedure: it should invert the permutation you
      obtained from the <tt>perm</tt> procedure.
 </li>

<li> The <tt>subkey</tt> procedure creates two lists: one for the
     first subkey, and one for the second.  Edit the procedure to include
     the second subkey as given on the bottom of page 95.
 </li>

 <li> Write a procedure called <tt>expperm</tt> which implements the
      expansion permutation on page 96; use the <tt>perm</tt> and
      <tt>invperm</tt> procedures as guides.
 </li>

 <li> Using the <tt>sbox0</tt> procedure as a guide, write a procedure
      to implement S-box 1.
 </li>

 <li> The mixing function shown in figure 8.5 in the notes is
      implemented as <tt>mix</tt>.  This procedure has been
      commented.
 </li>

 <li> Comment each line of the <tt>feistel</tt> and <tt>sdes</tt>
      procedures in a similar fashion.
 </li>

 <li> Test the <tt>sdes</tt> procedure on the example given in the notes.
 </li>

 <li> Modify your procedure to implement sDES decryption, using the
      scheme given on page 99.
 </li>
```

```
 <li> Test that your decryption procedure works; that it decrypts the
      ciphertext produced by your encryption procedure to the original
      plaintext.
 </li>
</ul>
\getchunk{page foot}
```

——–

crytoclass11.xhtml

— cryptoclass11.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<center>
 <h2>RCM3720 Cryptography, Network and Computer Security</h2>
 <h3>Laboratory Class 11: Finite Fields</h3>
</center>
<hr/>

<ul>
 <li> Enter the following definition of the finite field
<pre>
        3
   Z [x]/(x +x+1)
    2
</pre>
  <ul>
   <li> <span class="cmd">F:=FFP(PF 2,x^3+x+1) </span></li>
  </ul>
 </li>
 <li> To perform field operations, we need to create a generator of the field:
  a symbol which can be used to generate all elements as polynomials:
  <ul>
   <li> <span class="cmd">x:=generator()$F</span>
    <br/> Now field arithmetic is easy:
   </li>
   <li> (x^2+1)(x+1) in the field:
    <ul>
     <li> <span class="cmd">(x^2+1)*(x+1)</span> </li>
    </ul>
   </li>
   <li> 1/(x^2+x):
    <ul>
     <li> <span class="cmd">1/(x^2+x)</span>
      <br/>Note that Axiom returns its answer in terms of a dummy variable.
     </li>
    </ul>
   </li>
   <li> We can also list tables of powers:
```

```
    <ul>
     <li>
      <span class="cmd">
       for i in 0..7 repeat output (i::String, x^i)
      </span>
     </li>
    </ul>
   </li>
  </ul>
 </li>
 <li> Before we enter a new field, we need to clear <tt>x</tt> and its
  properties:
  <ul>
   <li> <span class="cmd">)cl pr x </span></li>
  </ul>
  Now for a slightly bigger field:
<pre>
           4  3
   Z [x]/(x +x +1)
    2
</pre>
  <ul>
   <li> <span class="cmd">F2:=FFP(PF 2,x^4+x^3+1)</span> </li>
  </ul>
  <ul>
   <li> Create a list of powers of <tt>x</tt>. </li>
   <li> Evaluate (x^3+x+1)/(x^3+x^2) in this field. </li>
  </ul>
 </li>
 <li> Enter the Rijndael field,
<pre>
         8  4  3
 Z [x]/(x +x +x +x+1)
  2
</pre>
 and call it <tt>GR</tt>.
 </li>
 <li> Determine whether <tt>x</tt> is a primitive element in this field:
  <ul>
   <li> <span class="cmd">x:=generator()$GR</span> </li>
   <li> <span class="cmd">primitive?(x)</span> </li>
  </ul>
 </li>
 <li>
  Is <tt>x+1</tt> a primitive element?
 </li>
 <li> Investigate the workings of MixColumn.  First create the matrix:
  <ul>
   <li>
    <span class="cmd">
     M:Matrix GR:=matrix([[x,x+1,1,1],[1,x,x+1,1],[1,1,x,x+1],[x+1,1,1,x]])
    </span>
   </li>
  </ul>
```

```
 </li>
 <li>
  Instead of multiplying a matrix <tt>C</tt> by <tt>M</tt>,
  we shall just look at a single column, created randomly:
  <ul>
   <li>
    <span class="cmd">
     C:Matrix GR:=matrix([[random()$FR] for j in 1..4])
    </span>
   </li>
  </ul>
 <li>
 </li>
  These can be multiplied directly in Axiom:
  <ul>
   <li> <span class="cmd">D:=M*C</span> </li>
  </ul>
 </li>
 <li> Remarkably enough, Axiom can operate on matrices over a finite field as
      easily as it can operate on numerical matrices.  For example, given that
<pre>
   D=MC
</pre>
 </li>
 <li>  it follows that
<pre>
      -1
   C=M  D
</pre>
 </li>
 <li> or that
<pre>
    -1
   M  D-C=0
</pre>
 </li>
 <li> To test this, first create the matrix inverse:
  <ul>
   <li> <span class="cmd">MI:=inverse(M)</span> </li>
  </ul>
 </li>
 <li>
  Now multiply by <tt>D</tt> and subtract <tt>C</tt>.  What does the result
  tell you about the truth of the final equation?
 </li>
 <li> To explore MixColumn a bit more, we shall look at the inverse of
      <tt>M</tt>.  First, here's a small function which converts from
      a polynomial to an integer (treating the coefficients of the
      polynomial as digits of a binary number):
  <ul>
   <li>
    <span class="cmd">
      poly2int(p)==(tmp:=reverse(coordinates(p)),return
          integer wholeRadix(tmp::LIST INT)$RadixExpansion(2))
```

```
    </span>
   </li>
  </ul>
 </li>
 <li> First check the matrix <tt>M</tt>:
  <ul>
   <li> <span class="cmd">map((x +-> poly2int(x)::INT), M)</span></li>
  </ul>
 </li>
 <li>
  Is this what you should have?
 </li>
 <li>
  Now apply the same command but to <tt>MI</tt> instead of to <tt>M</tt>.
  What is the result?
 </li>
</ul>
\getchunk{page foot}
```

dbopbinary.xhtml

— dbopbinary.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopbinary not implemented
\getchunk{page foot}
```

dbcharacteristic.xhtml

— dbcharacteristic.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcharacteristic not implemented
\getchunk{page foot}
```

dbcomplexcomplex.xhtml

— dbcomplexcomplex.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexcomplex not implemented
\getchunk{page foot}
```

dbcomplexconjugate.xhtml

— dbcomplexconjugate.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexconjugate not implemented
\getchunk{page foot}
```

dbcomplexfactor.xhtml

— dbcomplexfactor.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexfactor not implemented
\getchunk{page foot}
```

dbcomplexdoublefloat.xhtml

— dbcomplexdoublefloat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexdoublefloat not implemented
\getchunk{page foot}
```

dbcomplexfloat.xhtml

— dbcomplexfloat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexfloat not implemented
\getchunk{page foot}
```

dbcompleximag.xhtml

— dbcompleximag.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcompleximag not implemented
\getchunk{page foot}
```

dbcomplexnorm.xhtml

— dbcomplexnorm.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexnorm not implemented
\getchunk{page foot}
```

dbcomplexreal.xhtml

— dbcomplexreal.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexreal not implemented
\getchunk{page foot}
```

dbcomplexinteger.xhtml

— dbcomplexinteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbcomplexinteger not implemented
\getchunk{page foot}
```

———

dbexpressioninteger.xhtml

— dbexpressioninteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbexpressioninteger not implemented
\getchunk{page foot}
```

———

dbfractioninteger.xhtml

— dbfractioninteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbfractioninteger not implemented
\getchunk{page foot}
```

———

dbfractionpolynomialinteger.xhtml

— dbfractionpolynomialinteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbfractionpolynomialinteger not implemented
\getchunk{page foot}
```

———

dblookup.xhtml

— dblookup.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dblookup not implemented
\getchunk{page foot}
```

———–

dbopacos.xhtml

— dbopacos.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopacos not implemented
\getchunk{page foot}
```

———–

dbopacosh.xhtml

— dbopacosh.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopacosh not implemented
\getchunk{page foot}
```

———–

dbopacot.xhtml

— dbopacot.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopacot not implemented
\getchunk{page foot}
```

———–

dbopacoth.xhtml

— dbopacoth.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopacoth not implemented
\getchunk{page foot}
```

———

dbopacsc.xhtml

— dbopacsc.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopacsc not implemented
\getchunk{page foot}
```

———

dbopacsch.xhtml

— dbopacsch.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopacsch not implemented
\getchunk{page foot}
```

———

dbopaddmod.xhtml

— dbopaddmod.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopaddmod not implemented
\getchunk{page foot}
```

———

dbopairyai.xhtml

— dbopairyai.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopairyai not implemented
\getchunk{page foot}
```

dbopairybi.xhtml

— dbopairybi.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopairybi not implemented
\getchunk{page foot}
```

dbopapproximants.xhtml

— dbopapproximants.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopapproximants not implemented
\getchunk{page foot}
```

dbopasin.xhtml

— dbopasin.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopasin not implemented
\getchunk{page foot}
```

dbopasinh.xhtml

— dbopasinh.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopasinh not implemented
\getchunk{page foot}
```

dbopasec.xhtml

— dbopasec.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopasec not implemented
\getchunk{page foot}
```

dbopasech.xhtml

— dbopasech.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopasech not implemented
\getchunk{page foot}
```

dbopatan.xhtml

— dbopatan.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopatan not implemented
\getchunk{page foot}
```

dbopatanh.xhtml

— dbopatanh.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopatanh not implemented
\getchunk{page foot}
```

———-

dbopbernoullib.xhtml

— dbopbernoullib.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopbernoullib not implemented
\getchunk{page foot}
```

———-

dbopbesseli.xhtml

— dbopbesseli.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopbesseli not implemented
\getchunk{page foot}
```

———-

dbopbesselj.xhtml

— dbopbesselj.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopbesselj not implemented
\getchunk{page foot}
```

———-

dbopbesselk.xhtml

— dbopbesselk.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopbesselk not implemented
\getchunk{page foot}
```

———-

dbopbessely.xhtml

— dbopbessely.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopbessely not implemented
\getchunk{page foot}
```

———-

dbopbeta.xhtml

— dbopbeta.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopbeta not implemented
\getchunk{page foot}
```

———-

dbopcardinalnumber.xhtml

— dbopcardinalnumber.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcardinalnumber not implemented
\getchunk{page foot}
```

———-

dbopchebyshevt.xhtml

— dbopchebyshevt.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopchebyshevt not implemented
\getchunk{page foot}
```

dbopchebyshevu.xhtml

— dbopchebyshevu.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopchebyshevu not implemented
\getchunk{page foot}
```

dbopcoefficient.xhtml

— dbopcoefficient.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcoefficient not implemented
\getchunk{page foot}
```

dbopcoefficients.xhtml

— dbopcoefficients.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcoefficients not implemented
\getchunk{page foot}
```

dbopcoerce.xhtml

— dbopcoerce.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcoerce not implemented
\getchunk{page foot}
```

dbopcolumn.xhtml

— dbopcolumn.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcolumn not implemented
\getchunk{page foot}
```

dbopcompactfraction.xhtml

— dbopcompactfraction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcompactfraction not implemented
\getchunk{page foot}
```

dbopcomplexeigenvectors.xhtml

— dbopcomplexeigenvectors.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcomplexeigenvectors not implemented
\getchunk{page foot}
```

dbopcomplexelementary.xhtml

— **dbopcomplexelementary.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcomplexelementary not implemented
\getchunk{page foot}
```

dbopcomplexintegrate.xhtml

— **dbopcomplexintegrate.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcomplexintegrate not implemented
\getchunk{page foot}
```

dbopcomplexlimit.xhtml

— **dbopcomplexlimit.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcomplexlimit not implemented
\getchunk{page foot}
```

dbopcomplexsolve.xhtml

— **dbopcomplexsolve.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcomplexsolve not implemented
\getchunk{page foot}
```

dbopcontent.xhtml

— dbopcontent.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcontent not implemented
\getchunk{page foot}
```

———-

dbopcontinuedfraction.xhtml

— dbopcontinuedfraction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcontinuedfraction not implemented
\getchunk{page foot}
```

———-

dbopconvergents.xhtml

— dbopconvergents.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopconvergents not implemented
\getchunk{page foot}
```

———-

dbopconvert.xhtml

— dbopconvert.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopconvert not implemented
\getchunk{page foot}
```

———-

dbopcopy.xhtml

— dbopcopy.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcopy not implemented
\getchunk{page foot}
```

dbopcos.xhtml

— dbopcos.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopcos not implemented
\getchunk{page foot}
```

dbopcosh.xhtml

— dbopcosh.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopcosh not implemented
\getchunk{page foot}
```

dbopcot.xhtml

— dbopcot.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcot not implemented
\getchunk{page foot}
```

dbopcoth.xhtml

— dbopcoth.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcoth not implemented
\getchunk{page foot}
```

———–

dbopcount.xhtml

— dbopcount.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcount not implemented
\getchunk{page foot}
```

———–

dbopcountableq.xhtml

— dbopcountableq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcountableq not implemented
\getchunk{page foot}
```

———–

dbopcreate3space.xhtml

— dbopcreate3space.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopcreate3space not implemented
\getchunk{page foot}
```

———–

dbopcsc.xhtml

— dbopcsc.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopcsc not implemented
\getchunk{page foot}
```

dbopcsch.xhtml

— dbopcsch.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopcsch not implemented
\getchunk{page foot}
```

dbopcurve.xhtml

— dbopcurve.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopcurve not implemented
\getchunk{page foot}
```

dbopcycleragits.xhtml

— dbopcycleragits.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopcycleragits not implemented
\getchunk{page foot}
```

dbopcyclotomic.xhtml

— dbopcyclotomic.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopcyclotomic not implemented
\getchunk{page foot}
```

dbopd.xhtml

— dbopd.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopd not implemented
\getchunk{page foot}
```

dbopdecimal.xhtml

— dbopdecimal.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdecimal not implemented
\getchunk{page foot}
```

dbopdefiningpolynomial.xhtml

— dbopdefiningpolynomial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdefiningpolynomial not implemented
\getchunk{page foot}
```

dbopdegree.xhtml

— dbopdegree.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdegree not implemented
\getchunk{page foot}
```

dbopdenom.xhtml

— dbopdenom.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdenom not implemented
\getchunk{page foot}
```

dbopdraw.xhtml

— dbopdraw.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopdraw not implemented
\getchunk{page foot}
```

dbopdeterminant.xhtml

— dbopdeterminant.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdeterminant not implemented
\getchunk{page foot}
```

dbopdiagonalmatrix.xhtml

— dbopdiagonalmatrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdiagonalmatrix not implemented
\getchunk{page foot}
```

dbopdigamma.xhtml

— dbopdigamma.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdigamma not implemented
\getchunk{page foot}
```

dbopdigits.xhtml

— dbopdigits.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdigits not implemented
\getchunk{page foot}
```

dbopdimension.xhtml

— dbopdimension.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdimension not implemented
\getchunk{page foot}
```

dbopdivide.xhtml

— **dbopdivide.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdivide not implemented
\getchunk{page foot}
```

————

dbopdivisors.xhtml

— **dbopdivisors.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopdivisors not implemented
\getchunk{page foot}
```

————

dbopei.xhtml

— **dbopei.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopei not implemented
\getchunk{page foot}
```

————

dbopeigenmatrix.xhtml

— **dbopeigenmatrix.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopeigenmatrix not implemented
\getchunk{page foot}
```

————

dbopeigenvalues.xhtml

— dbopeigenvalues.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopeigenvalues not implemented
\getchunk{page foot}
```

dbopeigenvector.xhtml

— dbopeigenvector.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopeigenvector not implemented
\getchunk{page foot}
```

dbopeigenvectors.xhtml

— dbopeigenvectors.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopeigenvectors not implemented
\getchunk{page foot}
```

dbopelt.xhtml

— dbopelt.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopelt not implemented
\getchunk{page foot}
```

dbopequal.xhtml

— dbopequal.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopequal not implemented
\getchunk{page foot}
```

———-

dbopeulere.xhtml

— dbopeulere.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopeulere not implemented
\getchunk{page foot}
```

———-

dbopeulerphi.xhtml

— dbopeulerphi.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopeulerphi not implemented
\getchunk{page foot}
```

———-

dbopeval.xhtml

— dbopeval.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopeval not implemented
\getchunk{page foot}
```

———-

dbopevenq.xhtml

— dbopevenq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopevenq not implemented
\getchunk{page foot}
```

dbopexp.xhtml

— dbopexp.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopexp not implemented
\getchunk{page foot}
```

dbopexquo.xhtml

— dbopexquo.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopexquo not implemented
\getchunk{page foot}
```

dbopfactor.xhtml

— dbopfactor.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfactor not implemented
\getchunk{page foot}
```

dbopfactorfraction.xhtml

— dbopfactorfraction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfactorfraction not implemented
\getchunk{page foot}
```

dbopfibonacci.xhtml

— dbopfibonacci.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfibonacci not implemented
\getchunk{page foot}
```

dbopfiniteq.xhtml

— dbopfiniteq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfiniteq not implemented
\getchunk{page foot}
```

dbopfirstdenom.xhtml

— dbopfirstdenom.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfirstdenom not implemented
\getchunk{page foot}
```

dbopfirstnumer.xhtml

— **dbopfirstnumer.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfirstnumer not implemented
\getchunk{page foot}
```

————

dbopfractragits.xhtml

— **dbopfractragits.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopfractragits not implemented
\getchunk{page foot}
```

————

dbopfractionpart.xhtml

— **dbopfractionpart.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopfractionpart not implemented
\getchunk{page foot}
```

————

dbopgamma.xhtml

— **dbopgamma.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopgamma not implemented
\getchunk{page foot}
```

————

dbopgcd.xhtml

— dbopgcd.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopgcd not implemented
\getchunk{page foot}
```

———-

dbophermiteh.xhtml

— dbophermiteh.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbophermiteh not implemented
\getchunk{page foot}
```

———-

dbophex.xhtml

— dbophex.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbophex not implemented
\getchunk{page foot}
```

———-

dbophorizconcat.xhtml

— dbophorizconcat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbophorizconcat not implemented
\getchunk{page foot}
```

———-

dbophtrigs.xhtml

— dbophtrigs.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbophtrigs not implemented
\getchunk{page foot}
```

———–

dbophypergeometric0f1.xhtml

— dbophypergeometric0f1.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbophypergeometric0f1 not implemented
\getchunk{page foot}
```

———–

dbopinteger.xhtml

— dbopinteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopinteger not implemented
\getchunk{page foot}
```

———–

dbopintegrate.xhtml

— dbopintegrate.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopintegrate not implemented
\getchunk{page foot}
```

———–

dbopinverse.xhtml

— dbopinverse.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopinverse not implemented
\getchunk{page foot}
```

dbopinvmod.xhtml

— dbopinvmod.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopinvmod not implemented
\getchunk{page foot}
```

dbopjacobi.xhtml

— dbopjacobi.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopjacobi not implemented
\getchunk{page foot}
```

dboplaguerrel.xhtml

— dboplaguerrel.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dboplaguerrel not implemented
\getchunk{page foot}
```

dboplaurent.xhtml

— dboplaurent.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboplaurent not implemented
\getchunk{page foot}
```

dboplcm.xhtml

— dboplcm.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dboplcm not implemented
\getchunk{page foot}
```

dbopleadingcoefficient.xhtml

— dbopleadingcoefficient.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopleadingcoefficient not implemented
\getchunk{page foot}
```

dbopleadingmonomial.xhtml

— dbopleadingmonomial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopleadingmonomial not implemented
\getchunk{page foot}
```

dboplegendre.xhtml

— dboplegendre.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboplegendre not implemented
\getchunk{page foot}
```

dboplength.xhtml

— dboplength.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboplength not implemented
\getchunk{page foot}
```

dboplimit.xhtml

— dboplimit.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboplimit not implemented
\getchunk{page foot}
```

dboplog.xhtml

— dboplog.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dboplog not implemented
\getchunk{page foot}
```

dboploggamma.xhtml

— dboploggamma.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboploggamma not implemented
\getchunk{page foot}
```

dbopmainvariable.xhtml

— dbopmainvariable.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmainvariable not implemented
\getchunk{page foot}
```

dbopmakegraphimage.xhtml

— dbopmakegraphimage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopmakegraphimage not implemented
\getchunk{page foot}
```

dbopmakeobject.xhtml

— dbopmakeobject.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopmakeobject not implemented
\getchunk{page foot}
```

dbopmakeviewport3d.xhtml

— dbopmakeviewport3d.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopmakeviewport3d not implemented
\getchunk{page foot}
```

dbopmap.xhtml

— dbopmap.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmap not implemented
\getchunk{page foot}
```

dbopmapbang.xhtml

— dbopmapbang.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmapbang not implemented
\getchunk{page foot}
```

dbopmatrix.xhtml

— dbopmatrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmatrix not implemented
\getchunk{page foot}
```

dbopmax.xhtml

— dbopmax.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmax not implemented
\getchunk{page foot}
```

dbopmemberq.xhtml

— dbopmemberq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmemberq not implemented
\getchunk{page foot}
```

dbopmin.xhtml

— dbopmin.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmin not implemented
\getchunk{page foot}
```

dbopminimumdegree.xhtml

— dbopminimumdegree.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopminimumdegree not implemented
\getchunk{page foot}
```

dbopminus.xhtml

— dbopminus.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopminus not implemented
\getchunk{page foot}
```

dbopmoebiusmu.xhtml

— dbopmoebiusmu.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmoebiusmu.xhtml not implemented
\getchunk{page foot}
```

dbopmonicdivide.xhtml

— dbopmonicdivide.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmonicdivide not implemented
\getchunk{page foot}
```

dbopmulmod.xhtml

— dbopmulmod.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopmulmod not implemented
\getchunk{page foot}
```

dbopncols.xhtml

— dbopncols.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopncols not implemented
\getchunk{page foot}
```

———

dbopnegativeq.xhtml

— dbopnegativeq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnegativeq not implemented
\getchunk{page foot}
```

———

dbopnew.xhtml

— dbopnew.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnew not implemented
\getchunk{page foot}
```

———

dbopnextprime.xhtml

— dbopnextprime.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnextprime not implemented
\getchunk{page foot}
```

———

dbopnorm.xhtml

— **dbopnorm.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnorm not implemented
\getchunk{page foot}
```

dbopnrows.xhtml

— **dbopnrows.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnrows not implemented
\getchunk{page foot}
```

dbopnthfractionalterm.xhtml

— **dbopnthfractionalterm.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnthfractionalterm not implemented
\getchunk{page foot}
```

dbopnthroot.xhtml

— **dbopnthroot.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnthroot not implemented
\getchunk{page foot}
```

dbopnumer.xhtml

— dbopnumer.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnumer not implemented
\getchunk{page foot}
```

dbopnumeric.xhtml

— dbopnumeric.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnumeric not implemented
\getchunk{page foot}
```

dbopoddq.xhtml

— dbopoddq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopoddq not implemented
\getchunk{page foot}
```

dboponedimensionalarray.xhtml

— dboponedimensionalarray.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dboponedimensionalarray not implemented
\getchunk{page foot}
```

dbopoperator.xhtml

— dbopoperator.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopoperator not implemented
\getchunk{page foot}
```

————-

dboporthonormalbasis.xhtml

— dboporthonormalbasis.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboporthonormalbasis not implemented
\getchunk{page foot}
```

————-

dbopoutputfixed.xhtml

— dbopoutputfixed.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopoutputfixed not implemented
\getchunk{page foot}
```

————-

dbopoutputfloating.xhtml

— dbopoutputfloating.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopoutputfloating not implemented
\getchunk{page foot}
```

————-

dbopoutputgeneral.xhtml

— dbopoutputgeneral.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopoutputgeneral not implemented
\getchunk{page foot}
```

dbopoutputspacing.xhtml

— dbopoutputspacing.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopoutputspacing not implemented
\getchunk{page foot}
```

dboppadicfraction.xhtml

— dboppadicfraction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppadicfraction not implemented
\getchunk{page foot}
```

dbopnullity.xhtml

— dbopnullity.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnullity not implemented
\getchunk{page foot}
```

dbopnullspace.xhtml

— dbopnullspace.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnullspace not implemented
\getchunk{page foot}
```

———-

dbopnumberoffractionalterms.xhtml

— dbopnumberoffractionalterms.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopnumberoffractionalterms not implemented
\getchunk{page foot}
```

———-

dboppartialfraction.xhtml

— dboppartialfraction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppartialfraction not implemented
\getchunk{page foot}
```

———-

dboppartialquotients.xhtml

— dboppartialquotients.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppartialquotients not implemented
\getchunk{page foot}
```

———-

dbopplus.xhtml

— dbopplus.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopplus not implemented
\getchunk{page foot}
```

dboppattern.xhtml

— dboppattern.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppattern not implemented
\getchunk{page foot}
```

dboppermanent.xhtml

— dboppermanent.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppermanent not implemented
\getchunk{page foot}
```

dboppi.xhtml

— dboppi.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppi not implemented
\getchunk{page foot}
```

dboppolygamma.xhtml

— dboppolygamma.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppolygamma not implemented
\getchunk{page foot}
```

———-

dboppositiveq.xhtml

— dboppositiveq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppositiveq not implemented
\getchunk{page foot}
```

———-

dboppositiveremainder.xhtml

— dboppositiveremainder.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppositiveremainder not implemented
\getchunk{page foot}
```

———-

dbopprefixragits.xhtml

— dbopprefixragits.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopprefixragits not implemented
\getchunk{page foot}
```

———-

dbopprevprime.xhtml

— dbopprevprime.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopprevprime not implemented
\getchunk{page foot}
```

dbopprimefactor.xhtml

— dbopprimefactor.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopprimefactor not implemented
\getchunk{page foot}
```

dbopprimeq.xhtml

— dbopprimeq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopprimeq not implemented
\getchunk{page foot}
```

dbopprimes.xhtml

— dbopprimes.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopprimes not implemented
\getchunk{page foot}
```

dboppuiseux.xhtml

— dboppuiseux.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboppuiseux not implemented
\getchunk{page foot}
```

dbopqelt.xhtml

— dbopqelt.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopqelt not implemented
\getchunk{page foot}
```

dbopqseteltbang.xhtml

— dbopqseteltbang.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopqseteltbang not implemented
\getchunk{page foot}
```

dbopquatern.xhtml

— dbopquatern.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopquatern not implemented
\getchunk{page foot}
```

dbopradicaleigenvectors.xhtml

— dbopradicaleigenvectors.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopradicaleigenvectors not implemented
\getchunk{page foot}
```

dbopradicalsolve.xhtml

— dbopradicalsolve.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopradicalsolve not implemented
\getchunk{page foot}
```

dboprank.xhtml

— dboprank.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprank not implemented
\getchunk{page foot}
```

dbopratdenom.xhtml

— dbopratdenom.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopratdenom not implemented
\getchunk{page foot}
```

dboprealeigenvectors.xhtml

— dboprealeigenvectors.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprealeigenvectors not implemented
\getchunk{page foot}
```

———-

dboprealelementary.xhtml

— dboprealelementary.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprealelementary not implemented
\getchunk{page foot}
```

———-

dbopreduce.xhtml

— dbopreduce.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopreduce not implemented
\getchunk{page foot}
```

———-

dbopreductum.xhtml

— dbopreductum.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopreductum not implemented
\getchunk{page foot}
```

———-

dboprem.xhtml

— **dboprem.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprem not implemented
\getchunk{page foot}
```

dbopquo.xhtml

— **dbopquo.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopquo not implemented
\getchunk{page foot}
```

dbopresetvariableorder.xhtml

— **dbopresetvariableorder.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopresetvariableorder not implemented
\getchunk{page foot}
```

dbopresultant.xhtml

— **dbopresultant.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopresultant not implemented
\getchunk{page foot}
```

dboprootof.xhtml

— dboprootof.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprootof not implemented
\getchunk{page foot}
```

dboprootsimp.xhtml

— dboprootsimp.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprootsimp not implemented
\getchunk{page foot}
```

dboprootsof.xhtml

— dboprootsof.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprootsof not implemented
\getchunk{page foot}
```

dbopseries.xhtml

— dbopseries.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopseries not implemented
\getchunk{page foot}
```

dbopround.xhtml

— dbopround.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopround not implemented
\getchunk{page foot}
```

dboprow.xhtml

— dboprow.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprow not implemented
\getchunk{page foot}
```

dboprowechelon.xhtml

— dboprowechelon.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboprowechelon not implemented
\getchunk{page foot}
```

dbopsetcolumnbang.xhtml

— dbopsetcolumnbang.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsetcolumnbang not implemented
\getchunk{page foot}
```

dbopseteltbang.xhtml

— dbopseteltbang.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopseteltbang not implemented
\getchunk{page foot}
```

dbopsetrowbang.xhtml

— dbopsetrowbang.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsetrowbang not implemented
\getchunk{page foot}
```

dbopsetelt.xhtml

— dbopsetelt.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsetelt not implemented
\getchunk{page foot}
```

dbopsetsubmatrixbang.xhtml

— dbopsetsubmatrixbang.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsetsubmatrixbang not implemented
\getchunk{page foot}
```

dbopsign.xhtml

— dbopsign.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsign not implemented
\getchunk{page foot}
```

dbopsimplify.xhtml

— dbopsimplify.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsimplify not implemented
\getchunk{page foot}
```

dbopseriessolve.xhtml

— dbopseriessolve.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopseriessolve not implemented
\getchunk{page foot}
```

dbopsin.xhtml

— dbopsin.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsin not implemented
\getchunk{page foot}
```

dbopsingleintegerand.xhtml

— dbopsingleintegerand.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsingleintegerand not implemented
\getchunk{page foot}
```

dbopsingleintegernot.xhtml

— dbopsingleintegernot.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsingleintegernot not implemented
\getchunk{page foot}
```

dbopsingleintegeror.xhtml

— dbopsingleintegeror.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsingleintegeror not implemented
\getchunk{page foot}
```

dbopsingleintegerxor.xhtml

— dbopsingleintegerxor.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsingleintegerxor not implemented
\getchunk{page foot}
```

dbopsec.xhtml

— dbopsec.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopsec not implemented
\getchunk{page foot}
```

dbopsech.xhtml

— dbopsech.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopsech not implemented
\getchunk{page foot}
```

dbopsetvariableorder.xhtml

— dbopsetvariableorder.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopsetvariableorder not implemented
\getchunk{page foot}
```

dbopsinh.xhtml

— dbopsinh.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopsinh not implemented
\getchunk{page foot}
```

dbopsolve.xhtml

— dbopsolve.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsolve not implemented
\getchunk{page foot}
```

dbopsqrt.xhtml

— dbopsqrt.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsqrt not implemented
\getchunk{page foot}
```

dbopstar.xhtml

— dbopstar.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
       dbopstar not implemented
\getchunk{page foot}
```

dbopstarstar.xhtml

— dbopstarstar.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopstarstar not implemented
\getchunk{page foot}
```

dbopsubmatrix.xhtml

— dbopsubmatrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsubmatrix not implemented
\getchunk{page foot}
```

dbopsubmod.xhtml

— dbopsubmod.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopsubmod not implemented
\getchunk{page foot}
```

dbopsurface.xhtml

— dbopsurface.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopsurface not implemented
\getchunk{page foot}
```

dbopsumofkthpowerdivisors.xhtml

— dbopsumofkthpowerdivisors.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dbopsumofkthpowerdivisors.xhtml not implemented
\getchunk{page foot}
```

dboptan.xhtml

— **dboptan.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dboptan not implemented
\getchunk{page foot}
```

———-

dboptanh.xhtml

— **dboptanh.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dboptanh not implemented
\getchunk{page foot}
```

———-

dboptaylor.xhtml

— **dboptaylor.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboptaylor not implemented
\getchunk{page foot}
```

———-

dboptimes.xhtml

— **dboptimes.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     dboptimes not implemented
\getchunk{page foot}
```

———-

dboptotaldegree.xhtml

— dboptotaldegree.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboptotaldegree not implemented
\getchunk{page foot}
```

dboptrace.xhtml

— dboptrace.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboptrace not implemented
\getchunk{page foot}
```

dboptranspose.xhtml

— dboptranspose.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboptranspose not implemented
\getchunk{page foot}
```

dboptrigs.xhtml

— dboptrigs.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dboptrigs not implemented
\getchunk{page foot}
```

dboptruncate.xhtml

— dboptruncate.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dboptruncate not implemented
\getchunk{page foot}
```

dbopvariables.xhtml

— dbopvariables.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopvariables not implemented
\getchunk{page foot}
```

dbopvectorise.xhtml

— dbopvectorise.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopvectorise not implemented
\getchunk{page foot}
```

dbopvectorspace.xhtml

— dbopvectorspace.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopvectorspace not implemented
\getchunk{page foot}
```

dbopwrite.xhtml

— dbopwrite.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
dbopwrite not implemented
\getchunk{page foot}
```

dbopzeroof.xhtml

— dbopzeroof.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopzeroof not implemented
\getchunk{page foot}
```

dbopzerosof.xhtml

— dbopzerosof.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopzerosof not implemented
\getchunk{page foot}
```

dbopzeroq.xhtml

— dbopzeroq.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopzeroq not implemented
\getchunk{page foot}
```

dbopvertconcat.xhtml

— dbopvertconcat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopvertconcat not implemented
\getchunk{page foot}
```

dbopwholepart.xhtml

— dbopwholepart.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopwholepart not implemented
\getchunk{page foot}
```

dbpolynomialinteger.xhtml

— dbpolynomialinteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbpolynomialinteger not implemented
\getchunk{page foot}
```

dbpolynomialfractioninteger.xhtml

— dbpolynomialfractioninteger.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbpolynomialfractioninteger not implemented
\getchunk{page foot}
```

dbopwholeragits.xhtml

— dbopwholeragits.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      dbopwholeragits not implemented
\getchunk{page foot}
```

definiteintegral.xhtml

— definiteintegral.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    var myform = document.getElementById("form2");
    var ans='integrate('+myform.expr.value+','+myform.vars.value+'='+
           myform.lower.value+'..'+myform.upper.value+')';
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <form id="form2">
   Enter the function you want to integrate:<br/>
   <input type="text" id="expr" tabindex="10" size="50"
     value="1/(x^2+6)"/><br/>
   Enter the variable of integration:<br/>
   <input type="text" id="vars" tabindex="20" size="5" value="x"/><br/>
   Enter a lower limit:<br/>
   <input type="text" id="lower" tabindex="30" value="%minusInfinity"/><br/>
   Enter an upper limit:<br/>
   <input type="text" id="upper" tabindex="40" value="%plusInfinity"/><br/>
  </form>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

determinantofhilbert.xhtml

— determinantofhilbert.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Example: Determinant of a Hilbert Matrix</div>
  <hr/>
Consider the problem of computing the determinant of a 10 by 10
Hilbert matrix. The (i,j)-th entry of a Hilbert matrix is given by
1/(i+j+1).

First do the computation using rational numbers to obtain the exact result.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
  value="a:Matrix FRAC INT:=matrix [ [1/(i+j+1) for j in 0..9] for i in 0..9]"
   />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
This version of <a href="dbopdeterminant.xhtml">determinant</a> uses
Gaussian elimination.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="d:=determinant a" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="d::Float" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Now use hardware floats.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
  value=
   "b:Matrix DFLOAT:=matrix [ [1/(i+j+1$DFLOAT) for j in 0..9] for i in 0..9]"
   />
  <div id="ansp4"><div></div></div>
```

```
 </li>
</ul>
The result given by hardware floats is correct to only four significant
digits of precision. In the jargon of numerical analysis, the Hilbert
matrix is said to be "ill-conditioned".
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="determinant b" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Now repeat the computation at a higher precision using
<a href="db.xhtml?Float">Float</a>
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="digits 40" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
  value=
   "c:Matrix Float:=matrix [ [1/(i+j+1$Float) for j in 0..9] for i in 0..9]"
   />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p7','p8']);"
    value="determinant c" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Reset <a href="dbopdigits.xhtml">digits</a> to its default value.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="digits 20" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

differentiate.xhtml

— differentiate.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    var myform = document.getElementById("form2");
    return('differentiate('+myform.expr.value+',['+
                            myform.vars.value+'],['+
                            myform.powers.value+'])');
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <form id="form2">
   Enter the function you want to differentiate:<br/>
   <input type="text" id="expr" tabindex="10" size="50" value="sin(x*y)"/><br/>
   List the variables you want to differentiate with respect to:<br/>
   <input type="text" id="vars" tabindex="20" value="x,y"/><br/>
   List the number of times you want to differentiate with respect
   to each variable (leave blank if once for each)<br/>
   <input type="text" id="powers" tabindex="30" value="1,2"/><br/>
  </form>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

dlmf.xhtml

— dlmf.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function by R. A. Askey and R. Roy
  </div>
  <hr/>
<p>
The Gamma function is an extension of the factorial function to
real and complex numbers. For positive integers,
<m:math display="inline">
```

```
 <m:mi mathvariant="normal">&#x0393;</m:mi>
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mi>n</m:mi>
  <m:mo>)</m:mo>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mi>n</m:mi>
  <m:mo>-</m:mo>
  <m:mn>1</m:mn>
  <m:mo>)</m:mo>
  <m:mi mathvariant="normal">!</m:mi>
 </m:mrow>
</m:math>.
</p>

<p>
These pages explore Axiom's facilities for handling the Gamma function.
In particular we try to show that Axiom conforms to published standards.
</p>
<ul>
 <li><b>Notation</b></li>
 <ul>
  <li><a href="dlmfnotation.xhtml">Notation</a></li>
 </ul>
 <li><b>Properties</b></li>
 <ul>
  <li><a href="dlmfdefinitions.xhtml">Definitions</a></li>
  <li><a href="dlmfgraphics.xhtml">Graphics</a></li>
  <li><a href="dlmfspecialvaluesandextrema.xhtml">
       Special Values and Extrema</a></li>
  <li><a href="dlmffunctionrelations.xhtml">Function Relations</a></li>
  <li><a href="dlmfinequalities.xhtml">Inequalities</a></li>
  <li><a href="dlmfseriesexpansions.xhtml">Series Expansions</a></li>
  <li><a href="dlmfinfiniteproducts.xhtml">Infinite Products</a></li>
  <li><a href="dlmfintegralrepresentations.xhtml">
       Integral Representations</a></li>
  <li><a href="dlmfcontinuedfractions.xhtml">Continued Fractions</a></li>
  <li><a href="dlmfasymptoticexpansions.xhtml">Asymptotic Expansions</a></li>
  <li><a href="dlmfbetafunction.xhtml">Beta Function</a></li>
  <li><a href="dlmfintegrals.xhtml">Integrals</a></li>
  <li><a href="dlmfmultidimensionalintegral.xhtml">
       Multidimensional Integral</a></li>
  <li><a href="dlmfpolygammafunctions.xhtml">Polygamma Functions</a></li>
  <li><a href="dlmfsums.xhtml">Sums</a></li>
  <li><a href="dlmfbarnesgfunction.xhtml">
       Barnes <i>G</i>-Function (Double Gamma Function)</a></li>
  <li><a href="dlmfqgammaandbetafunctions.xhtml">
       <i>q</i>-Gamma and Beta Functions</a></li>
 </ul>
 <li><b>Applications</b></li>
 <ul>
```

```
  <li><a href="dlmfmathematicalapplications.xhtml">
      Mathematical Applications</a></li>
  <li><a href="dlmfphysicalapplications.xhtml">
      Physical Applications</a></li>
 </ul>
 <li><b>Computation</b></li>
 <ul>
  <li><a href="dlmfmethodsofcomputation.xhtml">
      Methods of Computation</a></li>
  <li><a href="dlmftables.xhtml">Tables</a></li>
  <li><a href="dlmfapproximations.xhtml">Approximations</a></li>
  <li>Axiom Software</li>
 </ul>
</ul>
\getchunk{page foot}
```

———-

dlmfapproximations.xhtml

— dlmfapproximations.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Approximations
  </div>
  <hr/>
<h3>Approximations</h3>
<h6>Contents</h6>
<ul>
 <li>Rational Approximations</li>
 <li>Expansions in Chebyshev Series</li>
 <li>Approximations in the Complex Plane</li>
</ul>

<h4>Rational Approximations</h4>

<p>
 <a href="http://dlmf.nist.gov/Contents/bib/C#cody:1967:ca">
  Cody and Hillstrom(1967)
 </a> gives minimax rational approximations for
 <m:math display="inline">
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
```

```
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math> for the ranges
<m:math display="inline">
 <m:mrow>
  <m:mn>0.5</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>1.5</m:mn>
 </m:mrow>
</m:math>,

<m:math display="inline">
 <m:mrow>
  <m:mn>1.5</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>4</m:mn>
 </m:mrow>
</m:math>,
<m:math display="inline">
 <m:mrow>
  <m:mn>4</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>12</m:mn>
 </m:mrow>
</m:math>; precision is variable.
<a href="http://dlmf.nist.gov/Contents/bib/H#hart:1968:ca">
 Hart <em>et.al.</em>(1968)
</a> gives minimax polynomial and rational approximations to

<m:math display="inline">
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>x</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> and
<m:math display="inline">
 <m:mrow>
  <m:mi>ln</m:mi>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
```

```
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math> in the intervals

<m:math display="inline">
 <m:mrow>
  <m:mn>0</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
</m:math>,

<m:math display="inline">
 <m:mrow>
  <m:mn>8</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>1000</m:mn>
 </m:mrow>
</m:math>,

<m:math display="inline">
 <m:mrow>
  <m:mn>12</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>1000</m:mn>
 </m:mrow>
</m:math>; precision is variable.

<a href="http://dlmf.nist.gov/Contents/bib/C#cody:1973:cap">
 Cody <em>et.al.</em>(1973)
</a> gives minimax rational approximations for
<m:math display="inline">
 <m:mrow>
  <m:mi>&#x03C8;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>x</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> for the ranges

<m:math display="inline">
```

```
  <m:mrow>
   <m:mn>0.5</m:mn>
   <m:mo>&#x2264;</m:mo>
   <m:mi>x</m:mi>
   <m:mo>&#x2264;</m:mo>
   <m:mn>3</m:mn>
  </m:mrow>
 </m:math> and

 <m:math display="inline">
  <m:mrow>
   <m:mn>3</m:mn>
   <m:mo>&#x2264;</m:mo>
   <m:mi>x</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math>; precision is variable.
</p>

<p>For additional approximations see
 <a href="http://dlmf.nist.gov/Contents/bib/H#hart:1968:ca">
  Hart <em>et.al.</em>(1968)
 </a>(Appendix B),
 <a href="http://dlmf.nist.gov/Contents/bib/L#luke:1975:mfa">
  Luke(1975)
 </a>(pp. 2223), and
 <a href="http://dlmf.nist.gov/Contents/bib/W#weniger:2003:dig">
  Weniger(2003)
 </a>.
</p>

<h4>Expansions in Chebyshev Series</h4>

<p>
 <a href="http://dlmf.nist.gov/Contents/bib/L#luke:1969:sfa2">
  Luke(1969)
 </a>
 gives the coefficients to 20D for the Chebyshev-series expansions of
 <m:math display="inline">
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>1</m:mn>
     <m:mo>+</m:mo>
     <m:mi>x</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
```

```
  <m:mfrac bevelled="true">
   <m:mn>1</m:mn>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mo>+</m:mo>
      <m:mi>x</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
 </m:math>,

 <m:math display="inline">
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>x</m:mi>
     <m:mo>+</m:mo>
     <m:mn>3</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>,

 <m:math display="inline">
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mn>3</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>,

 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03C8;</m:mi>
   <m:mrow>
```

```
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>x</m:mi>
    <m:mo>+</m:mo>
    <m:mn>3</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math>, and the first six derivatives of

<m:math display="inline">
 <m:mrow>
  <m:mi>&#x03C8;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>x</m:mi>
    <m:mo>+</m:mo>
    <m:mn>3</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> for

<m:math display="inline">
 <m:mrow>
  <m:mn>0</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>x</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
</m:math>. These coefficients are reproduced in
<a href="http://dlmf.nist.gov/Contents/bib/L#luke:1975:mfa">
 Luke(1975)
</a>.

<a href="http://dlmf.nist.gov/Contents/bib/C#clenshaw:1962:csm">
 Clenshaw(1962)
</a> also gives 20D Chebyshev-series coefficients for
<m:math display="inline">
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>+</m:mo>
    <m:mi>x</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
```

```
  </m:mrow>
 </m:math> and its reciprocal for

 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&#x2264;</m:mo>
   <m:mi>x</m:mi>
   <m:mo>&#x2264;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>. See
 <a href="http://dlmf.nist.gov/Contents/bib/L#luke:1975:mfa">
  Luke(1975)
 </a>(pp.2223) for additional expansions.
</p>

<h4>Approximations in the Complex Plane</h4>

<p>Rational approximations for
 <m:math display="inline">
  <m:mfrac bevelled="true">
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi>A</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
 </m:math>, where

 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>A</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
```

```
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>&#x03C0;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mfrac bevelled="true">
   <m:mn>1</m:mn>
   <m:mn>2</m:mn>
  </m:mfrac>
 </m:msup>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>z</m:mi>
    <m:mo>+</m:mo>
    <m:mi>c</m:mi>
    <m:mo>+</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>+</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:msup>
 <m:mrow>
  <m:mi>exp</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mi>c</m:mi>
      <m:mo>+</m:mo>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
```

```
         </m:mfrac>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>, and approximations for

 <m:math display="inline">
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>z</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> based on the Pad approximants for two forms of the incomplete
  gamma function are in
 <a href="http://dlmf.nist.gov/Contents/bib/L#luke:1969:sfa2">
  Luke(1969)
 </a>.
 <a href="http://dlmf.nist.gov/Contents/bib/L#luke:1975:mfa">
  Luke(1975)
 </a>(pp.1316) provides explicit rational approximations for

 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>&#x03C8;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>+</m:mo>
   <m:mi>&#x03B3;</m:mi>
  </m:mrow>
 </m:math>
</p>
\getchunk{page foot}
```

dlmfasymptoticexpansions.xhtml

— dlmfasymptoticexpansions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Asymptotic Expansions
  </div>
  <hr/>
<h3>Asymptotic Expansions</h3>

<h6>Contents</h6>
<ul>
 <li>Poincar-Type Expansions</li>
 <li>Error Bounds and Exponential Improvement</li>
 <li>Ratios</li>
</ul>

<h4>Poincar-Type Expansions</h4>

<p>As
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2192;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math> in the sector
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mo>-</m:mo>
     <m:mi>&#x03B4;</m:mi>
    </m:mrow>
    <m:mrow>
```

```
     <m:mo>(</m:mo>
     <m:mrow>
      <m:none/>
      <m:mo>&lt;</m:mo>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>,
</p>

<a name="equation1"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>&#x223C;</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>-</m:mo>
         <m:mstyle displaystyle="false">
          <m:mfrac>
           <m:mn>1</m:mn>
           <m:mn>2</m:mn>
          </m:mfrac>
         </m:mstyle>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mi>ln</m:mi>
        <m:mspace width="0.2em"/>
        <m:mi>z</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>-</m:mo>
```

```
 <m:mi>z</m:mi>
</m:mrow>
<m:mo>+</m:mo>
<m:mrow>
 <m:mstyle displaystyle="false">
  <m:mfrac>
   <m:mn>1</m:mn>
   <m:mn>2</m:mn>
  </m:mfrac>
 </m:mstyle>
 <m:mrow>
  <m:mi>ln</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>&#x03C0;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>+</m:mo>
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x2211;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:munderover>
 <m:mfrac>
  <m:msub>
   <m:mi>B</m:mi>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>k</m:mi>
   </m:mrow>
  </m:msub>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mi>k</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>k</m:mi>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
```

```
         <m:mo>)</m:mo>
        </m:mrow>
        <m:msup>
         <m:mi>z</m:mi>
         <m:mrow>
          <m:mrow>
           <m:mn>2</m:mn>
           <m:mi>k</m:mi>
          </m:mrow>
          <m:mo>-</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
        </m:msup>
       </m:mrow>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<a name="equation2"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>&#x223C;</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>ln</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>z</m:mi>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
```

```
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:mfrac>
      <m:msub>
       <m:mi>B</m:mi>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>k</m:mi>
       </m:mrow>
      </m:msub>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>k</m:mi>
       <m:msup>
        <m:mi>z</m:mi>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>k</m:mi>
        </m:mrow>
       </m:msup>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>
</div>

<p>For the Bernoulli numbers
<m:math>
 <m:msub>
  <m:mi>B</m:mi>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mi>k</m:mi>
  </m:mrow>
 </m:msub>
</m:math>,
Also,
</p>

<a name="equation3"/>
<div align="center">
<m:math display="block">
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
```

```
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>z</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>&#x223C;</m:mo>
 <m:mrow>
  <m:msup>
   <m:mi mathvariant="normal">&#x2147;</m:mi>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mi>z</m:mi>
   </m:mrow>
  </m:msup>
  <m:msup>
   <m:mi>z</m:mi>
   <m:mi>z</m:mi>
  </m:msup>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mfrac>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mi>z</m:mi>
    </m:mfrac>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mfrac bevelled="true">
    <m:mn>1</m:mn>
    <m:mn>2</m:mn>
   </m:mfrac>
  </m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:munderover>
     <m:mo movablelimits="false">&#x2211;</m:mo>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>=</m:mo>
      <m:mn>0</m:mn>
     </m:mrow>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:munderover>
    <m:mfrac>
     <m:msub>
      <m:mi>g</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
     <m:msup>
```

```
        <m:mi>z</m:mi>
        <m:mi>k</m:mi>
       </m:msup>
      </m:mfrac>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>
</div>

<div align="center">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>0</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>1</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>12</m:mn>
    </m:mfrac>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>2</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>288</m:mn>
```

```
    </m:mfrac>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>3</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:mn>139</m:mn>
      <m:mn>51840</m:mn>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>4</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:mn>571</m:mn>
      <m:mn>24 88320</m:mn>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>5</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>1 63879</m:mn>
     <m:mn>2090 18880</m:mn>
```

```
    </m:mfrac>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mn>6</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>52 46819</m:mn>
     <m:mn>7 52467 96800</m:mn>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>g</m:mi>
     <m:mi>k</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msqrt>
      <m:mn>2</m:mn>
     </m:msqrt>
     <m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mstyle>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>k</m:mi>
     </m:msub>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>k</m:mi>
      </m:mrow>
```

```
    </m:msub>
   </m:mrow>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
   <m:msub>
    <m:mi>a</m:mi>
    <m:mn>0</m:mn>
   </m:msub>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
    <m:msqrt>
     <m:mn>2</m:mn>
    </m:msqrt>
   </m:mrow>
  </m:mrow>
 </m:math>, and
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mn>0</m:mn>
      </m:msub>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:msub>
       <m:mi>a</m:mi>
```

```
       <m:mn>1</m:mn>
      </m:msub>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:msub>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>3</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mn>2</m:mn>
      </m:msub>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>-</m:mo>
        <m:mn>2</m:mn>
       </m:mrow>
      </m:msub>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:mfrac>
      </m:mstyle>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mn>0</m:mn>
      </m:msub>
```

```
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mstyle displaystyle="false">
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mi>k</m:mi>
      </m:mfrac>
     </m:mstyle>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msub>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>.
</div>

<p>
 <a href="http://dlmf.nist.gov/Contents/bib/W#wrench:1968:cts">
    Wrench(1968)
 </a> gives exact values of
 <m:math display="inline">
  <m:msub>
   <m:mi>g</m:mi>
   <m:mi>k</m:mi>
  </m:msub>
 </m:math> up to
 <m:math display="inline">
  <m:msub>
   <m:mi>g</m:mi>
   <m:mn>20</m:mn>
  </m:msub>
 </m:math>.
 <a href="http://dlmf.nist.gov/Contents/bib/S#spira:1971:cot">
  Spira(1971)
 </a>
 corrects errors in Wrench's results and also supplies exact and 45D values of
```

```
 <m:math display="inline">
  <m:msub>
   <m:mi>g</m:mi>
   <m:mi>k</m:mi>
  </m:msub>
 </m:math> for
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>21</m:mn>
    <m:mo>,</m:mo>
    <m:mn>22</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
    <m:mo>,</m:mo>
    <m:mn>30</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math>. For an asymptotic expansion of
 <m:math display="inline">
  <m:msub>
   <m:mi>g</m:mi>
   <m:mi>k</m:mi>
  </m:msub>
 </m:math> as
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>&#x2192;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math> see
 <a  href="http://dlmf.nist.gov/Contents/bib/B#boyd:1994:gfa">Boyd(1994)
 </a>.
</p>

<p>With the same conditions
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mi>z</m:mi>
       </m:mrow>
```

```
    <m:mo>+</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>&#x223C;</m:mo>
 <m:mrow>
  <m:msqrt>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>&#x03C0;</m:mi>
   </m:mrow>
  </m:msqrt>
  <m:msup>
   <m:mi mathvariant="normal">&#x2147;</m:mi>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mi>z</m:mi>
    </m:mrow>
   </m:mrow>
  </m:msup>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mi>z</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mfrac bevelled="true">
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:msup>
 </m:mrow>
```

```
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:none/>
     <m:mo>&gt;</m:mo>
     <m:mn>0</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> and
 <m:math display="inline">
  <m:mrow>
   <m:mi>b</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:none/>
     <m:mo>&#x2208;</m:mo>
     <m:mi mathvariant="normal">&#x2102;</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> are both fixed, and
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mi>h</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
```

```
</m:mrow>
<m:mo>&#x223C;</m:mo>
<m:mrow>
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mi>h</m:mi>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mstyle displaystyle="false">
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:mstyle>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mi>ln</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
  </m:mrow>
  <m:mo>-</m:mo>
  <m:mi>z</m:mi>
 </m:mrow>
 <m:mo>+</m:mo>
 <m:mrow>
  <m:mstyle displaystyle="false">
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mn>2</m:mn>
   </m:mfrac>
  </m:mstyle>
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>&#x03C0;</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mo>+</m:mo>
 <m:mrow>
  <m:munderover>
```

```
  <m:mo movablelimits="false">&#x2211;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:munderover>
 <m:mfrac>
  <m:mrow>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mi>k</m:mi>
   </m:msup>
   <m:mrow>
    <m:msub>
     <m:mi>B</m:mi>
     <m:mi>k</m:mi>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>h</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:msup>
    <m:mi>z</m:mi>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:msup>
  </m:mrow>
 </m:mfrac>
```

```
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
   <m:mi>h</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:none/>
      <m:mo>&#x2208;</m:mo>
      <m:mrow>
      <m:mo>[</m:mo>
      <m:mrow>
       <m:mn>0</m:mn>
       <m:mo>,</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>]</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> is fixed.
</p>

<p>Also as
 <m:math display="inline">
  <m:mrow>
   <m:mi>y</m:mi>
   <m:mo>&#x2192;</m:mo>
   <m:mrow>
    <m:mo>&#x00B1;</m:mo>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
```

```
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>x</m:mi>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>y</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>|</m:mo>
</m:mrow>
<m:mo>&#x223C;</m:mo>
<m:mrow>
 <m:msqrt>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mi>&#x03C0;</m:mi>
  </m:mrow>
 </m:msqrt>
 <m:msup>
  <m:mrow>
   <m:mo>|</m:mo>
   <m:mi>y</m:mi>
   <m:mo>|</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mo>-</m:mo>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mfrac bevelled="true">
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:msup>
 <m:msup>
  <m:mi mathvariant="normal">&#x2147;</m:mi>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mfrac bevelled="true">
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mrow>
      <m:mo>|</m:mo>
      <m:mi>y</m:mi>
      <m:mo>|</m:mo>
     </m:mrow>
    </m:mrow>
```

```
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mrow>
     </m:msup>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>uniformly for bounded real values of
 <m:math display="inline">
  <m:mi>x</m:mi>
 </m:math>.
</p>

<h4>Error Bounds and Exponential Improvement</h4>

<p>If the sums in the expansions
(<a href="#equation1">Equation 1</a>) and
(<a href="#equation2">Equation 2</a>) are terminated at
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mi>n</m:mi>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math> (
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>) and
 <m:math display="inline">
  <m:mi>z</m:mi>
 </m:math>
is real and positive, then the remainder terms are bounded in magnitude by
the first neglected terms and have the same sign. If
 <m:math display="inline">
  <m:mi>z</m:mi>
 </m:math>
is complex, then the remainder terms are bounded in magnitude by
 <m:math display="inline">
  <m:mrow>
   <m:msup>
    <m:mi>sec</m:mi>
    <m:mrow>
     <m:mn>2</m:mn>
```

```
     <m:mi>n</m:mi>
    </m:mrow>
   </m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mrow>
       <m:mi>ph</m:mi>
       <m:mspace width="0.2em"/>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:math> for
(<a href="#equation1">Equation 1</a>), and
 <m:math display="inline">
  <m:mrow>
   <m:msup>
    <m:mi>sec</m:mi>
    <m:mrow>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>n</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
     <m:mrow>
      <m:mi>ph</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>z</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> for
(<a href="#equation2">Equation 2</a>), times the first neglected terms.</p>

<p>For the remainder term in
```

```
(<a href="#equation3">Equation 3</a>) write
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msup>
      <m:mi mathvariant="normal">&#x2147;</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:msup>
     <m:msup>
      <m:mi>z</m:mi>
      <m:mi>z</m:mi>
     </m:msup>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mfrac>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>&#x03C0;</m:mi>
        </m:mrow>
        <m:mi>z</m:mi>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mfrac bevelled="true">
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:msup>
     <m:mrow>
       <m:mo>(</m:mo>
      <m:mrow>
        <m:mrow>
        <m:munderover>
         <m:mo movablelimits="false">&#x2211;</m:mo>
          <m:mrow>
          <m:mi>k</m:mi>
          <m:mo>=</m:mo>
```

```
          <m:mn>0</m:mn>
         </m:mrow>
         <m:mrow>
          <m:mi>K</m:mi>
          <m:mo>-</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
        </m:munderover>
        <m:mfrac>
         <m:msub>
          <m:mi>g</m:mi>
          <m:mi>k</m:mi>
          </m:msub>
         <m:msup>
          <m:mi>z</m:mi>
          <m:mi>k</m:mi>
          </m:msup>
        </m:mfrac>
       </m:mrow>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:msub>
         <m:mi>R</m:mi>
         <m:mi>K</m:mi>
        </m:msub>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mi>z</m:mi>
         <m:mo>)</m:mo>
        </m:mrow>
        </m:mrow>
       </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>K</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>,</m:mo>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mn>3</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
```

```
  </m:mrow>
 </m:math>.
</div>

<p>Then
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mrow>
      <m:msub>
       <m:mi>R</m:mi>
       <m:mi>K</m:mi>
      </m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>|</m:mo>
    </m:mrow>
    <m:mo>&#x2264;</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>+</m:mo>
         <m:mrow>
          <m:mi>&#x03B6;</m:mi>
          <m:mrow>
           <m:mo>(</m:mo>
           <m:mi>K</m:mi>
           <m:mo>)</m:mo>
          </m:mrow>
         </m:mrow>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mi>K</m:mi>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
```

```
      </m:mrow>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:msup>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mn>2</m:mn>
          <m:mi>&#x03C0;</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mrow>
         <m:mi>K</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
       <m:msup>
        <m:mrow>
         <m:mo>|</m:mo>
         <m:mi>z</m:mi>
         <m:mo>|</m:mo>
        </m:mrow>
        <m:mi>K</m:mi>
       </m:msup>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:mo movablelimits="false">min</m:mo>
        <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mrow>
            <m:mi>sec</m:mi>
            <m:mrow>
             <m:mo>(</m:mo>
             <m:mrow>
              <m:mi>ph</m:mi>
              <m:mspace width="0.2em"/>
              <m:mi>z</m:mi>
             </m:mrow>
             <m:mo>)</m:mo>
            </m:mrow>
           </m:mrow>
           <m:mo>,</m:mo>
           <m:mrow>
            <m:mn>2</m:mn>
            <m:msup>
```

```
             <m:mi>K</m:mi>
             <m:mstyle scriptlevel="+1">
             <m:mfrac>
              <m:mn>1</m:mn>
              <m:mn>2</m:mn>
             </m:mfrac>
            </m:mstyle>
           </m:msup>
          </m:mrow>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
    <m:mi>&#x03C0;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>


<h4>Ratios</h4>

<p>If
 <m:math display="inline">
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
```

```
   <m:mrow>
    <m:none/>
    <m:mo>&#x2208;</m:mo>
    <m:mi mathvariant="normal">&#x2102;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> and
<m:math display="inline">
 <m:mrow>
  <m:mi>b</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:none/>
    <m:mo>&#x2208;</m:mo>
    <m:mi mathvariant="normal">&#x2102;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> are fixed as
<m:math display="inline">
 <m:mrow>
  <m:mi>z</m:mi>
  <m:mo>&#x2192;</m:mo>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:mrow>
</m:math> in
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mo>|</m:mo>
   <m:mrow>
    <m:mi>ph</m:mi>
    <m:mspace width="0.2em"/>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>|</m:mo>
  </m:mrow>
  <m:mo>&#x2264;</m:mo>
  <m:mrow>
   <m:mrow>
    <m:mi>&#x03C0;</m:mi>
    <m:mo>-</m:mo>
    <m:mi>&#x03B4;</m:mi>
   </m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:none/>
     <m:mo>&lt;</m:mo>
     <m:mi>&#x03C0;</m:mi>
```

```
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>, then
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mfrac>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mi>a</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
    <m:mo>&#x223C;</m:mo>
    <m:msup>
     <m:mi>z</m:mi>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>-</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
    </m:msup>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
```

```
<m:mrow>
 <m:mrow>
  <m:mfrac>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mi>a</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
  <m:mo>&#x223C;</m:mo>
  <m:mrow>
   <m:msup>
    <m:mi>z</m:mi>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>-</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
   </m:msup>
   <m:mrow>
    <m:munderover>
     <m:mo movablelimits="false">&#x2211;</m:mo>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>=</m:mo>
      <m:mn>0</m:mn>
     </m:mrow>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:munderover>
    <m:mfrac>
     <m:mrow>
      <m:msub>
       <m:mi>G</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
```

```
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>,</m:mo>
          <m:mi>b</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
       <m:msup>
        <m:mi>z</m:mi>
        <m:mi>k</m:mi>
       </m:msup>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>Also, with the added condition
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>b</m:mi>
      <m:mo>-</m:mo>
      <m:mi>a</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mfrac>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
```

```
     <m:mo>+</m:mo>
     <m:mi>a</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>z</m:mi>
     <m:mo>+</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mfrac>
 <m:mo>&#x223C;</m:mo>
 <m:mrow>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>z</m:mi>
     <m:mo>+</m:mo>
     <m:mfrac>
      <m:mrow>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mn>2</m:mn>
     </m:mfrac>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>-</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:munderover>
    <m:mo movablelimits="false">&#x2211;</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>=</m:mo>
```

```
  <m:mn>0</m:mn>
 </m:mrow>
 <m:mi mathvariant="normal">&#x221E;</m:mi>
</m:munderover>
<m:mfrac>
 <m:mrow>
  <m:msub>
   <m:mi>H</m:mi>
   <m:mi>k</m:mi>
  </m:msub>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>,</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>z</m:mi>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mi>k</m:mi>
  </m:mrow>
 </m:msup>
</m:mfrac>
```

```
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>Here
</p>

<div align="center">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>G</m:mi>
      <m:mn>0</m:mn>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>G</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
```

```
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>-</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>G</m:mi>
     <m:mn>2</m:mn>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>,</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mstyle displaystyle="true">
     <m:mfrac>
```

```
  <m:mn>1</m:mn>
  <m:mn>12</m:mn>
 </m:mfrac>
</m:mstyle>
<m:mrow>
 <m:mo>(</m:mo>
 <m:mstyle displaystyle="true">
  <m:mtable rowspacing="0.2ex" columnspacing="0.4em">
   <m:mtr>
    <m:mtd>
     <m:mrow>
      <m:mi>a</m:mi>
       <m:mo>-</m:mo>
      <m:mi>b</m:mi>
      </m:mrow>
     </m:mtd>
   </m:mtr>
    <m:mtr>
    <m:mtd>
     <m:mn>2</m:mn>
     </m:mtd>
   </m:mtr>
  </m:mtable>
 </m:mstyle>
 <m:mo>)</m:mo>
</m:mrow>
<m:mrow>
 <m:mo>(</m:mo>
 <m:mrow>
  <m:mrow>
   <m:mn>3</m:mn>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mn>2</m:mn>
   </m:msup>
  </m:mrow>
  <m:mo>-</m:mo>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>a</m:mi>
```

```
          <m:mo>-</m:mo>
          <m:mi>b</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>H</m:mi>
      <m:mn>0</m:mn>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>H</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
```

```
    <m:mo>,</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:mrow>
  <m:mo>-</m:mo>
  <m:mrow>
   <m:mstyle displaystyle="true">
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>12</m:mn>
    </m:mfrac>
   </m:mstyle>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mstyle displaystyle="true">
     <m:mtable rowspacing="0.2ex" columnspacing="0.4em">
      <m:mtr>
       <m:mtd>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>-</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
       </m:mtd>
      </m:mtr>
      <m:mtr>
       <m:mtd>
        <m:mn>2</m:mn>
       </m:mtd>
      </m:mtr>
     </m:mtable>
    </m:mstyle>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>-</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:mrow>
```

```
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>H</m:mi>
     <m:mn>2</m:mn>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>,</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mstyle displaystyle="true">
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mn>240</m:mn>
     </m:mfrac>
    </m:mstyle>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mstyle displaystyle="true">
      <m:mtable rowspacing="0.2ex" columnspacing="0.4em">
      <m:mtr>
       <m:mtd>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>-</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
       </m:mtd>
      </m:mtr>
      <m:mtr>
       <m:mtd>
        <m:mn>4</m:mn>
       </m:mtd>
      </m:mtr>
     </m:mtable>
    </m:mstyle>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
```

```
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>-</m:mo>
          <m:mi>b</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mn>5</m:mn>
       <m:msup>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mrow>
           <m:mi>a</m:mi>
           <m:mo>-</m:mo>
           <m:mi>b</m:mi>
          </m:mrow>
          <m:mo>+</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mn>2</m:mn>
       </m:msup>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>In terms of generalized Bernoulli polynomials we have for
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>k</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>0</m:mn>
     <m:mo>,</m:mo>
```

```
     <m:mn>1</m:mn>
     <m:mo>,</m:mo>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>G</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mtable rowspacing="0.2ex" columnspacing="0.4em">
       <m:mtr>
        <m:mtd>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>-</m:mo>
          <m:mi>b</m:mi>
         </m:mrow>
        </m:mtd>
       </m:mtr>
       <m:mtr>
        <m:mtd>
         <m:mi>k</m:mi>
        </m:mtd>
       </m:mtr>
      </m:mtable>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:msubsup>
       <m:mi>B</m:mi>
       <m:mi>k</m:mi>
```

```
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>-</m:mo>
          <m:mi>b</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:msubsup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>a</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>H</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mtable rowspacing="0.2ex" columnspacing="0.4em">
       <m:mtr>
        <m:mtd>
         <m:mrow>
          <m:mi>a</m:mi>
```

```
     <m:mo>-</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
   </m:mtd>
  </m:mtr>
  <m:mtr>
   <m:mtd>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>k</m:mi>
    </m:mrow>
   </m:mtd>
  </m:mtr>
 </m:mtable>
 <m:mo>)</m:mo>
</m:mrow>
<m:mrow>
 <m:msubsup>
  <m:mi>B</m:mi>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mi>k</m:mi>
  </m:mrow>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>-</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:msubsup>
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mfrac>
   <m:mrow>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>-</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mn>2</m:mn>
  </m:mfrac>
  <m:mo>)</m:mo>
 </m:mrow>
</m:mrow>
```

```
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mfrac>
     <m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>+</m:mo>
         <m:mi>a</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>+</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mi>c</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
    <m:mo>&#x223C;</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
```

```
 <m:mrow>
  <m:mi>k</m:mi>
  <m:mo>=</m:mo>
  <m:mn>0</m:mn>
 </m:mrow>
 <m:mi mathvariant="normal">&#x221E;</m:mi>
</m:munderover>
<m:msup>
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mo>)</m:mo>
 </m:mrow>
 <m:mi>k</m:mi>
</m:msup>
<m:mfrac>
 <m:mrow>
  <m:msub>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>c</m:mi>
     <m:mo>-</m:mo>
     <m:mi>a</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi>k</m:mi>
  </m:msub>
  <m:msub>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>c</m:mi>
     <m:mo>-</m:mo>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi>k</m:mi>
  </m:msub>
 </m:mrow>
 <m:mrow>
  <m:mi>k</m:mi>
  <m:mi mathvariant="normal">!</m:mi>
 </m:mrow>
</m:mfrac>
<m:mrow>
 <m:mi mathvariant="normal">&#x0393;</m:mi>
 <m:mrow>
  <m:mo>(</m:mo>
```

```
        <m:mrow>
         <m:mrow>
          <m:mrow>
           <m:mrow>
            <m:mi>a</m:mi>
            <m:mo>+</m:mo>
            <m:mi>b</m:mi>
           </m:mrow>
           <m:mo>-</m:mo>
           <m:mi>c</m:mi>
          </m:mrow>
          <m:mo>+</m:mo>
          <m:mi>z</m:mi>
         </m:mrow>
         <m:mo>-</m:mo>
         <m:mi>k</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmfbarnesgfunction.xhtml

— dlmfbarnesgfunction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Barnes G-Function (Double Gamma Function)
  </div>
  <hr/>
<h3>Barnes
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">G</m:mi>
 </m:math>-Function (Double Gamma Function)
</h3>

<div align="center">
 <m:math display="inline">
  <m:mrow>
```

```
  <m:mrow>
   <m:mrow>
    <m:mi>G</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mspace width="0.2em"/>
    <m:mrow>
     <m:mi>G</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:mi>G</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mn>1</m:mn>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
```

```
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>G</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>n</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mn>2</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mn>3</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
     <m:mi mathvariant="normal">&#x22EF;</m:mi>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
```

```
   <m:mi>n</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mn>3</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>G</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>&#x03C0;</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mfrac bevelled="true">
        <m:mi>z</m:mi>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:msup>
      <m:mrow>
       <m:mi>exp</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mrow>
```

```
           <m:mstyle displaystyle="false">
            <m:mfrac>
             <m:mn>1</m:mn>
             <m:mn>2</m:mn>
            </m:mfrac>
           </m:mstyle>
           <m:mspace width="0.2em"/>
           <m:mi>z</m:mi>
           <m:mspace width="0.2em"/>
           <m:mrow>
            <m:mo>(</m:mo>
            <m:mrow>
             <m:mi>z</m:mi>
             <m:mo>+</m:mo>
             <m:mn>1</m:mn>
            </m:mrow>
            <m:mo>)</m:mo>
           </m:mrow>
          </m:mrow>
         </m:mrow>
         <m:mo>-</m:mo>
         <m:mrow>
          <m:mstyle displaystyle="false">
           <m:mfrac>
            <m:mn>1</m:mn>
            <m:mn>2</m:mn>
           </m:mfrac>
          </m:mstyle>
          <m:mi>&#x03B3;</m:mi>
          <m:msup>
           <m:mi>z</m:mi>
           <m:mn>2</m:mn>
          </m:msup>
         </m:mrow>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mo>&#x00D7;</m:mo>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x220F;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:munderover>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:msup>
        <m:mrow>
```

```
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mn>1</m:mn>
          <m:mo>+</m:mo>
          <m:mfrac>
           <m:mi>z</m:mi>
           <m:mi>k</m:mi>
          </m:mfrac>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mi>k</m:mi>
       </m:msup>
       <m:mrow>
        <m:mi>exp</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mrow>
           <m:mo>-</m:mo>
           <m:mi>z</m:mi>
          </m:mrow>
          <m:mo>+</m:mo>
          <m:mfrac>
           <m:msup>
            <m:mi>z</m:mi>
            <m:mn>2</m:mn>
           </m:msup>
           <m:mrow>
            <m:mn>2</m:mn>
            <m:mi>k</m:mi>
           </m:mrow>
          </m:mfrac>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>Ln</m:mi>
     <m:mspace width="0.2em"/>
     <m:mrow>
```

```
      <m:mi>G</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mspace width="0.2em"/>
        <m:mi>z</m:mi>
        <m:mspace width="0.2em"/>
        <m:mrow>
         <m:mi>ln</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mn>2</m:mn>
           <m:mi>&#x03C0;</m:mi>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mspace width="0.2em"/>
        <m:mi>z</m:mi>
        <m:mspace width="0.2em"/>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>z</m:mi>
          <m:mo>+</m:mo>
```

```
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mo>+</m:mo>
 <m:mrow>
  <m:mi>z</m:mi>
  <m:mspace width="0.2em"/>
  <m:mrow>
   <m:mi>Ln</m:mi>
   <m:mspace width="0.2em"/>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>-</m:mo>
<m:mrow>
 <m:msubsup>
  <m:mo>&#x222B;</m:mo>
  <m:mn>0</m:mn>
  <m:mi>z</m:mi>
 </m:msubsup>
 <m:mrow>
  <m:mi>Ln</m:mi>
  <m:mspace width="0.2em"/>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>t</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mspace width="0.2em"/>
 <m:mrow>
  <m:mi mathvariant="normal">&#x2146;</m:mi>
```

```
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>The
 <m:math display="inline">
  <m:mi>Ln</m:mi>
 </m:math>'s have their principal values on the positive real axis and are
             continued via continuity.
</p>

<p>When
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2192;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math> in
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mo>-</m:mo>
     <m:mi>&#x03B4;</m:mi>
    </m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:none/>
      <m:mo>&lt;</m:mo>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
```

```
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>Ln</m:mi>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi>G</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>&#x223C;</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>4</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:msup>
         <m:mi>z</m:mi>
         <m:mn>2</m:mn>
        </m:msup>
       </m:mrow>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mspace width="0.2em"/>
        <m:mrow>
         <m:mi mathvariant="normal">&#x0393;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>z</m:mi>
           <m:mo>+</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
```

```
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mstyle displaystyle="false">
          <m:mfrac>
           <m:mn>1</m:mn>
           <m:mn>2</m:mn>
          </m:mfrac>
         </m:mstyle>
         <m:mspace width="0.2em"/>
         <m:mi>z</m:mi>
         <m:mspace width="0.2em"/>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>z</m:mi>
           <m:mo>+</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>+</m:mo>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>12</m:mn>
         </m:mfrac>
        </m:mstyle>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mspace width="0.2em"/>
      <m:mrow>
       <m:mi>Ln</m:mi>
       <m:mspace width="0.2em"/>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mi>ln</m:mi>
       <m:mi>A</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
```

```
 <m:mrow>
  <m:mi>k</m:mi>
  <m:mo>=</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
 <m:mi mathvariant="normal">&#x221E;</m:mi>
</m:munderover>
<m:mfrac>
 <m:msub>
  <m:mi>B</m:mi>
  <m:mrow>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>k</m:mi>
   </m:mrow>
   <m:mo>+</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:msub>
 <m:mrow>
  <m:mn>2</m:mn>
  <m:mi>k</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mn>2</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:msup>
   <m:mi>z</m:mi>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>k</m:mi>
   </m:mrow>
  </m:msup>
 </m:mrow>
```

```
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   </m:mrow>
 </m:math>
</div>

<p>see
<a href="http://dlmf.nist.gov/Contents/bib/F#ferreira:2001:aae">
 Ferreira and Lpez(2001)
</a>. This reference also provides bounds for the error term. Here
 <m:math display="inline">
  <m:msub>
   <m:mi>B</m:mi>
   <m:mrow>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mn>2</m:mn>
   </m:mrow>
  </m:msub>
 </m:math> is the Bernoulli number,  and
 <m:math display="inline">
  <m:mi>A</m:mi>
 </m:math> is <em>Glaisher’s constant</em>, given by
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mi>A</m:mi>
    <m:mo>=</m:mo>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mi>C</m:mi>
    </m:msup>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>1.28242 71291 00622 63687</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where
</p>

<div align="center">
```

```
<m:math display="block">
 <m:mrow>
  <m:mrow>
   <m:mi>C</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:munder>
     <m:mo movablelimits="false">lim</m:mo>
     <m:mrow>
      <m:mi>n</m:mi>
      <m:mo>&#x2192;</m:mo>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:mrow>
    </m:munder>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mrow>
        <m:munderover>
         <m:mo movablelimits="false">&#x2211;</m:mo>
         <m:mrow>
          <m:mi>k</m:mi>
          <m:mo>=</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
         <m:mi>n</m:mi>
        </m:munderover>
        <m:mi>k</m:mi>
        <m:mspace width="0.2em"/>
        <m:mrow>
         <m:mi>ln</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi>k</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mrow>
           <m:mstyle displaystyle="false">
            <m:mfrac>
             <m:mn>1</m:mn>
             <m:mn>2</m:mn>
            </m:mfrac>
           </m:mstyle>
           <m:msup>
            <m:mi>n</m:mi>
            <m:mn>2</m:mn>
           </m:msup>
          </m:mrow>
          <m:mo>+</m:mo>
```

```
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mi>n</m:mi>
       </m:mrow>
       <m:mo>+</m:mo>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>12</m:mn>
        </m:mfrac>
       </m:mstyle>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi>ln</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>n</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   <m:mo>+</m:mo>
   <m:mrow>
    <m:mstyle displaystyle="false">
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mn>4</m:mn>
     </m:mfrac>
    </m:mstyle>
    <m:msup>
     <m:mi>n</m:mi>
     <m:mn>2</m:mn>
    </m:msup>
   </m:mrow>
  </m:mrow>
  <m:mo>)</m:mo>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mfrac>
  <m:mrow>
   <m:mi>&#x03B3;</m:mi>
   <m:mo>+</m:mo>
   <m:mrow>
    <m:mi>ln</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
```

```
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mn>12</m:mn>
 </m:mfrac>
 <m:mo>-</m:mo>
 <m:mfrac>
  <m:mrow>
   <m:msup>
    <m:mi>&#x03B6;</m:mi>
    <m:mo>&#x2032;</m:mo>
   </m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mn>2</m:mn>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:msup>
    <m:mi>&#x03C0;</m:mi>
    <m:mn>2</m:mn>
   </m:msup>
  </m:mrow>
 </m:mfrac>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mfrac>
  <m:mn>1</m:mn>
  <m:mn>12</m:mn>
 </m:mfrac>
 <m:mo>-</m:mo>
 <m:mrow>
  <m:msup>
   <m:mi>&#x03B6;</m:mi>
   <m:mo>&#x2032;</m:mo>
  </m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:mrow>
```

```
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>and
 <m:math display="inline">
  <m:msup>
   <m:mi>&#x03B6;</m:mi>
   <m:mo>&#x2032;</m:mo>
  </m:msup>
 </m:math> is the derivative of the zeta function
</p>

<p>For Glaisher's constant see also
 <a href="http://dlmf.nist.gov/Contents/bib/G#greene:1982:mft">
  Greene and Knuth(1982)
 </a>(p.100).
</p>
\getchunk{page foot}
```

dlmfbetafunction.xhtml

— dlmfbetafunction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Beta Function
  </div>
  <hr/>
<h3>Beta Function</h3>

<p>In this section all fractional powers have their principal values, except
where noted otherwise. In the next 4 equations it is assumed
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math> and
 <m:math display="inline">
```

```
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>.
</p>

<h5>Euler's Beta Integral</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">B</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mn>1</m:mn>
     </m:msubsup>
     <m:msup>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mi>t</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
```

```
      <m:mrow>
       <m:mi>b</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>a</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>b</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
```

```
 <m:msubsup>
  <m:mo>&#x222B;</m:mo>
  <m:mn>0</m:mn>
  <m:mfrac bevelled="true">
   <m:mi>&#x03C0;</m:mi>
   <m:mn>2</m:mn>
  </m:mfrac>
 </m:msubsup>
 <m:mrow>
  <m:msup>
   <m:mi>sin</m:mi>
   <m:mrow>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>a</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:msup>
  <m:mi>&#x03B8;</m:mi>
 </m:mrow>
 <m:mrow>
  <m:msup>
   <m:mi>cos</m:mi>
   <m:mrow>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:msup>
  <m:mi>&#x03B8;</m:mi>
 </m:mrow>
 <m:mrow>
  <m:mi mathvariant="normal">&#x2146;</m:mi>
  <m:mi>&#x03B8;</m:mi>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mstyle displaystyle="false">
  <m:mfrac>
   <m:mn>1</m:mn>
   <m:mn>2</m:mn>
  </m:mfrac>
 </m:mstyle>
 <m:mrow>
  <m:mi mathvariant="normal">B</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
```

```
        <m:mi>a</m:mi>
        <m:mo>,</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:msup>
        <m:mi>t</m:mi>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2146;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:mrow>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>+</m:mo>
         <m:mi>t</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
      </m:msup>
```

```
     </m:mfrac>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">B</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mn>1</m:mn>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:msup>
        <m:mi>t</m:mi>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
       <m:msup>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mn>1</m:mn>
          <m:mo>-</m:mo>
          <m:mi>t</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mrow>
         <m:mi>b</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
```

```
    </m:mrow>
   </m:msup>
  </m:mrow>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>t</m:mi>
     <m:mo>+</m:mo>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>+</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
  </m:msup>
 </m:mfrac>
 <m:mrow>
  <m:mi mathvariant="normal">&#x2146;</m:mi>
  <m:mi>t</m:mi>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mrow>
  <m:mi mathvariant="normal">B</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>,</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>+</m:mo>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi>a</m:mi>
  </m:mrow>
 </m:msup>
```

```
     <m:msup>
      <m:mi>z</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
     </m:msup>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>with
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mi>&#x03C0;</m:mi>
  </m:mrow>
 </m:math> and the integration path along the real axis.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mfrac bevelled="true">
       <m:mi>&#x03C0;</m:mi>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:msubsup>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>cos</m:mi>
        <m:mspace width="0.2em"/>
        <m:mi>t</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
```

```
     <m:mi>a</m:mi>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:msup>
   <m:mrow>
    <m:mi>cos</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>b</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x2146;</m:mi>
    <m:mi>t</m:mi>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mfrac>
    <m:mi>&#x03C0;</m:mi>
    <m:msup>
     <m:mn>2</m:mn>
     <m:mi>a</m:mi>
    </m:msup>
   </m:mfrac>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mrow>
      <m:mi mathvariant="normal">B</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>a</m:mi>
           <m:mo>+</m:mo>
           <m:mi>b</m:mi>
           <m:mo>+</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
```

```
           </m:mrow>
          </m:mrow>
          <m:mo>,</m:mo>
          <m:mrow>
           <m:mfrac>
            <m:mn>1</m:mn>
            <m:mn>2</m:mn>
           </m:mfrac>
           <m:mrow>
            <m:mo>(</m:mo>
            <m:mrow>
             <m:mrow>
              <m:mi>a</m:mi>
              <m:mo>-</m:mo>
              <m:mi>b</m:mi>
             </m:mrow>
             <m:mo>+</m:mo>
             <m:mn>1</m:mn>
            </m:mrow>
            <m:mo>)</m:mo>
           </m:mrow>
          </m:mrow>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
```

```
    <m:mn>0</m:mn>
    <m:mi>&#x03C0;</m:mi>
   </m:msubsup>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>sin</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:msup>
   <m:msup>
    <m:mi mathvariant="normal">&#x2147;</m:mi>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>b</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:msup>
   <m:mrow>
    <m:mi mathvariant="normal">&#x2146;</m:mi>
    <m:mi>t</m:mi>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mfrac>
    <m:mi>&#x03C0;</m:mi>
    <m:msup>
     <m:mn>2</m:mn>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
    </m:msup>
   </m:mfrac>
   <m:mfrac>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mfrac bevelled="true">
      <m:mrow>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mi>&#x03C0;</m:mi>
       <m:mi>b</m:mi>
      </m:mrow>
```

```
    <m:mn>2</m:mn>
   </m:mfrac>
  </m:msup>
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mrow>
    <m:mi mathvariant="normal">B</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>+</m:mo>
         <m:mi>b</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mo>,</m:mo>
      <m:mrow>
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>-</m:mo>
          <m:mi>b</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mfrac>
</m:mrow>
```

```
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:mi>cosh</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>b</m:mi>
         <m:mi>t</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>cosh</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi>t</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>a</m:mi>
```

```
       </m:mrow>
      </m:msup>
     </m:mfrac>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msup>
      <m:mn>4</m:mn>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">B</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>+</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
        <m:mo>,</m:mo>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>-</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mrow>
```

```
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x211C;</m:mi>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi mathvariant="normal">&#x221E;</m:mi>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:msubsup>
      <m:mfrac>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2146;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
       <m:mrow>
        <m:msup>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>w</m:mi>
           <m:mo>+</m:mo>
           <m:mrow>
            <m:mi mathvariant="normal">&#x2148;</m:mi>
            <m:mi>t</m:mi>
           </m:mrow>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
         <m:mi>a</m:mi>
        </m:msup>
        <m:msup>
```

```
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>z</m:mi>
           <m:mo>-</m:mo>
           <m:mrow>
            <m:mi mathvariant="normal">&#x2148;</m:mi>
            <m:mi>t</m:mi>
           </m:mrow>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
         <m:mi>b</m:mi>
        </m:msup>
       </m:mrow>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>w</m:mi>
        <m:mo>+</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:mi>a</m:mi>
       <m:mo>-</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>+</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">B</m:mi>
```

```
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>,</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mfrac>
  </m:mrow>
 </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>w</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
```

```
  </m:mrow>
 </m:math>
</div>

<p>The fractional powers have their principal values when
 <m:math display="inline">
  <m:mrow>
   <m:mi>w</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math> and
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>, and are continued via continuity.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mrow>
        <m:mi>c</m:mi>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x221E;</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mrow>
        <m:mi>c</m:mi>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x221E;</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
        </m:mrow>
```

```
      </m:mrow>
     </m:msubsup>
     <m:msup>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>a</m:mi>
      </m:mrow>
     </m:msup>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mi>t</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mrow>
     <m:mi>b</m:mi>
     <m:mrow>
      <m:mi mathvariant="normal">B</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>,</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mfrac>
  </m:mrow>
```

```
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>c</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mn>0</m:mn>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
```

```
     <m:mn>1</m:mn>
     <m:mo>+</m:mo>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:msubsup>
  <m:msup>
   <m:mi>t</m:mi>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:msup>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>t</m:mi>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mi>b</m:mi>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:mi>t</m:mi>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mfrac>
  <m:mrow>
   <m:mi>sin</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mi>b</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mi>&#x03C0;</m:mi>
 </m:mfrac>
 <m:mrow>
  <m:mi mathvariant="normal">B</m:mi>
```

```
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>,</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <!-- Need a better Axiom graphic for this
 <img width="302" height="151" alt="" src="bitmaps/12F1.png"/> -->
</div>

<div align="center">
 <m:math display="inline">
  <m:mi>t</m:mi>
 </m:math>-plane. Contour for first loop integral for the beta function.
</div>

<p>In the next two equations the fractional powers are continuous on the
   integration paths and take their principal values at the beginning.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:msup>
        <m:mi mathvariant="normal">&#x2147;</m:mi>
```

```
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>&#x03C0;</m:mi>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>a</m:mi>
    </m:mrow>
   </m:msup>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:mfrac>
 <m:mrow>
  <m:msubsup>
   <m:mo>&#x222B;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>0</m:mn>
     <m:mo>+</m:mo>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:msubsup>
  <m:msup>
   <m:mi>t</m:mi>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:msup>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>1</m:mn>
     <m:mo>+</m:mo>
     <m:mi>t</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>a</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:mi>t</m:mi>
```

```
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">B</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p> when
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math>
  <m:mi>a</m:mi>
 </m:math> is not an integer and the contour cuts the real axis between
 <m:math>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math> and the origin.
</p>

<div align="center">
 <!-- Need a better Axiom graphic for this
 <img width="302" height="151" alt="" src="bitmaps/12F2.png"/> -->
</div>

<div align="center">
 <m:math display="inline">
  <m:mi>t</m:mi>
 </m:math>-plane. Contour for second loop integral for the beta function.
</div>
```

```
<h5>Pochhammer's Integral</h5>
<p>When
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>,</m:mo>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>&#x2208;</m:mo>
   <m:mi mathvariant="normal">&#x2102;</m:mi>
  </m:mrow>
 </m:math>
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mi>P</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>+</m:mo>
        </m:mrow>
        <m:mo>,</m:mo>
        <m:mrow>
         <m:mn>0</m:mn>
         <m:mo>+</m:mo>
        </m:mrow>
        <m:mo>,</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>-</m:mo>
        </m:mrow>
        <m:mo>,</m:mo>
        <m:mrow>
         <m:mn>0</m:mn>
         <m:mo>-</m:mo>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:msubsup>
     <m:msup>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>-</m:mo>
```

```
   <m:mn>1</m:mn>
  </m:mrow>
 </m:msup>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>-</m:mo>
    <m:mi>t</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mi>b</m:mi>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:msup>
 <m:mrow>
  <m:mi mathvariant="normal">&#x2146;</m:mi>
  <m:mi>t</m:mi>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mo>-</m:mo>
 <m:mrow>
  <m:mn>4</m:mn>
  <m:msup>
   <m:mi mathvariant="normal">&#x2147;</m:mi>
   <m:mrow>
    <m:mi>&#x03C0;</m:mi>
    <m:mspace width="0.2em"/>
    <m:mi mathvariant="normal">&#x2148;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:mi>sin</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mi>a</m:mi>
    </m:mrow>
```

```
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi>sin</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>&#x03C0;</m:mi>
         <m:mi>b</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">B</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>,</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where the contour starts from an arbitrary point
 <m:math display="inline">
  <m:mi>P</m:mi>
 </m:math> in the interval
 <m:math display="inline">
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:math>,circles
 <m:math display="inline">
  <m:mn>1</m:mn>
 </m:math> and then
 <m:math display="inline">
  <m:mn>0</m:mn>
 </m:math> in the positive sense, circles
```

```
 <m:math display="inline">
  <m:mn>1</m:mn>
 </m:math> and then
 <m:math display="inline">
  <m:mn>0</m:mn>
 </m:math> in the negative sense, and returns to
 <m:math display="inline">
  <m:mi>P</m:mi>
 </m:math>. It can always be deformed into the contour shown here.
</p>

<div align="center">
 <!-- Need a better Axiom graphic for this
 <img width="302" height="104" alt="" src="bitmaps/12F3.png"/> -->
</div>

<div align="center">
 <m:math display="inline">
  <m:mi>t</m:mi>
 </m:math>-plane. Contour for Pochhammer's integral.
</div>
\getchunk{page foot}
```

dlmfcontinuedfractions.xhtml

— dlmfcontinuedfractions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Continued Fractions
  </div>
  <hr/>
<h3>Continued Fractions</h3>

<p>For
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
```

```
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mi>ln</m:mi>
       <m:mrow>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mi>z</m:mi>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
      <m:mo>+</m:mo>
      <m:mi>z</m:mi>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>-</m:mo>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mi>ln</m:mi>
       <m:mspace width="0.2em"/>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mrow>
       <m:mi>ln</m:mi>
```

```
   <m:mspace width="0.2em"/>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>&#x03C0;</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mrow>
  <m:mfrac>
   <m:msub>
    <m:mi>a</m:mi>
    <m:mn>0</m:mn>
   </m:msub>
   <m:mrow>
    <m:mrow>
     <m:mo>+</m:mo>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
   </m:mrow>
  </m:mfrac>
  <m:mrow>
   <m:mfrac>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mn>1</m:mn>
    </m:msub>
    <m:mrow>
     <m:mrow>
      <m:mo>+</m:mo>
      <m:mi>z</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
    </m:mrow>
   </m:mfrac>
   <m:mrow>
    <m:mfrac>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mn>2</m:mn>
     </m:msub>
     <m:mrow>
      <m:mrow>
       <m:mo>+</m:mo>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>+</m:mo>
```

```
         </m:mrow>
        </m:mfrac>
        <m:mrow>
         <m:mfrac>
          <m:msub>
           <m:mi>a</m:mi>
           <m:mn>3</m:mn>
          </m:msub>
          <m:mrow>
           <m:mrow>
            <m:mo>+</m:mo>
            <m:mi>z</m:mi>
           </m:mrow>
           <m:mo>+</m:mo>
          </m:mrow>
         </m:mfrac>
         <m:mrow>
          <m:mfrac>
           <m:msub>
            <m:mi>a</m:mi>
            <m:mn>4</m:mn>
           </m:msub>
           <m:mrow>
            <m:mrow>
             <m:mo>+</m:mo>
             <m:mi>z</m:mi>
            </m:mrow>
            <m:mo>+</m:mo>
           </m:mrow>
          </m:mfrac>
          <m:mfrac>
           <m:msub>
            <m:mi>a</m:mi>
            <m:mn>5</m:mn>
           </m:msub>
           <m:mrow>
            <m:mi>z</m:mi>
            <m:mo>+</m:mo>
           </m:mrow>
          </m:mfrac>
         </m:mrow>
        </m:mrow>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mi mathvariant="normal">&#x22EF;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="inline">
```

```
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mn>0</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>12</m:mn>
    </m:mfrac>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mn>1</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>30</m:mn>
    </m:mfrac>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mn>2</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mn>53</m:mn>
     <m:mn>210</m:mn>
    </m:mfrac>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mn>3</m:mn>
    </m:msub>
```

```
   <m:mo>=</m:mo>
   <m:mfrac>
    <m:mn>195</m:mn>
    <m:mn>371</m:mn>
   </m:mfrac>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:msub>
    <m:mi>a</m:mi>
    <m:mn>4</m:mn>
   </m:msub>
   <m:mo>=</m:mo>
   <m:mfrac>
    <m:mn>22999</m:mn>
    <m:mn>22737</m:mn>
   </m:mfrac>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:msub>
    <m:mi>a</m:mi>
    <m:mn>5</m:mn>
   </m:msub>
   <m:mo>=</m:mo>
   <m:mfrac>
    <m:mn>299 44523</m:mn>
    <m:mn>197 33142</m:mn>
   </m:mfrac>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:msub>
    <m:mi>a</m:mi>
    <m:mn>6</m:mn>
   </m:msub>
   <m:mo>=</m:mo>
   <m:mfrac>
    <m:mn>10 95352 41009</m:mn>
    <m:mn>4 82642 75462</m:mn>
   </m:mfrac>
  </m:mrow>
```

```
  </m:mrow>
 </m:math>
</div>

<p>For rational values of
 <m:math display="inline">
  <m:msub>
   <m:mi>a</m:mi>
   <m:mn>7</m:mn>
  </m:msub>
 </m:math> to
 <m:math display="inline">
  <m:msub>
   <m:mi>a</m:mi>
   <m:mn>11</m:mn>
  </m:msub>
 </m:math> and 40S values of
 <m:math display="inline">
  <m:msub>
   <m:mi>a</m:mi>
   <m:mn>0</m:mn>
  </m:msub>
 </m:math> to
 <m:math display="inline">
  <m:msub>
   <m:mi>a</m:mi>
   <m:mn>40</m:mn>
  </m:msub>
 </m:math>, see
<a href="http://dlmf.nist.gov/Contents/bib/C#char:1980:osc">
   Char(1980)
</a>. Also see
<a href="http://dlmf.nist.gov/Contents/bib/J#jones:1980:con">
   Jones and Thron(1980)
</a>(pp.348350) and
<a href="http://dlmf.nist.gov/Contents/bib/L#lorentzen:1992:cfa">
   Lorentzen and Waadeland(1992)
</a>(pp.221224) for further information.
</p>
\getchunk{page foot}
```

———-

dlmfdefinitions.xhtml

— dlmfdefinitions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
```

```
   Digital Library of Mathematical Functions
  </a><br/>
  The Gamma Function -- Definitions
 </div>
 <hr/>
<h3>Definitions</h3>
<h6>Contents</h6>
<ul>
 <li>Gamma and Psi Functions</li>
 <li>Euler's Constant</li>
 <li>Pochhammer's Symbol</li>
</ul>
<h4>Gamma and Psi Functions</h4>
<h5>Euler's Integral</h5>
<m:math display="block">
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x00393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:msubsup>
     <m:mo>&#x222B;</m:mo>
     <m:mn>0</m:mn>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:msubsup>
    <m:msup>
     <m:mi mathvariant="normal">&#x02147;</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:msup>
    <m:msup>
     <m:mi>t</m:mi>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
    </m:msup>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
```

```
</m:math>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
    <m:mn>0</m:mn>
  </m:mrow>
 </m:math>.
</div>

When
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x211C;</m:mi>
   <m:mi>z</m:mi>
  </m:mrow>
  <m:mo>&#x2264;</m:mo>
  <m:mn>0</m:mn>
 </m:mrow>
</m:math>,

<m:math display="inline">
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>z</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> is defined by analytic continuation. It is a meromorphic
          function with no zeros, and with simple poles of residue

<m:math display="inline">
 <m:mfrac bevelled="true">
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi>n</m:mi>
  </m:msup>
  <m:mrow>
   <m:mi>n</m:mi>
```

```
   <m:mi mathvariant="normal">!</m:mi>
  </m:mrow>
 </m:mfrac>
</m:math> at

<m:math display="inline">
 <m:mrow>
  <m:mi>z</m:mi>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi>n</m:mi>
  </m:mrow>
 </m:mrow>
</m:math>.

<m:math display="inline">
 <m:mfrac bevelled="true">
  <m:mn>1</m:mn>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mfrac>
</m:math> is entire, with simple zeros at

<m:math display="inline">
 <m:mrow>
  <m:mi>z</m:mi>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi>n</m:mi>
  </m:mrow>
 </m:mrow>
</m:math>.

<p>
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
```

```
    <m:mfrac bevelled="true">
     <m:mrow>
      <m:msup>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mo>&#x2032;</m:mo>
      </m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
</p>

<p>
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03C8;</m:mi>
```

```
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> is meromorphic with simple poles of residue
 <m:math display="inline">
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math> at
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mi>n</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>.
</p>

<h4>Euler’s Constant</h4>
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mi>&#x03B3;</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munder>
      <m:mo movablelimits="false">lim</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>&#x2192;</m:mo>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:mrow>
     </m:munder>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>3</m:mn>
```

```
        </m:mfrac>
        <m:mo>+</m:mo>
        <m:mi mathvariant="normal">&#x2026;</m:mi>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mi>n</m:mi>
        </m:mfrac>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mi>ln</m:mi>
        <m:mi>n</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>0.57721 56649 01532 86060</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>

<h4>Pochhammer's Symbol</h4>
<div align="center">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>a</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mn>0</m:mn>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math>

 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>a</m:mi>
      <m:mo>)</m:mo>
```

```
     </m:mrow>
     <m:mi>n</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mn>2</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi mathvariant="normal">&#x22EF;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>n</m:mi>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>a</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi>n</m:mi>
```

```
    </m:msub>
    <m:mo>=</m:mo>
    <m:mfrac bevelled="true">
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>n</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>a</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:math>

<div align="right">
  <m:math display="inline">
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>&#x2260;</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>n</m:mi>
     </m:mrow>
     <m:mo>,</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>n</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>,</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>n</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
```

```
      <m:mn>2</m:mn>
     </m:mrow>
     <m:mo>,</m:mo>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:math>
 </div>
\getchunk{page foot}
```

dlmffunctionrelations.xhtml

— dlmffunctionrelations.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Function Relations
  </div>
  <hr/>
<h3>Functional Relations</h3>
<h6>Contents</h6>
<ul>
 <li>Recurrence</li>
 <li>Reflection</li>
 <li>Multiplication</li>
 <li>Bohr-Mollerup Theorem</li>
</ul>
<h4>Recurrence</h4>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
       </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
```

```
    <m:mrow>
     <m:mi>z</m:mi>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mi>z</m:mi>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
```

```
<h4>Reflection</h4>
<a name="equation3"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac bevelled="true">
     <m:mi>&#x03C0;</m:mi>
     <m:mrow>
      <m:mi>sin</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>&#x03C0;</m:mi>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
```

```
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>&#x0177;</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac bevelled="true">
      <m:mi>&#x03C0;</m:mi>
      <m:mrow>
       <m:mi>tan</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>&#x03C0;</m:mi>
         <m:mi>z</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
```

```
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>&#x0177;</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>

<h4>Multiplication</h4>
<div align="left">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
```

```
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msup>
      <m:mi>&#x03C0;</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mfrac bevelled="true">
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mrow>
     </m:msup>
     <m:msup>
      <m:mn>2</m:mn>
      <m:mrow>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
```

```
        <m:mo>+</m:mo>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="left">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mn>3</m:mn>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>3</m:mn>
```

```
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:mrow>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mi>&#x03C0;</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:msup>
  <m:msup>
   <m:mn>3</m:mn>
   <m:mrow>
    <m:mrow>
     <m:mn>3</m:mn>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mfrac bevelled="true">
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>z</m:mi>
     <m:mo>+</m:mo>
```

```
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>3</m:mn>
         </m:mfrac>
        </m:mstyle>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>2</m:mn>
          <m:mn>3</m:mn>
         </m:mfrac>
        </m:mstyle>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="left">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>n</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
```

```
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>&#x03C0;</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mfrac bevelled="true">
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>-</m:mo>
         <m:mi>n</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:msup>
     <m:msup>
      <m:mi>n</m:mi>
      <m:mrow>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mrow>
```

```
        <m:mo>(</m:mo>
        <m:mfrac bevelled="true">
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x220F;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>0</m:mn>
       </m:mrow>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:munderover>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mi>k</m:mi>
         <m:mi>n</m:mi>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x220F;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
```

```
    </m:mrow>
    <m:mrow>
     <m:mi>n</m:mi>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:munderover>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mfrac>
     <m:mi>k</m:mi>
     <m:mi>n</m:mi>
    </m:mfrac>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mfrac bevelled="true">
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mn>2</m:mn>
    </m:mfrac>
   </m:msup>
   <m:msup>
    <m:mi>n</m:mi>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac bevelled="true">
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
    </m:mrow>
   </m:msup>
  </m:mrow>
 </m:mrow>
</m:mrow>
```

```
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>&#x03C8;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mi>z</m:mi>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi>&#x03C8;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>z</m:mi>
           <m:mo>+</m:mo>
           <m:mstyle displaystyle="false">
            <m:mfrac>
             <m:mn>1</m:mn>
             <m:mn>2</m:mn>
            </m:mfrac>
           </m:mstyle>
          </m:mrow>
          <m:mo>)</m:mo>
```

```
         </m:mrow>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mi>ln</m:mi>
      <m:mn>2</m:mn>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mi>n</m:mi>
      </m:mfrac>
      <m:mrow>
       <m:munderover>
        <m:mo movablelimits="false">&#x2211;</m:mo>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>=</m:mo>
         <m:mn>0</m:mn>
        </m:mrow>
        <m:mrow>
         <m:mi>n</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:munderover>
       <m:mi>&#x03C8;</m:mi>
```

```
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mi>k</m:mi>
         <m:mi>n</m:mi>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mi>n</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 </m:math>
</div>

<a name="bohrmolleruptheorem"/>
<h4>Bohr-Mollerup Theorem</h4>

<br/>
If a positive function
<m:math display="inline">
 <m:mrow>
  <m:mi>f</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>x</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> on
<m:math display="inline">
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>,</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
  <m:mo>)</m:mo>
 </m:mrow>
</m:math> satisfies
<m:math display="inline">
 <m:mrow>
  <m:mrow>
```

```
   <m:mi>f</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>x</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mrow>
    <m:mi>f</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>x</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>,

<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mi>f</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mn>1</m:mn>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
</m:math>, and

<m:math display="inline">
 <m:mrow>
  <m:mi>ln</m:mi>
  <m:mrow>
   <m:mi>f</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math> is convex, then
```

```
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mi>f</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>.
\getchunk{page foot}
```

dlmfgraphics.xhtml

— dlmfgraphics.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Graphics
  </div>
  <hr/>
<h3>Graphics</h3>
<h6>Contents</h6>
<ul>
 <li>Real Argument</li>
 <li>The Psi Function</li>
 <li>Complex Argument</li>
</ul>
<h4>Real Argument</h4>
 <img width="403" height="482" src="bitmaps/gammareal3.png"/>
 <br/>
This graph shows the
 <m:math display="inline">
```

```
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> and
 <m:math display="inline">
  <m:mfrac bevelled="true">
   <m:mn>1</m:mn>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>x</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
 </m:math>.

To create these two graphs in Axiom:
<pre>
 -- Draw the first graph in a viewport
 viewport1:=draw(Gamma(i), i=-4.2..4, adaptive==true, unit==[1.0,1.0])
 -- Draw the second graph in a viewport
 viewport2:=draw(1/Gamma(i), i=-4.2..4, adaptive==true, unit==[1.0,1.0])
 -- Get the Gamma graph from the first viewport and layer it on top
 putGraph(viewport2,getGraph(viewport1,1),2)
 -- Remove the points and leave the lines
 points(viewport2,1,"off")
 points(viewport2,2,"off")
 -- Show the combined graph
 makeViewport2D(viewport2)
</pre>

 <img width="300" height="176" alt="" src="bitmaps/loggamma.png"/>
 <br/>
 <m:math display="inline">
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>x</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>. This function is convex on
 <m:math display="inline">
```

```
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:math>;
 <br/>
  compare <a href="dlmffunctionalrelations.xhtml#bohrmolleruptheorem">
           Functional Relations</a>
 <p>
You can construct this graph with the Axiom commands:
<pre>
  -- draw the graph of log(Gamma) in a viewport
  viewport1:=draw(log Gamma(i), i=0..8, adaptive==true, unit==[1.0,1.0])
  -- turn off the points and leave the lines
  points(viewport1,1,"off")
</pre>
</p>
 <br/>

 <h4>The Psi Function
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03C8;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>
</h4>

<p> This function is a special case of the polygamma function.
In particular,
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03C8;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>x</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> is equal to polygamma(0,x).
 </p>
 <br/>
 <br/>

 <img width="522" height="556" alt="" src="bitmaps/psi.png"/>
```

```
 <br/>
You can reconstruct this graph in Axiom by:
<pre>
  -- first construct the psi function
  psi(x)==polygamma(0,x)
  -- draw the graph in a viewport
  viewport:=draw(psi(y),y=-3.5..4,adaptive==true)
  -- make the gradient obvious
  scale(viewport,1,0.9,22.5)
  -- and recenter the graph
  translate(viewport,1,0,-0.02)
  -- turn off the points and keep the line
  points(viewport,1,"off")
</pre>

 <h4>Complex Argument</h4>

 <img width="400" height="400" alt="" src="bitmaps/gammacomplex.png"/>
 <br/>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mi>y</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>.
 <br/>

You can reconstruct this image in Axiom with:
<pre>
  -- Set up the default viewpoint
  viewPhiDefault(-%pi/4)
  -- define the point set function
  gam(x,y)==
    g:=Gamma complex(x,y)
    point [x,y,max(min(real g,4),-4), argument g]
  -- draw the image and remember the viewport
  viewport:=draw(gam, -4..4,-3..3,var1Steps==100,var2Steps==100)
  -- set the color mapping for the image
  colorDef(viewport,blue(),blue())
  -- and smoothly shade it
  drawStyle(viewport,"smooth")
```

```
</pre>
 <img width="400" height="400" src="bitmaps/gammacomplexinverse.png"/>
<br/>
 <m:math display="inline">
  <m:mfrac bevelled="true">
   <m:mn>1</m:mn>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>x</m:mi>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>y</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
 </m:math>
 <br/>

<p>
You can reproduce this image from Axiom with:
<pre>
  -- Set up the default viewpoint
  viewPhiDefault(-%pi/4)
  -- Define the complex Gamma inverse function
  gaminv(x,y)==
    g:=1/(Gamma complex(x,y))
    point [x,y,max(min(real g,4),-4), argument g]
  -- draw the 3D image and remember the viewport
  viewport:=draw(gaminv, -4..4,-3..3,var1Steps==100,var2Steps==100)
  -- make the image a uniform color
  colorDef(viewport,blue(),blue())
  -- and make it pretty
  drawStyle(viewport,"smooth")
</pre>
</p>


<p>
To get these exact images with the colored background you need
to use GIMP to set the background. The steps I used are:
<ol>
<li>Save the image as a pixmap</li>
<li>Open the saved file in gimp</li>
<li>Dialogs->Colors->ColorPicker button</li>
<li>Eyedrop the color of the web page</li>
```

```
<li>Set the color as the foreground on the FG/BG page</li>
<li>Dialogs->Layers</li>
<li>Duplicate Layer</li>
<li>Layer->Stack->Select bottom layer</li>
<li>Edit->Fill with Foreground color</li>
<li>(on Layers panel)Select image</li>
<li>(on Layers panel) Mode->Darken Only</li>
</ol>
Note that you may have to use "lighten only" first before it will
allow you to choose "darken only".
</p>

\getchunk{page foot}
```

dlmfinequalities.xhtml

— dlmfinequalities.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Inequalities
  </div>
  <hr/>
<h3>Inequalities</h3>
<h6>Contents</h6>
<ul>
 <li>Real Variables</li>
 <li>Complex Variables</li>
</ul>

<h4>Real Variables</h4>
<p>Throughout this subsection
<m:math display="inline">
 <m:mrow>
  <m:mi>x</m:mi>
  <m:mo>&gt;</m:mo>
  <m:mn>0</m:mn>
 </m:mrow>
</m:math>.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
```

```
<m:mn>1</m:mn>
<m:mo>&lt;</m:mo>
<m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>&#x03C0;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mfrac bevelled="true">
    <m:mn>1</m:mn>
    <m:mn>2</m:mn>
   </m:mfrac>
  </m:mrow>
 </m:msup>
 <m:msup>
  <m:mi>x</m:mi>
  <m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mfrac bevelled="true">
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mo>-</m:mo>
   <m:mi>x</m:mi>
  </m:mrow>
 </m:msup>
 <m:msup>
  <m:mi mathvariant="normal">&#x2147;</m:mi>
  <m:mi>x</m:mi>
 </m:msup>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>x</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>&lt;</m:mo>
<m:msup>
 <m:mi mathvariant="normal">&#x2147;</m:mi>
 <m:mfrac bevelled="true">
  <m:mn>1</m:mn>
  <m:mrow>
```

```
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>12</m:mn>
        <m:mi>x</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mfrac>
    </m:msup>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>x</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
     <m:mo>+</m:mo>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mfrac bevelled="true">
         <m:mn>1</m:mn>
         <m:mi>x</m:mi>
        </m:mfrac>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
    <m:mo>&#x2264;</m:mo>
    <m:mn>2</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
```

```
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x0393;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mi>x</m:mi>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mn>2</m:mn>
      </m:msup>
     </m:mfrac>
     <m:mo>+</m:mo>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x0393;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mfrac bevelled="true">
           <m:mn>1</m:mn>
           <m:mi>x</m:mi>
          </m:mfrac>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mn>2</m:mn>
      </m:msup>
     </m:mfrac>
    </m:mrow>
    <m:mo>&#x2264;</m:mo>
    <m:mn>2</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
```

```
<m:mrow>
 <m:mrow>
  <m:msup>
   <m:mi>x</m:mi>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>-</m:mo>
    <m:mi>s</m:mi>
   </m:mrow>
  </m:msup>
  <m:mo>&lt;</m:mo>
  <m:mfrac>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mi>s</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
  <m:mo>&lt;</m:mo>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>x</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>-</m:mo>
    <m:mi>s</m:mi>
   </m:mrow>
  </m:msup>
```

```
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>s</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>exp</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>-</m:mo>
         <m:mi>s</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mi>&#x03C8;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>x</m:mi>
          <m:mo>+</m:mo>
          <m:msup>
           <m:mi>s</m:mi>
           <m:mfrac bevelled="true">
            <m:mn>1</m:mn>
            <m:mn>2</m:mn>
           </m:mfrac>
          </m:msup>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
```

```
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>&#x2264;</m:mo>
  <m:mfrac>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mi>s</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
  <m:mo>&#x2264;</m:mo>
  <m:mrow>
   <m:mi>exp</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:mi>s</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>x</m:mi>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mstyle displaystyle="false">
```

```
            <m:mfrac>
             <m:mn>1</m:mn>
             <m:mn>2</m:mn>
            </m:mfrac>
           </m:mstyle>
           <m:mrow>
            <m:mo>(</m:mo>
            <m:mrow>
             <m:mi>s</m:mi>
             <m:mo>+</m:mo>
             <m:mn>1</m:mn>
            </m:mrow>
            <m:mo>)</m:mo>
           </m:mrow>
          </m:mrow>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>s</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>
</div>

<h4>Complex Variables</h4>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>x</m:mi>
        <m:mo>+</m:mo>
```

```
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>|</m:mo>
    </m:mrow>
    <m:mo>&#x2264;</m:mo>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>x</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>|</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>x</m:mi>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>|</m:mo>
    </m:mrow>
    <m:mo>&#x2265;</m:mo>
    <m:mrow>
     <m:msup>
```

```
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>sech</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>&#x03C0;</m:mi>
          <m:mi>y</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mfrac bevelled="true">
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>x</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mn>2</m:mn>
   </m:mfrac>
  </m:mrow>
 </m:math>
</div>

<p>For
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>b</m:mi>
    <m:mo>-</m:mo>
    <m:mi>a</m:mi>
```

```
   </m:mrow>
   <m:mo>&#x2265;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>, and
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mi>x</m:mi>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>y</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math> with
 <m:math display="inline">
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>&#x2223;</m:mo>
     <m:mfrac>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>+</m:mo>
         <m:mi>a</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
```

```
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>+</m:mo>
         <m:mi>b</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
     <m:mo>&#x2223;</m:mo>
    </m:mrow>
    <m:mo>&#x2264;</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:msup>
      <m:mrow>
       <m:mo>|</m:mo>
       <m:mi>z</m:mi>
       <m:mo>|</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mi>b</m:mi>
       <m:mo>-</m:mo>
       <m:mi>a</m:mi>
      </m:mrow>
     </m:msup>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>For
 <m:math display="inline">
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
```

```
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>|</m:mo>
  </m:mrow>
  <m:mo>&#x2264;</m:mo>
  <m:mrow>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mfrac bevelled="true">
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
   </m:msup>
   <m:msup>
    <m:mrow>
     <m:mo>|</m:mo>
     <m:mi>z</m:mi>
     <m:mo>|</m:mo>
    </m:mrow>
    <m:mrow>
     <m:mi>x</m:mi>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mfrac bevelled="true">
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:msup>
   <m:msup>
    <m:mi mathvariant="normal">&#x2147;</m:mi>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac bevelled="true">
      <m:mrow>
       <m:mi>&#x03C0;</m:mi>
       <m:mrow>
        <m:mo>|</m:mo>
        <m:mi>y</m:mi>
        <m:mo>|</m:mo>
```

```
          </m:mrow>
         </m:mrow>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mrow>
      </m:msup>
      <m:mrow>
       <m:mi>exp</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mstyle displaystyle="false">
          <m:mfrac>
           <m:mn>1</m:mn>
           <m:mn>6</m:mn>
          </m:mfrac>
         </m:mstyle>
         <m:msup>
          <m:mrow>
           <m:mo>|</m:mo>
           <m:mi>z</m:mi>
           <m:mo>|</m:mo>
          </m:mrow>
          <m:mrow>
           <m:mo>-</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
         </m:msup>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmfinfiniteproducts.xhtml

— dlmfinfiniteproducts.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
```

```
   The Gamma Function -- Infinite Products
  </div>
  <hr/>
<h3>Infinite Products</h3>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munder>
      <m:mo movablelimits="false">lim</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>&#x2192;</m:mo>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:mrow>
     </m:munder>
     <m:mfrac>
      <m:mrow>
       <m:mrow>
       <m:mi>k</m:mi>
       <m:mi mathvariant="normal">!</m:mi>
      </m:mrow>
      <m:msup>
       <m:mi>k</m:mi>
       <m:mi>z</m:mi>
      </m:msup>
     </m:mrow>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi mathvariant="normal">&#x22EF;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
```

```
        <m:mo>+</m:mo>
        <m:mi>k</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
    <m:mo>=</m:mo>
    <m:mrow>
```

```
     <m:mi>z</m:mi>
     <m:msup>
      <m:mi mathvariant="normal">&#x2147;</m:mi>
      <m:mrow>
       <m:mi>&#x03B3;</m:mi>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x220F;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:munderover>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mi>z</m:mi>
         <m:mi>k</m:mi>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:msup>
       <m:mi mathvariant="normal">&#x2147;</m:mi>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mfrac bevelled="true">
         <m:mi>z</m:mi>
         <m:mi>k</m:mi>
        </m:mfrac>
       </m:mrow>
      </m:msup>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msup>
     <m:mrow>
      <m:mo>|</m:mo>
```

```
  <m:mfrac>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>x</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mi>y</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
  <m:mo>|</m:mo>
 </m:mrow>
 <m:mn>2</m:mn>
</m:msup>
<m:mo>=</m:mo>
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x220F;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:munderover>
 <m:mo>(</m:mo>
 <m:mrow>
  <m:mn>1</m:mn>
  <m:mo>+</m:mo>
  <m:mfrac>
   <m:msup>
    <m:mi>y</m:mi>
    <m:mn>2</m:mn>
   </m:msup>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>x</m:mi>
```

```
          <m:mo>+</m:mo>
          <m:mi>k</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mn>2</m:mn>
       </m:msup>
      </m:mfrac>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi>m</m:mi>
     </m:munderover>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
```

```
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi>m</m:mi>
     </m:munderover>
     <m:msub>
      <m:mi>b</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>then
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x220F;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>0</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:mfrac>
      <m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:msub>
          <m:mi>a</m:mi>
          <m:mn>1</m:mn>
         </m:msub>
         <m:mo>+</m:mo>
         <m:mi>k</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
```

```
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mn>2</m:mn>
     </m:msub>
     <m:mo>+</m:mo>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi mathvariant="normal">&#x22EF;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mi>m</m:mi>
     </m:msub>
     <m:mo>+</m:mo>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>b</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mo>+</m:mo>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>b</m:mi>
      <m:mn>2</m:mn>
     </m:msub>
     <m:mo>+</m:mo>
     <m:mi>k</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi mathvariant="normal">&#x22EF;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
```

```
         <m:msub>
          <m:mi>b</m:mi>
          <m:mi>m</m:mi>
         </m:msub>
         <m:mo>+</m:mo>
         <m:mi>k</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>b</m:mi>
         <m:mn>1</m:mn>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>b</m:mi>
         <m:mn>2</m:mn>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mi mathvariant="normal">&#x22EF;</m:mi>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>b</m:mi>
         <m:mi>m</m:mi>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
```

```
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>a</m:mi>
         <m:mn>1</m:mn>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>a</m:mi>
         <m:mn>2</m:mn>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mi mathvariant="normal">&#x22EF;</m:mi>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>a</m:mi>
         <m:mi>m</m:mi>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>provided that none of the
 <m:math display="inline">
  <m:msub>
   <m:mi>b</m:mi>
   <m:mi>k</m:mi>
  </m:msub>
 </m:math>
 is zero or a negative integer.
</p>
\getchunk{page foot}
```

dlmfintegrals.xhtml

— dlmfintegrals.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Integrals
  </div>
  <hr/>
<h3>Integrals</h3>

<a name="equation1"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mrow>
        <m:mi>c</m:mi>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x221E;</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mrow>
        <m:mi>c</m:mi>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x221E;</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
        </m:mrow>
       </m:mrow>
      </m:msubsup>
```

```
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>s</m:mi>
       <m:mo>+</m:mo>
       <m:mi>a</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>b</m:mi>
       <m:mo>-</m:mo>
       <m:mi>s</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:msup>
     <m:mi>z</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>s</m:mi>
     </m:mrow>
    </m:msup>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:mi>s</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mfrac>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:msup>
     <m:mi>z</m:mi>
```

```
       <m:mi>a</m:mi>
      </m:msup>
     </m:mrow>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
     </m:msup>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x211C;</m:mi>
     <m:mi>a</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>&lt;</m:mo>
```

```
   <m:mi>c</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>b</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mi>&#x03C0;</m:mi>
  </m:mrow>
 </m:math>.
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi mathvariant="normal">&#x221E;</m:mi>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:msubsup>
      <m:msup>
       <m:mrow>
        <m:mo>|</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x0393;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
```

```
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mn>2</m:mn>
  </m:msup>
  <m:msup>
   <m:mi mathvariant="normal">&#x2147;</m:mi>
   <m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mi>t</m:mi>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:mi>t</m:mi>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mfrac>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
```

```
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mrow>
         <m:mi>sin</m:mi>
         <m:mspace width="0.2em"/>
         <m:mi>b</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>a</m:mi>
      </m:mrow>
     </m:msup>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>b</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mi>&#x03C0;</m:mi>
  </m:mrow>
 </m:math>.
</div>

<h5>Barnes's Beta Integral</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
      </m:mrow>
     </m:mfrac>
```

```
<m:mrow>
 <m:msubsup>
  <m:mo>&#x222B;</m:mo>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:msubsup>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>a</m:mi>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>b</m:mi>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>c</m:mi>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
```

```
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>d</m:mi>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mfrac>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mi>c</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mi>d</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>b</m:mi>
```

```
         <m:mo>+</m:mo>
         <m:mi>c</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>b</m:mi>
         <m:mo>+</m:mo>
         <m:mi>d</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
        <m:mo>+</m:mo>
        <m:mi>c</m:mi>
        <m:mo>+</m:mo>
        <m:mi>d</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x211C;</m:mi>
     <m:mi>a</m:mi>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x211C;</m:mi>
     <m:mi>b</m:mi>
    </m:mrow>
```

```
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x211C;</m:mi>
     <m:mi>c</m:mi>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x211C;</m:mi>
     <m:mi>d</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>
</div>

<h5>Ramanujan's Beta Integral</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2146;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>+</m:mo>
          <m:mi>t</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
       <m:mrow>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
```

```
     <m:mrow>
      <m:mi>b</m:mi>
      <m:mo>+</m:mo>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>c</m:mi>
      <m:mo>-</m:mo>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>d</m:mi>
      <m:mo>-</m:mo>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mfrac>
</m:mrow>
<m:mo>=</m:mo>
<m:mfrac>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>+</m:mo>
     <m:mi>b</m:mi>
     <m:mo>+</m:mo>
     <m:mi>c</m:mi>
     <m:mo>+</m:mo>
     <m:mi>d</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mn>3</m:mn>
   </m:mrow>
```

```
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>+</m:mo>
         <m:mi>c</m:mi>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>a</m:mi>
         <m:mo>+</m:mo>
         <m:mi>d</m:mi>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>b</m:mi>
         <m:mo>+</m:mo>
         <m:mi>c</m:mi>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
```

```
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>b</m:mi>
          <m:mo>+</m:mo>
          <m:mi>d</m:mi>
         </m:mrow>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
      <m:mo>+</m:mo>
      <m:mi>c</m:mi>
      <m:mo>+</m:mo>
      <m:mi>d</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>3</m:mn>
  </m:mrow>
 </m:math>.
</div>

<h5>de Branges-Wilson Beta Integral</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
```

```
 <m:mn>1</m:mn>
 <m:mrow>
  <m:mn>4</m:mn>
  <m:mi>&#x03C0;</m:mi>
 </m:mrow>
</m:mfrac>
<m:mo></m:mo>
<m:mrow>
 <m:msubsup>
  <m:mo>&#x222B;</m:mo>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:msubsup>
 <m:mfrac>
  <m:mrow>
   <m:msubsup>
    <m:mo>&#x220F;</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>=</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mn>4</m:mn>
   </m:msubsup>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:msub>
       <m:mi>a</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
```

```
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mfrac>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:mi>t</m:mi>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mfrac>
 <m:mrow>
  <m:msub>
   <m:mo>&#x220F;</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>&#x2264;</m:mo>
```

```
     <m:mi>j</m:mi>
     <m:mo>&lt;</m:mo>
     <m:mi>k</m:mi>
     <m:mo>&#x2264;</m:mo>
     <m:mn>4</m:mn>
    </m:mrow>
   </m:msub>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mi>j</m:mi>
     </m:msub>
     <m:mo>+</m:mo>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mo>+</m:mo>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mn>2</m:mn>
     </m:msub>
     <m:mo>+</m:mo>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mn>3</m:mn>
     </m:msub>
     <m:mo>+</m:mo>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mn>4</m:mn>
     </m:msub>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mfrac>
</m:mrow>
```

```
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
    <m:mo>,</m:mo>
    <m:mn>4</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>
\getchunk{page foot}
```

dlmfintegralrepresentations.xhtml

— dlmfintegralrepresentations.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Integral Representations
```

```
  </div>
  <hr/>
<h3>Integral Representations</h3>

<h6>Contents</h6>
<ul>
 <li>Gamma Function</li>
 <li>Psi Function and Euler's Constant</li>
</ul>

<h4>Gamma Function</h4>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mi>&#x03BC;</m:mi>
     </m:mfrac>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mfrac>
        <m:mi>&#x03BD;</m:mi>
        <m:mi>&#x03BC;</m:mi>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:msup>
       <m:mi>z</m:mi>
       <m:mfrac bevelled="true">
        <m:mi>&#x03BD;</m:mi>
        <m:mi>&#x03BC;</m:mi>
       </m:mfrac>
      </m:msup>
     </m:mfrac>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mrow>
      <m:mi>exp</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
```

```
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:msup>
          <m:mi>t</m:mi>
          <m:mi>&#x03BC;</m:mi>
         </m:msup>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:msup>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mi>&#x03BD;</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>&#x03BD;</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03BC;</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>, and
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
```

```
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>. (The fractional powers have their principal values.)
</p>

<h5>Hankel’s Loop Integral</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi mathvariant="normal">&#x221E;</m:mi>
       </m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>0</m:mn>
         <m:mo>+</m:mo>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:msubsup>
      <m:msup>
       <m:mi mathvariant="normal">&#x2147;</m:mi>
       <m:mi>t</m:mi>
      </m:msup>
```

```
      <m:msup>
       <m:mi>t</m:mi>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
      </m:msup>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2146;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where the contour begins at
 <m:math display="inline">
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math>, circles the origin once in the positive direction, and returns to
 <m:math display="inline">
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math>.
 <m:math display="inline">
  <m:msup>
   <m:mi>t</m:mi>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mi>z</m:mi>
   </m:mrow>
  </m:msup>
 </m:math> has its principal value where
 <m:math display="inline">
  <m:mi>t</m:mi>
 </m:math> crosses the positive real axis, and is continuous.
</p>

<div align="center">
 <!-- need a better Axiom graphic than this
  <img width="302" height="150" alt="" src="bitmaps/9F1.png"/> -->
</div>

<div align="center">
 <m:math display="inline">
  <m:mi>t</m:mi>
 </m:math>-plane. Contour for Hankel's loop integral.
```

```
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi>c</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:msup>
      <m:mrow>
       <m:mo>|</m:mo>
       <m:mi>t</m:mi>
       <m:mo>|</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:msup>
      <m:mi mathvariant="normal">&#x2147;</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mi>c</m:mi>
```

```
        <m:msup>
         <m:mi>t</m:mi>
         <m:mn>2</m:mn>
        </m:msup>
       </m:mrow>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>c</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>
</div>

<p>where the path is the real axis.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
```

```
<m:mrow>
 <m:msubsup>
  <m:mo>&#x222B;</m:mo>
  <m:mn>1</m:mn>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:msubsup>
 <m:msup>
  <m:mi>t</m:mi>
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:msup>
 <m:msup>
  <m:mi mathvariant="normal">&#x2147;</m:mi>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi>t</m:mi>
  </m:mrow>
 </m:msup>
 <m:mrow>
  <m:mi mathvariant="normal">&#x2146;</m:mi>
  <m:mi>t</m:mi>
 </m:mrow>
</m:mrow>
<m:mo>+</m:mo>
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x2211;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:munderover>
 <m:mfrac>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi>k</m:mi>
  </m:msup>
  <m:mrow>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>z</m:mi>
```

```
          <m:mo>+</m:mo>
          <m:mi>k</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mi mathvariant="normal">!</m:mi>
        </m:mrow>
       </m:mrow>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
```

```
<m:mo>=</m:mo>
<m:mrow>
 <m:msubsup>
  <m:mo>&#x222B;</m:mo>
  <m:mn>0</m:mn>
  <m:mi mathvariant="normal">&#x221E;</m:mi>
 </m:msubsup>
 <m:msup>
  <m:mi>t</m:mi>
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:msup>
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mrow>
   <m:msup>
    <m:mi mathvariant="normal">&#x2147;</m:mi>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:msup>
   <m:mo>-</m:mo>
   <m:mrow>
    <m:munderover>
     <m:mo movablelimits="false">&#x2211;</m:mo>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>=</m:mo>
      <m:mn>0</m:mn>
     </m:mrow>
     <m:mi>n</m:mi>
    </m:munderover>
    <m:mfrac>
     <m:mrow>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi>k</m:mi>
      </m:msup>
      <m:msup>
       <m:mi>t</m:mi>
       <m:mi>k</m:mi>
      </m:msup>
     </m:mrow>
```

```
         <m:mrow>
          <m:mi>k</m:mi>
          <m:mi mathvariant="normal">!</m:mi>
         </m:mrow>
        </m:mfrac>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:mspace width="0.2em"/>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>n</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mi>n</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
```

```
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi>cos</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mi>&#x03C0;</m:mi>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:msup>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi>cos</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
```

```
<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi>sin</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mi>&#x03C0;</m:mi>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
```

```
     <m:msup>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi>sin</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>.
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
```

```
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mi>n</m:mi>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi>cos</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mfrac>
        <m:mi>&#x03C0;</m:mi>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>n</m:mi>
        </m:mrow>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mrow>
      <m:mi>cos</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:msup>
        <m:mi>t</m:mi>
        <m:mi>n</m:mi>
       </m:msup>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
```

```
 <m:math display="inline">
  <m:mrow>
   <m:mi>n</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mn>3</m:mn>
    <m:mo>,</m:mo>
    <m:mn>4</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mi>n</m:mi>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi>sin</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mfrac>
        <m:mi>&#x03C0;</m:mi>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>n</m:mi>
        </m:mrow>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
```

```
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mrow>
      <m:mi>sin</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:msup>
        <m:mi>t</m:mi>
        <m:mi>n</m:mi>
       </m:msup>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>n</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mn>3</m:mn>
    <m:mo>,</m:mo>
    <m:mn>4</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>

<h5>Binet’s Formula</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>ln</m:mi>
```

```
  <m:mspace width="0.2em"/>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>-</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mspace width="0.2em"/>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>z</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>-</m:mo>
   <m:mi>z</m:mi>
  </m:mrow>
  <m:mo>+</m:mo>
  <m:mrow>
   <m:mstyle displaystyle="false">
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
   </m:mstyle>
   <m:mrow>
    <m:mi>ln</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
```

```
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mrow>
       <m:msubsup>
        <m:mo>&#x222B;</m:mo>
        <m:mn>0</m:mn>
        <m:mi mathvariant="normal">&#x221E;</m:mi>
       </m:msubsup>
       <m:mfrac>
        <m:mrow>
         <m:mi>arctan</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mfrac bevelled="true">
           <m:mi>t</m:mi>
           <m:mi>z</m:mi>
          </m:mfrac>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mrow>
         <m:msup>
          <m:mi mathvariant="normal">&#x2147;</m:mi>
          <m:mrow>
           <m:mn>2</m:mn>
           <m:mi>&#x03C0;</m:mi>
           <m:mi>t</m:mi>
          </m:mrow>
         </m:msup>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:mfrac>
       <m:mspace width="0.2em"/>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2146;</m:mi>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
```

```
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mo>&#x2061;</m:mo>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mfrac bevelled="true">
    <m:mi>&#x03C0;</m:mi>
    <m:mn>2</m:mn>
   </m:mfrac>
  </m:mrow>
 </m:math> and the inverse tangent has its principal value.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>z</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mi>&#x03B3;</m:mi>
       <m:mi>z</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>&#x03C0;</m:mi>
```

```
   <m:mi mathvariant="normal">&#x2148;</m:mi>
  </m:mrow>
 </m:mfrac>
 <m:mrow>
  <m:msubsup>
   <m:mo>&#x222B;</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>c</m:mi>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>c</m:mi>
    </m:mrow>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:msubsup>
  <m:mfrac>
   <m:mrow>
    <m:mi>&#x03C0;</m:mi>
    <m:msup>
     <m:mi>z</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>s</m:mi>
     </m:mrow>
    </m:msup>
   </m:mrow>
   <m:mrow>
    <m:mi>s</m:mi>
    <m:mspace width="0.2em"/>
    <m:mrow>
     <m:mi>sin</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>&#x03C0;</m:mi>
       <m:mi>s</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
```

```
          </m:mrow>
         </m:mrow>
        </m:mrow>
       </m:mfrac>
       <m:mrow>
        <m:mi>&#x03B6;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mi>s</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
       <m:mspace width="0.2em"/>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2146;</m:mi>
        <m:mi>s</m:mi>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mo>&#x2061;</m:mo>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mrow>
    <m:mi>&#x03C0;</m:mi>
    <m:mo>-</m:mo>
    <m:mi>&#x03B4;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math> (
 <m:math display="inline">
  <m:mrow>
   <m:none/>
   <m:mo>&lt;</m:mo>
   <m:mi>&#x03C0;</m:mi>
  </m:mrow>
```

```
 </m:math>),
 <m:math display="inline">
  <m:mrow>
   <m:mn>1</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>c</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:math>, and
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03B6;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>s</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>
</p>

<p>For additional representations see
<a href="http://dlmf.nist.gov/Contents/bib/W#whittaker:1927:cma">
   Whittaker and Watson(1927)</a>
</p>

<h4>Psi Function and Euler’s Constant</h4>
<p>For
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>z</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
```

```
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mfrac>
        <m:msup>
         <m:mi mathvariant="normal">&#x2147;</m:mi>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mi>t</m:mi>
         </m:mrow>
        </m:msup>
        <m:mi>t</m:mi>
       </m:mfrac>
       <m:mo>-</m:mo>
       <m:mfrac>
        <m:msup>
         <m:mi mathvariant="normal">&#x2147;</m:mi>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mrow>
           <m:mi>z</m:mi>
           <m:mi>t</m:mi>
          </m:mrow>
         </m:mrow>
        </m:msup>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>-</m:mo>
         <m:msup>
          <m:mi mathvariant="normal">&#x2147;</m:mi>
          <m:mrow>
           <m:mo>-</m:mo>
           <m:mi>t</m:mi>
          </m:mrow>
         </m:msup>
        </m:mrow>
       </m:mfrac>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
```

```
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>ln</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>z</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:msubsup>
       <m:mo>&#x222B;</m:mo>
       <m:mn>0</m:mn>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:msubsup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mi>t</m:mi>
        </m:mfrac>
        <m:mo>-</m:mo>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mrow>
          <m:mn>1</m:mn>
          <m:mo>-</m:mo>
          <m:msup>
           <m:mi mathvariant="normal">&#x2147;</m:mi>
           <m:mrow>
            <m:mo>-</m:mo>
            <m:mi>t</m:mi>
           </m:mrow>
          </m:msup>
         </m:mrow>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
```

```
      <m:mspace width="0.2em"/>
      <m:msup>
       <m:mi mathvariant="normal">&#x2147;</m:mi>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mi>t</m:mi>
         <m:mi>z</m:mi>
        </m:mrow>
       </m:mrow>
      </m:msup>
      <m:mspace width="0.2em"/>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2146;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:msup>
        <m:mi mathvariant="normal">&#x2147;</m:mi>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mi>t</m:mi>
        </m:mrow>
       </m:msup>
       <m:mo>-</m:mo>
       <m:mfrac>
```

```
        <m:mn>1</m:mn>
        <m:msup>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mn>1</m:mn>
           <m:mo>+</m:mo>
           <m:mi>t</m:mi>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
         <m:mi>z</m:mi>
        </m:msup>
       </m:mfrac>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mfrac>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2146;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
      <m:mi>t</m:mi>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>ln</m:mi>
      <m:mspace width="0.2em"/>
      <m:mi>z</m:mi>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
```

```
    <m:mn>2</m:mn>
    <m:mi>z</m:mi>
   </m:mrow>
  </m:mfrac>
  <m:mo>-</m:mo>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mrow>
    <m:msubsup>
     <m:mo>&#x222B;</m:mo>
     <m:mn>0</m:mn>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:msubsup>
    <m:mfrac>
     <m:mrow>
      <m:mi>t</m:mi>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2146;</m:mi>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:msup>
         <m:mi>t</m:mi>
         <m:mn>2</m:mn>
        </m:msup>
        <m:mo>+</m:mo>
        <m:msup>
         <m:mi>z</m:mi>
         <m:mn>2</m:mn>
        </m:msup>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:msup>
         <m:mi mathvariant="normal">&#x2147;</m:mi>
         <m:mrow>
          <m:mn>2</m:mn>
          <m:mi>&#x03C0;</m:mi>
          <m:mi>t</m:mi>
         </m:mrow>
        </m:msup>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
```

```
       </m:mfrac>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mi>&#x03B3;</m:mi>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:msup>
        <m:mi mathvariant="normal">&#x2147;</m:mi>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mi>t</m:mi>
        </m:mrow>
       </m:msup>
       <m:mo>-</m:mo>
       <m:msup>
        <m:mi mathvariant="normal">&#x2147;</m:mi>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mrow>
          <m:mi>z</m:mi>
          <m:mi>t</m:mi>
         </m:mrow>
        </m:mrow>
       </m:msup>
      </m:mrow>
      <m:mrow>
```

```
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:msup>
        <m:mi mathvariant="normal">&#x2147;</m:mi>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mi>t</m:mi>
        </m:mrow>
       </m:msup>
      </m:mrow>
     </m:mfrac>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msubsup>
      <m:mo>&#x222B;</m:mo>
      <m:mn>0</m:mn>
      <m:mn>1</m:mn>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:msup>
        <m:mi>t</m:mi>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
      </m:mrow>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
```

```
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>&#x03B3;</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>&#x03C0;</m:mi>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
       </m:mrow>
      </m:mfrac>
      <m:mrow>
       <m:msubsup>
        <m:mo>&#x222B;</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mi>c</m:mi>
         </m:mrow>
         <m:mo>-</m:mo>
         <m:mrow>
          <m:mi mathvariant="normal">&#x221E;</m:mi>
          <m:mspace width="0.2em"/>
          <m:mi mathvariant="normal">&#x2148;</m:mi>
         </m:mrow>
        </m:mrow>
        <m:mrow>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mi>c</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mrow>
```

```
        <m:mi mathvariant="normal">&#x221E;</m:mi>
        <m:mspace width="0.2em"/>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
       </m:mrow>
      </m:mrow>
     </m:msubsup>
     <m:mfrac>
      <m:mrow>
       <m:mi>&#x03C0;</m:mi>
       <m:msup>
        <m:mi>z</m:mi>
        <m:mrow>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mi>s</m:mi>
         </m:mrow>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
      </m:mrow>
      <m:mrow>
       <m:mi>sin</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>&#x03C0;</m:mi>
         <m:mi>s</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:mi>&#x03B6;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi>s</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>s</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:mrow>
```

```
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mo>-</m:mo>
     <m:mi>&#x03B4;</m:mi>
    </m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:none/>
      <m:mo>&lt;</m:mo>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math> and
 <m:math display="inline">
  <m:mrow>
   <m:mn>1</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>c</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:math>.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mi>&#x03B3;</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
```

```
    <m:msubsup>
     <m:mo>&#x222B;</m:mo>
     <m:mn>0</m:mn>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:msubsup>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:msup>
    <m:mspace width="0.2em"/>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>t</m:mi>
    </m:mrow>
    <m:mspace width="0.2em"/>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:mi>t</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:msubsup>
    <m:mo>&#x222B;</m:mo>
    <m:mn>0</m:mn>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:msubsup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mo>-</m:mo>
     <m:msup>
      <m:mi mathvariant="normal">&#x2147;</m:mi>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>t</m:mi>
      </m:mrow>
     </m:msup>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
```

```
 <m:mspace width="0.2em"/>
 <m:mfrac>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:mi>t</m:mi>
  </m:mrow>
  <m:mi>t</m:mi>
 </m:mfrac>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mrow>
  <m:msubsup>
   <m:mo>&#x222B;</m:mo>
   <m:mn>0</m:mn>
   <m:mn>1</m:mn>
  </m:msubsup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>-</m:mo>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:msup>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mspace width="0.2em"/>
  <m:mfrac>
   <m:mrow>
    <m:mi mathvariant="normal">&#x2146;</m:mi>
    <m:mi>t</m:mi>
   </m:mrow>
   <m:mi>t</m:mi>
  </m:mfrac>
 </m:mrow>
 <m:mo>-</m:mo>
 <m:mrow>
  <m:msubsup>
   <m:mo>&#x222B;</m:mo>
   <m:mn>1</m:mn>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:msubsup>
  <m:msup>
   <m:mi mathvariant="normal">&#x2147;</m:mi>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mi>t</m:mi>
   </m:mrow>
```

```
    </m:msup>
    <m:mfrac>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
     <m:mi>t</m:mi>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:msubsup>
    <m:mo>&#x222B;</m:mo>
    <m:mn>0</m:mn>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:msubsup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:msup>
       <m:mi mathvariant="normal">&#x2147;</m:mi>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:msup>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:msup>
        <m:mi mathvariant="normal">&#x2147;</m:mi>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mi>t</m:mi>
        </m:mrow>
       </m:msup>
      </m:mrow>
     </m:mfrac>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:msup>
       <m:mi mathvariant="normal">&#x2147;</m:mi>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mi>t</m:mi>
       </m:mrow>
      </m:msup>
      <m:mi>t</m:mi>
     </m:mfrac>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
```

```
     <m:mspace width="0.2em"/>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmfmathematicalapplications.xhtml

— dlmfmathematicalapplications.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Mathematical Applications
  </div>
  <hr/>
<h3>Mathematical Applications</h3>
<h6>Contents</h6>
<ul>
 <li>Summation of Rational Functions</li>
 <li>Mellin-Barnes Integrals</li>
 <li>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">n</m:mi>
 </m:math>-Dimensional Sphere</li>
</ul>

<h4>Summation of Rational Functions</h4>

<p>As shown in
 <a href="http://dlmf.nist.gov/Contents/bib/T#temme:1996:sfi">
  Temme(1996)
 </a>(3.4), the results given in
 <a href="dlmfseriesexpansions.xhtml">
  Series Expansions
 </a> can be used to sum infinite series of rational functions.
</p>

<h5>Example</h5>
```

```
<div align="center">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>S</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mstyle displaystyle="false">
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>0</m:mn>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:munderover>
     </m:mstyle>
     <m:msub>
      <m:mi>a</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
   </m:mrow>
   <m:mo>,</m:mo>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mi>k</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mstyle displaystyle="true">
     <m:mfrac>
      <m:mi>k</m:mi>
      <m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mn>3</m:mn>
          <m:mi>k</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mn>2</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
```

```
         <m:mrow>
          <m:mn>2</m:mn>
          <m:mi>k</m:mi>
         </m:mrow>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mstyle>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>By decomposition into partial fractions</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>a</m:mi>
     <m:mi>k</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>2</m:mn>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>+</m:mo>
       <m:mfrac>
        <m:mn>2</m:mn>
        <m:mn>3</m:mn>
       </m:mfrac>
      </m:mrow>
     </m:mfrac>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
       <m:mi>k</m:mi>
```

```
    <m:mo>+</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mn>2</m:mn>
    </m:mfrac>
   </m:mrow>
  </m:mfrac>
  <m:mo>-</m:mo>
  <m:mfrac>
   <m:mn>1</m:mn>
   <m:mrow>
    <m:mi>k</m:mi>
    <m:mo>+</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:mfrac>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:mrow>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
    </m:mfrac>
    <m:mo>-</m:mo>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>+</m:mo>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mo>-</m:mo>
  <m:mrow>
   <m:mn>2</m:mn>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mrow>
```

```
          <m:mi>k</m:mi>
          <m:mo>+</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
        </m:mfrac>
        <m:mo>-</m:mo>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mrow>
          <m:mi>k</m:mi>
          <m:mo>+</m:mo>
          <m:mfrac>
           <m:mn>2</m:mn>
           <m:mn>3</m:mn>
          </m:mfrac>
         </m:mrow>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>Hence from (
 <a href="dlmfseriesexpansions.xhtml#equation6">Series Expansions 6
 </a>), ( Special Values and Extrema
 <a href="dlmfspecialvaluesandextrema.xhtml#equation13">
  Equation 13
 </a> and
 <a href="dlmfspecialvaluesandextrema.xhtml#equation19">
  Equation 19
 </a>)
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mi>S</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
```

```
   </m:mstyle>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>-</m:mo>
 <m:mrow>
  <m:mn>2</m:mn>
  <m:mrow>
   <m:mi>&#x03C8;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mstyle displaystyle="false">
     <m:mfrac>
      <m:mn>2</m:mn>
      <m:mn>3</m:mn>
     </m:mfrac>
    </m:mstyle>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mo>-</m:mo>
 <m:mi>&#x03B3;</m:mi>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mrow>
  <m:mn>3</m:mn>
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mn>3</m:mn>
  </m:mrow>
 </m:mrow>
 <m:mo>-</m:mo>
 <m:mrow>
  <m:mn>2</m:mn>
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:mrow>
 <m:mo>-</m:mo>
 <m:mrow>
  <m:mstyle displaystyle="false">
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mn>3</m:mn>
   </m:mfrac>
  </m:mstyle>
  <m:mi>&#x03C0;</m:mi>
  <m:msqrt>
   <m:mn>3</m:mn>
  </m:msqrt>
 </m:mrow>
```

```
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>
</div>

<h4>Mellin-Barnes Integrals</h4>
<p>Many special functions
 <m:math display="inline">
  <m:mrow>
   <m:mi>f</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> can be represented as a <em>Mellin-Barnes integral</em>, that is,
  an integral of a product of gamma functions, reciprocals of gamma
  functions, and a power of
 <m:math display="inline">
  <m:mi>z</m:mi>
 </m:math>, the integration contour being doubly-infinite and eventually
  parallel to the imaginary axis. The left-hand side of (
 <a href="dlmfintegrals.xhtml#equation1">
  Integral Equation 1
 </a>) is a typical example. By translating the contour parallel to itself
 and summing the residues of the integrand, asymptotic expansions of
 <m:math display="inline">
  <m:mrow>
   <m:mi>f</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math> for large
 <m:math display="inline">
  <m:mrow>
   <m:mo>|</m:mo>
   <m:mi>z</m:mi>
   <m:mo>|</m:mo>
  </m:mrow>
 </m:math>, or small
 <m:math display="inline">
  <m:mrow>
   <m:mo>|</m:mo>
   <m:mi>z</m:mi>
   <m:mo>|</m:mo>
  </m:mrow>
 </m:math>, can be obtained complete with an integral representation of the
  error term.
</p>
```

```
<h4>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">n</m:mi>
 </m:math>-Dimensional Sphere</h4>

<p>The volume
 <m:math display="inline">
  <m:mi>V</m:mi>
 </m:math> and surface area
 <m:math display="inline">
  <m:mi>A</m:mi>
 </m:math> of the
 <m:math display="inline">
  <m:mi>n</m:mi>
 </m:math>-dimensional sphere of radius
 <m:math display="inline">
  <m:mi>r</m:mi>
 </m:math> are given by
</p>

<div align="center">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi>V</m:mi>
    <m:mo>=</m:mo>
    <m:mstyle displaystyle="true">
     <m:mfrac>
      <m:mrow>
       <m:msup>
        <m:mi>&#x03C0;</m:mi>
        <m:mrow>
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
         <m:mi>n</m:mi>
        </m:mrow>
       </m:msup>
       <m:msup>
        <m:mi>r</m:mi>
        <m:mi>n</m:mi>
       </m:msup>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mfrac>
           <m:mn>1</m:mn>
           <m:mn>2</m:mn>
```

```
         </m:mfrac>
         <m:mi>n</m:mi>
        </m:mrow>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mstyle>
  </m:mrow>
  <m:mo>,</m:mo>
 </m:mrow>
</m:math>
<m:math display="inline">
 <m:mrow>
  <m:mrow>
   <m:mi>S</m:mi>
   <m:mo>=</m:mo>
   <m:mstyle displaystyle="true">
    <m:mfrac>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:msup>
       <m:mi>&#x03C0;</m:mi>
       <m:mrow>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
        <m:mi>n</m:mi>
       </m:mrow>
      </m:msup>
      <m:msup>
       <m:mi>r</m:mi>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:msup>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
        <m:mi>n</m:mi>
       </m:mrow>
```

```
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mfrac>
    </m:mstyle>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mstyle displaystyle="true">
      <m:mfrac>
       <m:mi>n</m:mi>
       <m:mi>r</m:mi>
      </m:mfrac>
     </m:mstyle>
     <m:mi>V</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmfmethodsofcomputation.xhtml

— dlmfmethodsofcomputation.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Methods of Computation
  </div>
  <hr/>
<h3>Methods of Computation</h3>

<p>An effective way of computing
 <m:math display="inline">
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>
in the right half-plane is backward recurrence, beginning with a value
generated from the
```

```
<a href="dlmfasymptoticexpansions.xhtml#equation3">
 asymptotic expansion
</a>
Or we can use forward recurrence, with an
<a href="dlmfseriesexpansions.xhtml#equation3">
 initial value
</a>.
For the left half-plane we can continue the backward recurrence or
make use of the
<a href="dlmffunctionrelations.xhtml#equation3">
 reflection formula
</a>.
</p>

<p>Similarly for
 <m:math display="inline">
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03C8;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>, and the polygamma functions.
</p>

<p>For a comprehensive survey see
 <a href="http://dlmf.nist.gov/Contents/bib/V#vanderlaan:1984:csf">
  van der Laan and Temme(1984)
 </a>(Chapter III).
 See also
 <a href="http://dlmf.nist.gov/Contents/bib/B#borwein:1992:feg">
  Borwein and Zucker(1992)
 </a>.
</p>
\getchunk{page foot}
```

dlmfmultidimensionalintegral.xhtml

— dlmfmultidimensionalintegral.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Multidimensional Integral
  </div>
  <hr/>
<h3>Multidimensional Integrals</h3>

<p>Let
 <m:math display="inline">
  <m:msub>
   <m:mi>V</m:mi>
   <m:mi>n</m:mi>
  </m:msub>
 </m:math> be the simplex:
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>t</m:mi>
     <m:mn>1</m:mn>
    </m:msub>
    <m:mo>+</m:mo>
    <m:msub>
     <m:mi>t</m:mi>
     <m:mn>2</m:mn>
    </m:msub>
    <m:mo>+</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
    <m:mo>+</m:mo>
    <m:msub>
     <m:mi>t</m:mi>
     <m:mi>n</m:mi>
    </m:msub>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:msub>
    <m:mi>t</m:mi>
    <m:mi>k</m:mi>
```

```
   </m:msub>
   <m:mo>&#x2265;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>. Then for
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:msub>
     <m:mi>z</m:mi>
     <m:mi>k</m:mi>
    </m:msub>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>,</m:mo>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mi>n</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mo>&#x222B;</m:mo>
      <m:msub>
       <m:mi>V</m:mi>
       <m:mi>n</m:mi>
      </m:msub>
     </m:msub>
     <m:msubsup>
      <m:mi>t</m:mi>
      <m:mn>1</m:mn>
```

```
 <m:mrow>
  <m:msub>
   <m:mi>z</m:mi>
   <m:mn>1</m:mn>
  </m:msub>
  <m:mo>-</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
</m:msubsup>
<m:msubsup>
 <m:mi>t</m:mi>
 <m:mn>2</m:mn>
 <m:mrow>
  <m:msub>
   <m:mi>z</m:mi>
   <m:mn>2</m:mn>
  </m:msub>
  <m:mo>-</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
</m:msubsup>
<m:mi mathvariant="normal">&#x22EF;</m:mi>
<m:msubsup>
 <m:mi>t</m:mi>
 <m:mi>n</m:mi>
 <m:mrow>
  <m:msub>
   <m:mi>z</m:mi>
   <m:mi>n</m:mi>
  </m:msub>
  <m:mo>-</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
</m:msubsup>
<m:mrow>
 <m:mi mathvariant="normal">&#x2146;</m:mi>
 <m:msub>
  <m:mi>t</m:mi>
  <m:mn>1</m:mn>
 </m:msub>
</m:mrow>
<m:mrow>
 <m:mi mathvariant="normal">&#x2146;</m:mi>
 <m:msub>
  <m:mi>t</m:mi>
  <m:mn>2</m:mn>
 </m:msub>
</m:mrow>
<m:mi mathvariant="normal">&#x22EF;</m:mi>
<m:mrow>
 <m:mi mathvariant="normal">&#x2146;</m:mi>
 <m:msub>
  <m:mi>t</m:mi>
  <m:mi>n</m:mi>
```

```
      </m:msub>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>z</m:mi>
         <m:mn>1</m:mn>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>z</m:mi>
         <m:mn>2</m:mn>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mi mathvariant="normal">&#x22EF;</m:mi>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:msub>
         <m:mi>z</m:mi>
         <m:mi>n</m:mi>
        </m:msub>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:msub>
         <m:mi>z</m:mi>
         <m:mn>1</m:mn>
        </m:msub>
        <m:mo>+</m:mo>
        <m:msub>
```

```
         <m:mi>z</m:mi>
         <m:mn>2</m:mn>
        </m:msub>
        <m:mo>+</m:mo>
        <m:mi mathvariant="normal">&#x2026;</m:mi>
        <m:mo>+</m:mo>
        <m:msub>
         <m:mi>z</m:mi>
         <m:mi>n</m:mi>
        </m:msub>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mo>&#x222B;</m:mo>
      <m:msub>
       <m:mi>V</m:mi>
       <m:mi>n</m:mi>
      </m:msub>
     </m:msub>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:munderover>
          <m:mo movablelimits="false">&#x2211;</m:mo>
          <m:mrow>
           <m:mi>k</m:mi>
           <m:mo>=</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mi>n</m:mi>
         </m:munderover>
         <m:msub>
          <m:mi>t</m:mi>
          <m:mi>k</m:mi>
         </m:msub>
        </m:mrow>
       </m:mrow>
```

```
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:msub>
       <m:mi>z</m:mi>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:msub>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
    </m:msup>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x220F;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi>n</m:mi>
     </m:munderover>
     <m:msubsup>
      <m:mi>t</m:mi>
      <m:mi>k</m:mi>
      <m:mrow>
       <m:msub>
        <m:mi>z</m:mi>
        <m:mi>k</m:mi>
       </m:msub>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msubsup>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2146;</m:mi>
      <m:msub>
       <m:mi>t</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mfrac>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:msub>
```

```
    <m:mi>z</m:mi>
    <m:mn>1</m:mn>
   </m:msub>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:msub>
    <m:mi>z</m:mi>
    <m:mn>2</m:mn>
   </m:msub>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mi mathvariant="normal">&#x22EF;</m:mi>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:msub>
    <m:mi>z</m:mi>
    <m:mrow>
     <m:mi>n</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:msub>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mrow>
 <m:mi mathvariant="normal">&#x0393;</m:mi>
 <m:mrow>
  <m:mo>(</m:mo>
  <m:mrow>
   <m:msub>
    <m:mi>z</m:mi>
    <m:mn>1</m:mn>
   </m:msub>
   <m:mo>+</m:mo>
   <m:msub>
    <m:mi>z</m:mi>
    <m:mn>2</m:mn>
   </m:msub>
   <m:mo>+</m:mo>
   <m:mi mathvariant="normal">&#x2026;</m:mi>
   <m:mo>+</m:mo>
   <m:msub>
    <m:mi>z</m:mi>
    <m:mrow>
```

```
          <m:mi>n</m:mi>
          <m:mo>+</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
        </m:msub>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<h5>Selberg-type Integrals</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0394;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:msub>
        <m:mi>t</m:mi>
        <m:mn>1</m:mn>
       </m:msub>
       <m:mo>,</m:mo>
       <m:msub>
        <m:mi>t</m:mi>
        <m:mn>2</m:mn>
       </m:msub>
       <m:mo>,</m:mo>
       <m:mi mathvariant="normal">&#x2026;</m:mi>
       <m:mo>,</m:mo>
       <m:msub>
        <m:mi>t</m:mi>
        <m:mi>n</m:mi>
       </m:msub>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munder>
      <m:mo movablelimits="false">&#x220F;</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>&#x2264;</m:mo>
       <m:mi>j</m:mi>
```

```
       <m:mo>&lt;</m:mo>
       <m:mi>k</m:mi>
       <m:mo>&#x2264;</m:mo>
       <m:mi>n</m:mi>
      </m:mrow>
     </m:munder>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:msub>
       <m:mi>t</m:mi>
       <m:mi>j</m:mi>
      </m:msub>
      <m:mo>-</m:mo>
      <m:msub>
       <m:mi>t</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>Then
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mo>&#x222B;</m:mo>
      <m:msup>
       <m:mrow>
        <m:mo>[</m:mo>
        <m:mrow>
         <m:mn>0</m:mn>
         <m:mo>,</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>]</m:mo>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:msup>
     </m:msub>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:msub>
      <m:mi>t</m:mi>
```

```
 <m:mn>2</m:mn>
</m:msub>
<m:mi mathvariant="normal">&#x22EF;</m:mi>
<m:msub>
 <m:mi>t</m:mi>
 <m:mi>m</m:mi>
</m:msub>
<m:msup>
 <m:mrow>
  <m:mo>|</m:mo>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0394;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mo>,</m:mo>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
     <m:mo>,</m:mo>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mi>n</m:mi>
     </m:msub>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>|</m:mo>
 </m:mrow>
 <m:mrow>
  <m:mn>2</m:mn>
  <m:mi>c</m:mi>
 </m:mrow>
</m:msup>
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x220F;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mi>n</m:mi>
 </m:munderover>
 <m:msubsup>
  <m:mi>t</m:mi>
  <m:mi>k</m:mi>
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
```

```
     </m:mrow>
    </m:msubsup>
    <m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:msub>
        <m:mi>t</m:mi>
        <m:mi>k</m:mi>
       </m:msub>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:mi>b</m:mi>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
    </m:msup>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>+</m:mo>
         <m:mi>c</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi>n</m:mi>
    </m:msup>
   </m:mfrac>
```

```
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x220F;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mi>m</m:mi>
 </m:munderover>
 <m:mfrac>
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mo>+</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>n</m:mi>
      <m:mo>-</m:mo>
      <m:mi>k</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mi>c</m:mi>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mo>+</m:mo>
   <m:mi>b</m:mi>
   <m:mo>+</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>n</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mi>k</m:mi>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mi>c</m:mi>
   </m:mrow>
  </m:mrow>
 </m:mfrac>
 <m:mrow>
  <m:munderover>
   <m:mo movablelimits="false">&#x220F;</m:mo>
```

```
 <m:mrow>
  <m:mi>k</m:mi>
  <m:mo>=</m:mo>
  <m:mn>1</m:mn>
 </m:mrow>
 <m:mi>n</m:mi>
</m:munderover>
<m:mfrac>
 <m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mi>k</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>c</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>b</m:mi>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mi>k</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>c</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
```

```
        </m:mrow>
        <m:mrow>
         <m:mi mathvariant="normal">&#x0393;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mn>1</m:mn>
           <m:mo>+</m:mo>
           <m:mrow>
            <m:mi>k</m:mi>
            <m:mi>c</m:mi>
           </m:mrow>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
       </m:mrow>
       <m:mrow>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>a</m:mi>
          <m:mo>+</m:mo>
          <m:mi>b</m:mi>
          <m:mo>+</m:mo>
          <m:mrow>
           <m:mrow>
            <m:mo>(</m:mo>
            <m:mrow>
             <m:mrow>
              <m:mn>2</m:mn>
              <m:mi>n</m:mi>
             </m:mrow>
             <m:mo>-</m:mo>
             <m:mi>k</m:mi>
             <m:mo>-</m:mo>
             <m:mn>1</m:mn>
            </m:mrow>
            <m:mo>)</m:mo>
           </m:mrow>
           <m:mi>c</m:mi>
          </m:mrow>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>
```

```
</div>

<p>provided that
 <m:math display="inline">
  <m:mrow>
   <m:mi mathvariant="normal">&#x211C;</m:mi>
   <m:mi>a</m:mi>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>c</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mo>min</m:mo>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mfrac bevelled="true">
        <m:mn>1</m:mn>
        <m:mi>n</m:mi>
       </m:mfrac>
       <m:mo>,</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x211C;</m:mi>
        <m:mfrac bevelled="true">
         <m:mi>a</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>n</m:mi>
           <m:mo>-</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mfrac>
       </m:mrow>
       <m:mo>,</m:mo>
```

```
      <m:mrow>
       <m:mi mathvariant="normal">&#x211C;</m:mi>
       <m:mfrac bevelled="true">
        <m:mi>b</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>n</m:mi>
          <m:mo>-</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mfrac>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 </m:math>
</p>

<p>Secondly,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mo>&#x222B;</m:mo>
      <m:msup>
       <m:mrow>
        <m:mo>[</m:mo>
        <m:mrow>
         <m:mn>0</m:mn>
         <m:mo>,</m:mo>
         <m:mi mathvariant="normal">&#x221E;</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:msup>
     </m:msub>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mn>2</m:mn>
```

```
</m:msub>
<m:mi mathvariant="normal">&#x22EF;</m:mi>
<m:msub>
 <m:mi>t</m:mi>
 <m:mi>m</m:mi>
</m:msub>
<m:msup>
 <m:mrow>
  <m:mo>|</m:mo>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0394;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mo>,</m:mo>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
     <m:mo>,</m:mo>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mi>n</m:mi>
     </m:msub>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>|</m:mo>
 </m:mrow>
 <m:mrow>
  <m:mn>2</m:mn>
  <m:mi>c</m:mi>
 </m:mrow>
</m:msup>
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x220F;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mi>n</m:mi>
 </m:munderover>
 <m:msubsup>
  <m:mi>t</m:mi>
  <m:mi>k</m:mi>
  <m:mrow>
   <m:mi>a</m:mi>
   <m:mo>-</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
```

```
    </m:msubsup>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:msub>
       <m:mi>t</m:mi>
       <m:mi>k</m:mi>
      </m:msub>
     </m:mrow>
    </m:msup>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:munderover>
    <m:mo movablelimits="false">&#x220F;</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>=</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mi>m</m:mi>
   </m:munderover>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>a</m:mi>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mi>k</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>c</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mfrac>
    <m:mrow>
     <m:msubsup>
```

```
     <m:mo>&#x220F;</m:mo>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>=</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mi>n</m:mi>
    </m:msubsup>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mo></m:mo>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>n</m:mi>
          <m:mo>-</m:mo>
          <m:mi>k</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mi>c</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mi>c</m:mi>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
   <m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
```

```
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mn>1</m:mn>
           <m:mo>+</m:mo>
           <m:mi>c</m:mi>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:msup>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>when
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>c</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mo>min</m:mo>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mfrac bevelled="true">
        <m:mn>1</m:mn>
        <m:mi>n</m:mi>
       </m:mfrac>
       <m:mo>,</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x211C;</m:mi>
        <m:mfrac bevelled="true">
         <m:mi>a</m:mi>
```

```
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>n</m:mi>
           <m:mo>-</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mfrac>
       </m:mrow>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>.
</p>

<p>Thirdly,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>&#x03C0;</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mfrac bevelled="true">
        <m:mi>n</m:mi>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:msup>
     </m:mfrac>
     <m:mrow>
      <m:msub>
       <m:mo>&#x222B;</m:mo>
       <m:msup>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mrow>
           <m:mo>-</m:mo>
```

```
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mi>n</m:mi>
 </m:msup>
</m:msub>
<m:msup>
 <m:mrow>
  <m:mo>|</m:mo>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0394;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mn>1</m:mn>
     </m:msub>
     <m:mo>,</m:mo>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
     <m:mo>,</m:mo>
     <m:msub>
      <m:mi>t</m:mi>
      <m:mi>n</m:mi>
     </m:msub>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>|</m:mo>
 </m:mrow>
 <m:mrow>
  <m:mn>2</m:mn>
  <m:mi>c</m:mi>
 </m:mrow>
</m:msup>
<m:mrow>
 <m:munderover>
  <m:mo movablelimits="false">&#x220F;</m:mo>
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
  <m:mi>n</m:mi>
 </m:munderover>
 <m:mrow>
  <m:mi>exp</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
```

```
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mstyle>
       <m:msubsup>
        <m:mi>t</m:mi>
        <m:mi>k</m:mi>
        <m:mn>2</m:mn>
       </m:msubsup>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x2146;</m:mi>
    <m:msub>
     <m:mi>t</m:mi>
     <m:mi>k</m:mi>
    </m:msub>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mfrac>
 <m:mrow>
  <m:msubsup>
   <m:mo>&#x220F;</m:mo>
   <m:mrow>
    <m:mi>k</m:mi>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mi>n</m:mi>
  </m:msubsup>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mi>c</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
```

```
     </m:mrow>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mn>1</m:mn>
          <m:mo>+</m:mo>
          <m:mi>c</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>n</m:mi>
     </m:msup>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<h5>Dyson's Integral</h5>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>&#x03C0;</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:msup>
     </m:mfrac>
     <m:mrow>
      <m:msub>
       <m:mo>&#x222B;</m:mo>
       <m:msup>
        <m:mrow>
         <m:mo>[</m:mo>
         <m:mrow>
```

```
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>&#x03C0;</m:mi>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi>&#x03C0;</m:mi>
   </m:mrow>
   <m:mo>]</m:mo>
  </m:mrow>
  <m:mi>n</m:mi>
 </m:msup>
</m:msub>
<m:munder>
 <m:mo movablelimits="false">&#x220F;</m:mo>
 <m:mrow>
  <m:mn>1</m:mn>
  <m:mo>&#x2264;</m:mo>
  <m:mi>j</m:mi>
  <m:mo>&lt;</m:mo>
  <m:mi>k</m:mi>
  <m:mo>&#x2264;</m:mo>
  <m:mi>n</m:mi>
 </m:mrow>
</m:munder>
<m:msup>
 <m:mrow>
  <m:mo>|</m:mo>
  <m:mrow>
   <m:msup>
    <m:mi mathvariant="normal">&#x2147;</m:mi>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:msub>
      <m:mi>&#x03B8;</m:mi>
      <m:mi>j</m:mi>
     </m:msub>
    </m:mrow>
   </m:msup>
   <m:mo>-</m:mo>
   <m:msup>
    <m:mi mathvariant="normal">&#x2147;</m:mi>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2148;</m:mi>
     <m:msub>
      <m:mi>&#x03B8;</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
   </m:msup>
  </m:mrow>
  <m:mo>|</m:mo>
 </m:mrow>
 <m:mrow>
  <m:mn>2</m:mn>
```

```
    <m:mi>b</m:mi>
   </m:mrow>
  </m:msup>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:msub>
    <m:mi>&#x03B8;</m:mi>
    <m:mn>1</m:mn>
   </m:msub>
  </m:mrow>
  <m:mi mathvariant="normal">&#x22EF;</m:mi>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:msub>
    <m:mi>&#x03B8;</m:mi>
    <m:mi>n</m:mi>
   </m:msub>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mfrac>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mi>b</m:mi>
     <m:mi>n</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mo>+</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
```

```
      <m:mi>n</m:mi>
     </m:msup>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mrow>
    <m:mfrac bevelled="true">
     <m:mn>1</m:mn>
     <m:mi>n</m:mi>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>
\getchunk{page foot}
```

dlmfnotation.xhtml

— **dlmfnotation.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Digital Library of Mathematical Functions<br/>
                      The Gamma Function -- Notation
  </div>
  <hr/>
 <div class="content">
  <div class="section">
   <h3>Notation</h3>
   <div class="table" id="T1">
    <table align="center">
     <tbody>
      <tr>
       <th align="left">
        <m:math display="inline">
         <m:mrow>
          <m:mi>j</m:mi>
          <m:mo>,</m:mo>
```

```
          <m:mi>m</m:mi>
          <m:mo>,</m:mo>
          <m:mi>n</m:mi>
         </m:mrow>
        </m:math>
       </th>
       <td align="justify">nonnegative integers.</td>
      </tr>
      <tr>
       <th align="left">
        <m:math display="inline">
         <m:mi>k</m:mi>
        </m:math>
       </th>
       <td>except in <a href="dlmfphysicalapplications.xhtml">
                        Physical Applications</a>
       </td>
      </tr>
      <tr>
       <th align="left">
        <m:math display="inline">
         <m:mrow>
          <m:mi>x</m:mi>
          <m:mo>,</m:mo>
          <m:mi>y</m:mi>
         </m:mrow>
        </m:math>
       </th>
       <td align="justify">real variables.</td>
      </tr>
      <tr>
       <th align="left">
        <m:math display="inline">
         <m:mrow>
          <m:mi>z</m:mi>
          <m:mo>=</m:mo>
          <m:mrow>
           <m:mi>x</m:mi>
           <m:mo>+</m:mo>
           <m:mrow>
            <m:mi mathvariant="normal">&#x2148;</m:mi>
            <m:mi>y</m:mi>
           </m:mrow>
          </m:mrow>
         </m:mrow>
        </m:math>
       </th>
      <td align="justify">complex variable.</td>
     </tr>
     <tr>
      <th align="left">
       <m:math display="inline">
        <m:mrow>
         <m:mi>a</m:mi>
```

```
    <m:mo>,</m:mo>
    <m:mi>b</m:mi>
    <m:mo>,</m:mo>
    <m:mi>q</m:mi>
    <m:mo>,</m:mo>
    <m:mi>s</m:mi>
    <m:mo>,</m:mo>
    <m:mi>w</m:mi>
   </m:mrow>
  </m:math>
 </th>
 <td align="justify">real or complex variables with
  <m:math display="inline">
   <m:mrow>
    <m:mrow>
     <m:mo>&#x2223;</m:mo>
     <m:mi>q</m:mi>
     <m:mo>&#x2223;</m:mo>
    </m:mrow>
    <m:mo>&lt;</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:math>.
 </td>
</tr>
<tr>
 <th align="left">
  <m:math display="inline">
   <m:mi>&#x03B4;</m:mi>
  </m:math>
 </th>
 <td align="justify">arbitrary small positive constant.</td>
</tr>
<tr>
 <th align="left">
  <m:math display="inline">
   <m:mi mathvariant="normal">&#x2102;</m:mi>
  </m:math>
 </th>
 <td align="justify">complex plane (excluding infinity).</td>
</tr>
<tr>
 <th align="left">
  <m:math display="inline">
   <m:mi mathvariant="normal">&#x211D;</m:mi>
  </m:math>
 </th>
 <td align="justify">real line (excluding infinity).</td>
</tr>
<tr>
 <th align="left">
  <m:math display="inline">
   <m:mrow>
    <m:mo>(</m:mo>
```

```
    <m:mstyle scriptlevel="+1">
     <m:mtable rowspacing="0.2ex" columnspacing="0.4em">
      <m:mtr>
       <m:mtd>
        <m:mi>n</m:mi>
       </m:mtd>
      </m:mtr>
      <m:mtr>
       <m:mtd>
        <m:mi>m</m:mi>
       </m:mtd>
      </m:mtr>
     </m:mtable>
    </m:mstyle>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:math>
 </th>
 <td align="justify">binomial coefficient
  <m:math display="inline">
   <m:mfrac>
    <m:mrow>
     <m:mi>n</m:mi>
     <m:mi mathvariant="normal">!</m:mi>
    </m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi>m</m:mi>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>-</m:mo>
        <m:mi>m</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mfrac>
  </m:math>.
 </td>
</tr>
<tr>
 <th align="left">empty sums</th>
 <td align="justify">zero.</td>
</tr>
<tr>
 <th align="left">empty products</th>
 <td align="justify">unity.</td>
```

```
   </tr>
  </tbody>
 </table>
</div>

<div class="para" id="p1">
 <p>The main functions treated in this chapter are the gamma function
  <m:math display="inline">
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:math>,the psi function
  <m:math display="inline">
   <m:mrow>
    <m:mi>&#x03C8;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:math>,the beta function
  <m:math display="inline">
   <m:mrow>
    <m:mi mathvariant="normal">B</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>,</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:math>, and the
  <m:math display="inline">
   <m:mi>q</m:mi>
  </m:math>-gamma function
  <m:math display="inline">
   <m:mrow>
    <m:msub>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mi>q</m:mi>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
```

```
    </m:mrow>
   </m:mrow>
  </m:math>.
 </p>
</div>

<div class="para" id="p2">
 <p>The notation
  <m:math display="inline">
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:math> is due to Legendre. Alternative notations for this function are:
  <m:math display="inline">
   <m:mrow>
    <m:mi mathvariant="normal">&#x03A0;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:math> (Gauss) and
  <m:math display="inline">
   <m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>z</m:mi>
      <m:mo>-</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
    <m:mi mathvariant="normal">!</m:mi>
   </m:mrow>
  </m:math>. Alternative notations for the psi function are:
 </p>
</div>

<div class="table" id="T2">
 <table align="center">
  <thead>
   <tr>
    <th align="left">
```

```
   <m:math display="inline">
    <m:mrow>
     <m:mi mathvariant="normal">&#x03A8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:math>
  </th>
  <th align="left">Gauss;
    <a href="http://dlmf.nist.gov/Contents/bib/J#jahnke:1945:tof">
     Jahnke and Emde(1945)
    </a>
  </th>
 </tr>
</thead>
<tbody>
 <tr>
  <th align="left">
   <m:math display="inline">
    <m:mrow>
     <m:mi>&#x03A8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:math>
  </th>
  <td align="left">
    <a href="http://dlmf.nist.gov/Contents/bib/W#whittaker:1927:cma">
     Whittaker and Watson(1927)
    </a>
  </td>
 </tr>
 <tr>
  <th align="left">
   <m:math display="inline">
    <m:mrow>
     <m:mi mathvariant="normal">&#x03A8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:math>
```

```
      </th>
      <td align="left">
        <a href="http://dlmf.nist.gov/Contents/bib/D#davis:1933:thm">
          Davis(1933)
        </a>
      </td>
     </tr>
     <tr>
      <th align="left">
       <m:math display="inline">
        <m:mrow>
         <m:mi mathvariant="sans-serif">F</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>z</m:mi>
           <m:mo>-</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
       </m:math>
      </th>
      <td align="left">
        <a href="http://dlmf.nist.gov/Contents/bib/P#pairman:1919:tdt">
         Pairman(1919)
        </a>
      </td>
     </tr>
    </tbody>
   </table>
  </div>
 </div>
</div>
\getchunk{page foot}
```

dlmfphysicalapplications.xhtml

— dlmfphysicalapplications.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Physical Applications
  </div>
```

```
  <hr/>
<h3>Physical Applications</h3>

<p>Suppose the potential energy of a gas of
 <m:math display="inline">
  <m:mi>n</m:mi>
 </m:math> point charges with positions
 <m:math display="inline">
  <m:mrow>
   <m:msub>
    <m:mi>x</m:mi>
    <m:mn>1</m:mn>
   </m:msub>
   <m:mo>,</m:mo>
   <m:msub>
    <m:mi>x</m:mi>
    <m:mn>2</m:mn>
   </m:msub>
   <m:mo>,</m:mo>
   <m:mi mathvariant="normal">&#x2026;</m:mi>
   <m:mo>,</m:mo>
   <m:msub>
    <m:mi>x</m:mi>
    <m:mi>n</m:mi>
   </m:msub>
  </m:mrow>
 </m:math> and free to move on the infinite line
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mi>x</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math>, is given by
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mi>W</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mrow>
```

```
       <m:munderover>
        <m:mo movablelimits="false">&#x2211;</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2113;</m:mi>
         <m:mo>=</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mi>n</m:mi>
       </m:munderover>
       <m:msubsup>
        <m:mi>x</m:mi>
        <m:mi mathvariant="normal">&#x2113;</m:mi>
        <m:mn>2</m:mn>
       </m:msubsup>
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:munder>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>&#x2264;</m:mo>
        <m:mi mathvariant="normal">&#x2113;</m:mi>
        <m:mo>&lt;</m:mo>
        <m:mi>j</m:mi>
        <m:mo>&#x2264;</m:mo>
        <m:mi>n</m:mi>
       </m:mrow>
      </m:munder>
      <m:mi>ln</m:mi>
      <m:mrow>
       <m:mo>|</m:mo>
       <m:mrow>
        <m:msub>
         <m:mi>x</m:mi>
         <m:mi mathvariant="normal">&#x2113;</m:mi>
        </m:msub>
        <m:mo>-</m:mo>
        <m:msub>
         <m:mi>x</m:mi>
         <m:mi>j</m:mi>
        </m:msub>
       </m:mrow>
       <m:mo>|</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>The probability density of the positions when the gas is in thermodynamic
```

```
   equilibrium is:
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>P</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:msub>
        <m:mi>x</m:mi>
        <m:mn>1</m:mn>
       </m:msub>
       <m:mo>,</m:mo>
       <m:mi mathvariant="normal">&#x2026;</m:mi>
       <m:mo>,</m:mo>
       <m:msub>
        <m:mi>x</m:mi>
        <m:mi>n</m:mi>
       </m:msub>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi>C</m:mi>
     <m:mrow>
      <m:mi>exp</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mfrac bevelled="true">
         <m:mi>W</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>k</m:mi>
           <m:mi>T</m:mi>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mfrac>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
```

```
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mi>k</m:mi>
 </m:math> is the Boltzmann constant,
 <m:math display="inline">
  <m:mi>T</m:mi>
 </m:math> the temperature and
 <m:math display="inline">
  <m:mi>C</m:mi>
 </m:math> a constant.
 Then the partition function (with
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03B2;</m:mi>
   <m:mo>=</m:mo>
   <m:mfrac bevelled="true">
    <m:mn>1</m:mn>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mi>T</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mfrac>
  </m:mrow>
 </m:math>) is given by
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>&#x03C8;</m:mi>
      <m:mi>n</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>&#x03B2;</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msub>
      <m:mo>&#x222B;</m:mo>
      <m:msup>
       <m:mi mathvariant="normal">&#x211D;</m:mi>
```

```
      <m:mi>n</m:mi>
     </m:msup>
    </m:msub>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mi>&#x03B2;</m:mi>
       <m:mi>W</m:mi>
      </m:mrow>
     </m:mrow>
    </m:msup>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2146;</m:mi>
     <m:mi>x</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>&#x03C0;</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mfrac bevelled="true">
        <m:mi>n</m:mi>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:msup>
      <m:msup>
       <m:mi>&#x03B2;</m:mi>
       <m:mrow>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mfrac bevelled="true">
           <m:mi>n</m:mi>
           <m:mn>2</m:mn>
          </m:mfrac>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mfrac bevelled="true">
```

```
      <m:mrow>
       <m:mi>&#x03B2;</m:mi>
       <m:mi>n</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>n</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mn>4</m:mn>
     </m:mfrac>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:msup>
 </m:mrow>
 <m:mo>&#x00D7;</m:mo>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mstyle>
       <m:mi>&#x03B2;</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi>n</m:mi>
  </m:mrow>
 </m:msup>
</m:mrow>
<m:mrow>
 <m:munderover>
```

```
       <m:mo movablelimits="false">&#x220F;</m:mo>
       <m:mrow>
        <m:mi>j</m:mi>
        <m:mo>=</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:munderover>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mstyle displaystyle="false">
          <m:mfrac>
           <m:mn>1</m:mn>
           <m:mn>2</m:mn>
          </m:mfrac>
         </m:mstyle>
         <m:mi>j</m:mi>
         <m:mi>&#x03B2;</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>For
 <m:math display="inline">
  <m:mi>n</m:mi>
 </m:math> charges free to move on a circular wire of radius
 <m:math display="inline">
  <m:mn>1</m:mn>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mi>W</m:mi>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:munder>
       <m:mo movablelimits="false">&#x2211;</m:mo>
```

```
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>&#x2264;</m:mo>
        <m:mi mathvariant="normal">&#x2113;</m:mi>
        <m:mo>&lt;</m:mo>
        <m:mi>j</m:mi>
        <m:mo>&#x2264;</m:mo>
        <m:mi>n</m:mi>
       </m:mrow>
      </m:munder>
      <m:mi>ln</m:mi>
      <m:mrow>
       <m:mo>|</m:mo>
       <m:mrow>
        <m:msup>
         <m:mi mathvariant="normal">&#x2147;</m:mi>
         <m:mrow>
          <m:mi mathvariant="normal">&#x2148;</m:mi>
          <m:msub>
           <m:mi>&#x03B8;</m:mi>
           <m:mi mathvariant="normal">&#x2113;</m:mi>
          </m:msub>
         </m:mrow>
        </m:msup>
        <m:mo>-</m:mo>
        <m:msup>
         <m:mi mathvariant="normal">&#x2147;</m:mi>
         <m:mrow>
          <m:mi mathvariant="normal">&#x2148;</m:mi>
          <m:msub>
           <m:mi>&#x03B8;</m:mi>
           <m:mi>j</m:mi>
          </m:msub>
         </m:mrow>
        </m:msup>
       </m:mrow>
       <m:mo>|</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>and the partition function is given by</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi>&#x03C8;</m:mi>
```

```
    <m:mi>n</m:mi>
   </m:msub>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>&#x03B2;</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>&#x03C0;</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi>n</m:mi>
    </m:msup>
   </m:mfrac>
   <m:mrow>
    <m:msub>
     <m:mo>&#x222B;</m:mo>
     <m:msup>
      <m:mrow>
       <m:mo>[</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mi>&#x03C0;</m:mi>
        </m:mrow>
        <m:mo>,</m:mo>
        <m:mi>&#x03C0;</m:mi>
       </m:mrow>
       <m:mo>]</m:mo>
      </m:mrow>
      <m:mi>n</m:mi>
     </m:msup>
    </m:msub>
    <m:msup>
     <m:mi mathvariant="normal">&#x2147;</m:mi>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mi>&#x03B2;</m:mi>
       <m:mi>W</m:mi>
      </m:mrow>
     </m:mrow>
    </m:msup>
    <m:mrow>
```

```
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:msub>
    <m:mi>&#x03B8;</m:mi>
    <m:mn>1</m:mn>
   </m:msub>
  </m:mrow>
  <m:mi mathvariant="normal">&#x22EF;</m:mi>
  <m:mrow>
   <m:mi mathvariant="normal">&#x2146;</m:mi>
   <m:msub>
    <m:mi>&#x03B8;</m:mi>
    <m:mi>n</m:mi>
   </m:msub>
  </m:mrow>
 </m:mrow>
</m:mrow>
<m:mo>=</m:mo>
<m:mrow>
 <m:mrow>
  <m:mi mathvariant="normal">&#x0393;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>+</m:mo>
    <m:mrow>
     <m:mstyle displaystyle="false">
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:mstyle>
     <m:mi>n</m:mi>
     <m:mi>&#x03B2;</m:mi>
    </m:mrow>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:msup>
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
```

```
           </m:mstyle>
           <m:mi>&#x03B2;</m:mi>
          </m:mrow>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>n</m:mi>
      </m:mrow>
     </m:msup>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

———-

dlmfpolygammafunctions.xhtml

— dlmfpolygammafunctions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Polygamma Functions
  </div>
  <hr/>
<h3>Polygamma Functions</h3>

<p>The functions
 <m:math display="inline">
  <m:mrow>
   <m:msup>
    <m:mi>&#x03C8;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>n</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
```

```
   <m:mi>z</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math>,
<m:math display="inline">
 <m:mrow>
  <m:mi>n</m:mi>
  <m:mo>=</m:mo>
  <m:mrow>
   <m:mn>1</m:mn>
   <m:mo>,</m:mo>
   <m:mn>2</m:mn>
   <m:mo>,</m:mo>
   <m:mi mathvariant="normal">&#x2026;</m:mi>
  </m:mrow>
 </m:mrow>
</m:math>, are called the <em>polygamma functions</em>. In particular,
<m:math display="inline">
 <m:mrow>
  <m:msup>
   <m:mi>&#x03C8;</m:mi>
   <m:mo>&#x2032;</m:mo>
  </m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mi>z</m:mi>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
</m:math> is the <em>trigamma function</em>;
<m:math display="inline">
 <m:msup>
  <m:mi>&#x03C8;</m:mi>
  <m:mrow>
   <m:mi>&#x2032;</m:mi>
   <m:mi>&#x2032;</m:mi>
  </m:mrow>
 </m:msup></m:math>,
<m:math display="inline">
 <m:msup>
  <m:mi>&#x03C8;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mn>3</m:mn>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:msup>
</m:math>,
<m:math display="inline">
 <m:msup>
  <m:mi>&#x03C8;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
```

```
    <m:mn>4</m:mn>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:msup>
 </m:math> are the <em>tetra-,</em> <em>penta-,</em> and
 <em>hexagamma functions</em> respectively. Most properties of these
 functions follow straightforwardly by differentiation of properties
 of the psi function. This includes asymptotic expansions.
</p>

<p>In the second and third equations,
 <m:math display="inline">
  <m:mrow>
   <m:mi>n</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>,</m:mo>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mn>3</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>; for
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03B6;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>n</m:mi>
     <m:mo>+</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:math>
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi>&#x03C8;</m:mi>
      <m:mo>&#x2032;</m:mo>
     </m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
```

```
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>0</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>+</m:mo>
         <m:mi>z</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mn>2</m:mn>
      </m:msup>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
```

```
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>n</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mn>1</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi>n</m:mi>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
     <m:mrow>
      <m:mi>&#x03B6;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
```

```
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>n</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
     <m:mrow>
      <m:mi>n</m:mi>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
```

```
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:msup>
        <m:mn>2</m:mn>
        <m:mrow>
         <m:mi>n</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mrow>
      <m:mi>&#x03B6;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>n</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi>&#x03C8;</m:mi>
      <m:mo>&#x2032;</m:mo>
     </m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mstyle>
```

```
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mstyle displaystyle="false">
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:mstyle>
     <m:msup>
      <m:mi>&#x03C0;</m:mi>
      <m:mn>2</m:mn>
     </m:msup>
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mn>4</m:mn>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:munderover>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:msup>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mrow>
           <m:mn>2</m:mn>
           <m:mi>k</m:mi>
          </m:mrow>
          <m:mo>-</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mn>2</m:mn>
       </m:msup>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
```

```
 </m:math>
</div>

<p>As
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2192;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math> in
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mrow>
     <m:mi>ph</m:mi>
     <m:mspace width="0.2em"/>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2264;</m:mo>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C0;</m:mi>
     <m:mo>-</m:mo>
     <m:mi>&#x03B4;</m:mi>
    </m:mrow>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:none/>
      <m:mo>&lt;</m:mo>
      <m:mi>&#x03C0;</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi>&#x03C8;</m:mi>
      <m:mo>&#x2032;</m:mo>
     </m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
```

```
    <m:mi>z</m:mi>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mo>&#x223C;</m:mo>
  <m:mrow>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mi>z</m:mi>
   </m:mfrac>
   <m:mo>+</m:mo>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:msup>
      <m:mi>z</m:mi>
      <m:mn>2</m:mn>
     </m:msup>
    </m:mrow>
   </m:mfrac>
   <m:mo>+</m:mo>
   <m:mrow>
    <m:munderover>
     <m:mo movablelimits="false">&#x2211;</m:mo>
     <m:mrow>
      <m:mi>k</m:mi>
      <m:mo>=</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:munderover>
    <m:mfrac>
     <m:msub>
      <m:mi>B</m:mi>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>k</m:mi>
      </m:mrow>
     </m:msub>
     <m:msup>
      <m:mi>z</m:mi>
      <m:mrow>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>k</m:mi>
       </m:mrow>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:msup>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
```

```
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmfqgammaandbetafunctions.xhtml

— dlmfqgammaandbetafunctions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- q-Gamma and Beta Functions
  </div>
  <hr/>
<h3>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">q</m:mi>
 </m:math>-Gamma and Beta Functions
</h3>

<ul>
 <li>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">q</m:mi>
 </m:math>-Factorials</li>
 <li>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">q</m:mi>
 </m:math>-Gamma Function</li>
 <li>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">q</m:mi>
 </m:math>-Beta Function</li>
</ul>

<h4>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">q</m:mi>
 </m:math>-Factorials</h4>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
```

```
    <m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mspace width="0.2em"/>
       <m:mo>;</m:mo>
       <m:mi>q</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi>n</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x220F;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>0</m:mn>
      </m:mrow>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
     </m:munderover>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:msup>
        <m:mi>q</m:mi>
        <m:mi>k</m:mi>
       </m:msup>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>n</m:mi>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
```

```
    <m:mo>,</m:mo>
    <m:mn>1</m:mn>
    <m:mo>,</m:mo>
    <m:mn>2</m:mn>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mrow>
      <m:mi>n</m:mi>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
     <m:mi>q</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>1</m:mn>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mi>q</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi mathvariant="normal">&#x22EF;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mi>q</m:mi>
       <m:mo>+</m:mo>
       <m:mi mathvariant="normal">&#x2026;</m:mi>
       <m:mo>+</m:mo>
       <m:msup>
        <m:mi>q</m:mi>
        <m:mrow>
         <m:mi>n</m:mi>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:msup>
      </m:mrow>
      <m:mo>)</m:mo>
```

```
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>q</m:mi>
        <m:mspace width="0.2em"/>
        <m:mo>;</m:mo>
        <m:mi>q</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>n</m:mi>
     </m:msub>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mi>q</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>n</m:mi>
      </m:mrow>
     </m:msup>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>When
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mi>q</m:mi>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
```

```
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:mspace width="0.2em"/>
       <m:mo>;</m:mo>
       <m:mi>q</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mi mathvariant="normal">&#x221E;</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x220F;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>0</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mn>1</m:mn>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
       <m:msup>
        <m:mi>q</m:mi>
        <m:mi>k</m:mi>
       </m:msup>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<h4>
 <m:math display="inline">
 <m:mi mathvariant="bold-italic">q</m:mi></m:math>-Gamma Function</h4>

<p>When
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
```

```
   <m:mi>q</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac bevelled="true">
     <m:mrow>
      <m:msub>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>q</m:mi>
         <m:mspace width="0.2em"/>
         <m:mo>;</m:mo>
         <m:mi>q</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:msub>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>1</m:mn>
         <m:mo>-</m:mo>
         <m:mi>q</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
      </m:msup>
```

```
     </m:mrow>
     <m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:msup>
         <m:mi>q</m:mi>
         <m:mi>z</m:mi>
        </m:msup>
        <m:mspace width="0.2em"/>
        <m:mo>;</m:mo>
        <m:mi>q</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:msub>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mn>1</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mn>2</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:mrow>
```

```
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mrow>
      <m:mi>n</m:mi>
      <m:mi mathvariant="normal">!</m:mi>
     </m:mrow>
     <m:mi>q</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>z</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
```

```
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:msup>
        <m:mi>q</m:mi>
        <m:mi>z</m:mi>
       </m:msup>
      </m:mrow>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>-</m:mo>
       <m:mi>q</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mrow>
      <m:msub>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mi>q</m:mi>
      </m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>Also,
 <m:math display="inline">
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:msub>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mi>q</m:mi>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>x</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math> is convex for
 <m:math display="inline">
  <m:mrow>
   <m:mi>x</m:mi>
```

```
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>, and the analog of the
 <a href="dlmffunctionrelations.xhtml#bohrmolleruptheorem">
  Bohr-Mollerup theorem
 </a> holds.
</p>

<p>If
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>q</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mi>r</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>, then
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>x</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>&lt;</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>r</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>x</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
```

```
<p>when
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>x</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math> or when
 <m:math display="inline">
  <m:mrow>
   <m:mi>x</m:mi>
   <m:mo>&gt;</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:math>, and
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>x</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>&gt;</m:mo>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mi>r</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>x</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>when
 <m:math display="inline">
```

```
  <m:mrow>
   <m:mn>1</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>x</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:math>.
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:munder>
      <m:mo movablelimits="false">lim</m:mo>
      <m:mrow>
       <m:mi>q</m:mi>
       <m:mo>&#x2192;</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>-</m:mo>
       </m:mrow>
      </m:mrow>
     </m:munder>
     <m:mrow>
      <m:msub>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mi>q</m:mi>
      </m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>For generalized asymptotic expansions of
 <m:math display="inline">
```

```
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mspace width="0.2em"/>
   <m:mrow>
    <m:msub>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mi>q</m:mi>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mi>z</m:mi>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math> as
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mi>z</m:mi>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&#x2192;</m:mo>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math> see
 <a href="http://dlmf.nist.gov/Contents/bib/O#oldedaalhuis:1994:aef">
  Olde Daalhuis(1994)
 </a> and
 <a href="http://dlmf.nist.gov/Contents/bib/M#moak:1984:tqa">
  Moak(1984)
 </a>.
</p>

<h4>
 <m:math display="inline">
  <m:mi mathvariant="bold-italic">q</m:mi>
 </m:math>-Beta Function
</h4>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msub>
      <m:mi mathvariant="normal">B</m:mi>
      <m:mi>q</m:mi>
     </m:msub>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>a</m:mi>
```

```
       <m:mo>,</m:mo>
       <m:mi>b</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mrow>
      <m:mrow>
       <m:msub>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mi>q</m:mi>
       </m:msub>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>a</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:msub>
        <m:mi mathvariant="normal">&#x0393;</m:mi>
        <m:mi>q</m:mi>
       </m:msub>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>b</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:msub>
       <m:mi mathvariant="normal">&#x0393;</m:mi>
       <m:mi>q</m:mi>
      </m:msub>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>+</m:mo>
        <m:mi>b</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
```

```
<m:math display="block">
 <m:mrow>
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi mathvariant="normal">B</m:mi>
     <m:mi>q</m:mi>
    </m:msub>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mi>a</m:mi>
      <m:mo>,</m:mo>
      <m:mi>b</m:mi>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:msubsup>
     <m:mo>&#x222B;</m:mo>
     <m:mn>0</m:mn>
     <m:mn>1</m:mn>
    </m:msubsup>
    <m:mfrac>
     <m:mrow>
      <m:msup>
       <m:mi>t</m:mi>
       <m:mrow>
        <m:mi>a</m:mi>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:msup>
      <m:msub>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>t</m:mi>
          <m:mi>q</m:mi>
         </m:mrow>
         <m:mspace width="0.2em"/>
         <m:mo>;</m:mo>
         <m:mi>q</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:msub>
     </m:mrow>
     <m:msub>
      <m:mrow>
```

```
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>t</m:mi>
          <m:msup>
           <m:mi>q</m:mi>
           <m:mi>b</m:mi>
          </m:msup>
         </m:mrow>
         <m:mo>;</m:mo>
         <m:mi>q</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:msub>
     </m:mfrac>
     <m:mrow>
      <m:msub>
       <m:mi mathvariant="normal">&#x2146;</m:mi>
       <m:mi>q</m:mi>
      </m:msub>
      <m:mi>t</m:mi>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>q</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
    <m:mi>a</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mi mathvariant="normal">&#x211C;</m:mi>
```

```
    <m:mi>b</m:mi>
   </m:mrow>
   <m:mo>&gt;</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>.
</div>
\getchunk{page foot}
```

dlmfseriesexpansions.xhtml

— dlmfseriesexpansions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Series Expansions
  </div>
  <hr/>
<h3>Series Expansions</h3>
<h6>Contents</h6>
<ul>
 <li>Maclaurin Series</li>
 <li>Other Series</li>
</ul>
<h4>Maclaurin Series</h4>
<p>Throughout this subsection
 <m:math display="inline">
  <m:mrow>
   <m:mi>&#x03B6;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mi>k</m:mi>
    <m:mo>)</m:mo>
  </m:mrow>
  </m:mrow>
 </m:math> is
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mfrac>
     <m:mn>1</m:mn>
     <m:mrow>
```

```
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>z</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:msub>
      <m:mi>c</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
     <m:msup>
      <m:mi>z</m:mi>
      <m:mi>k</m:mi>
     </m:msup>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>where
 <m:math display="inline">
  <m:mrow>
   <m:msub>
    <m:mi>c</m:mi>
    <m:mn>1</m:mn>
   </m:msub>
   <m:mo>=</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,

 <m:math display="inline">
  <m:mrow>
   <m:msub>
    <m:mi>c</m:mi>
    <m:mn>2</m:mn>
   </m:msub>
   <m:mo>=</m:mo>
   <m:mi>&#x03B3;</m:mi>
  </m:mrow>
```

```
 </m:math>, and
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>-</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:msub>
      <m:mi>c</m:mi>
      <m:mi>k</m:mi>
     </m:msub>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mrow>
        <m:mrow>
         <m:mi>&#x03B3;</m:mi>
         <m:msub>
          <m:mi>c</m:mi>
          <m:mrow>
           <m:mi>k</m:mi>
           <m:mo>-</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
         </m:msub>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mrow>
          <m:mi>&#x03B6;</m:mi>
          <m:mrow>
           <m:mo>(</m:mo>
           <m:mn>2</m:mn>
           <m:mo>)</m:mo>
          </m:mrow>
         </m:mrow>
         <m:msub>
          <m:mi>c</m:mi>
          <m:mrow>
           <m:mi>k</m:mi>
           <m:mo>-</m:mo>
           <m:mn>2</m:mn>
```

```
     </m:mrow>
    </m:msub>
   </m:mrow>
  </m:mrow>
  <m:mo>+</m:mo>
  <m:mrow>
   <m:mrow>
    <m:mi>&#x03B6;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mn>3</m:mn>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:msub>
    <m:mi>c</m:mi>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>-</m:mo>
     <m:mn>3</m:mn>
    </m:mrow>
   </m:msub>
  </m:mrow>
 </m:mrow>
 <m:mo>-</m:mo>
 <m:mi mathvariant="normal">&#x2026;</m:mi>
</m:mrow>
<m:mo>+</m:mo>
<m:mrow>
 <m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
  <m:mi>k</m:mi>
 </m:msup>
 <m:mrow>
  <m:mi>&#x03B6;</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mi>k</m:mi>
    <m:mo>-</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:msub>
  <m:mi>c</m:mi>
```

```
       <m:mn>1</m:mn>
      </m:msub>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mi>k</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mn>3</m:mn>
  </m:mrow>
 </m:math>.
</div>

<p>For 15D numerical values of
 <m:math display="inline">
  <m:msub>
   <m:mi>c</m:mi>
   <m:mi>k</m:mi>
  </m:msub>
 </m:math> see
 <a href="http://dlmf.nist.gov/Contents/bib/#abramowitz:1964:hmf">
 Abramowitz and Stegun(1964)</a>(p.256), and
for 31D values see
<a href="http://dlmf.nist.gov/Contents/bib/W#wrench:1968:cts">
   Wrench(1968)</a>.
</p>

<a name="equation3"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>ln</m:mi>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
```

```
<m:mrow>
 <m:mrow>
  <m:mo>-</m:mo>
  <m:mrow>
   <m:mi>ln</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mn>1</m:mn>
     <m:mo>+</m:mo>
     <m:mi>z</m:mi>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mo>+</m:mo>
 <m:mrow>
  <m:mi>z</m:mi>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mn>1</m:mn>
    <m:mo>-</m:mo>
    <m:mi>&#x03B3;</m:mi>
   </m:mrow>
   <m:mo>)</m:mo>
  </m:mrow>
 </m:mrow>
 <m:mo>+</m:mo>
 <m:mrow>
  <m:munderover>
   <m:mo movablelimits="false">&#x2211;</m:mo>
   <m:mrow>
    <m:mi>k</m:mi>
    <m:mo>=</m:mo>
    <m:mn>2</m:mn>
   </m:mrow>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:munderover>
  <m:msup>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
   <m:mi>k</m:mi>
  </m:msup>
  <m:mrow>
   <m:mo>(</m:mo>
   <m:mrow>
```

```
        <m:mrow>
         <m:mi>&#x03B6;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mi>k</m:mi>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mfrac>
       <m:msup>
        <m:mi>z</m:mi>
        <m:mi>k</m:mi>
       </m:msup>
       <m:mi>k</m:mi>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mi>z</m:mi>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:math>.
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mi>z</m:mi>
      </m:mrow>
```

```
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>&#x03B3;</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>2</m:mn>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:munderover>
      <m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mo>-</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
       <m:mi>k</m:mi>
      </m:msup>
      <m:mrow>
       <m:mi>&#x03B6;</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mi>k</m:mi>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:msup>
       <m:mi>z</m:mi>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:msup>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
```

```
<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mi>z</m:mi>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>,
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mn>1</m:mn>
       <m:mo>+</m:mo>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mi>z</m:mi>
        </m:mrow>
       </m:mfrac>
       <m:mo>-</m:mo>
       <m:mrow>
        <m:mfrac>
         <m:mi>&#x03C0;</m:mi>
         <m:mn>2</m:mn>
        </m:mfrac>
        <m:mrow>
         <m:mi>cot</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>&#x03C0;</m:mi>
           <m:mi>z</m:mi>
```

```
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
       </m:mrow>
      </m:mrow>
      <m:mo>+</m:mo>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mrow>
        <m:msup>
         <m:mi>z</m:mi>
         <m:mn>2</m:mn>
        </m:msup>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
      </m:mfrac>
      <m:mo>+</m:mo>
      <m:mn>1</m:mn>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mi>&#x03B3;</m:mi>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:munderover>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi>&#x03B6;</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mrow>
            <m:mn>2</m:mn>
            <m:mi>k</m:mi>
           </m:mrow>
           <m:mo>+</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
```

```
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:msup>
       <m:mi>z</m:mi>
       <m:mrow>
        <m:mn>2</m:mn>
        <m:mi>k</m:mi>
       </m:mrow>
      </m:msup>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="right">
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:mo>|</m:mo>
    <m:mi>z</m:mi>
    <m:mo>|</m:mo>
   </m:mrow>
   <m:mo>&lt;</m:mo>
   <m:mn>2</m:mn>
  </m:mrow>
 </m:math>,
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>&#x00B1;</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>.
</div>

<p>For 20D numerical values of the coefficients of the Maclaurin series for
 <m:math display="inline">
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mi>z</m:mi>
     <m:mo>+</m:mo>
```

```
     <m:mn>3</m:mn>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
</m:math> see
<a href="http://dlmf.nist.gov/Contents/bib/L#luke:1969:sfa2">
   Luke(1969)</a>(p.299).
</p>

<p>When
 <m:math display="inline">
  <m:mrow>
   <m:mi>z</m:mi>
   <m:mo>&#x2260;</m:mo>
   <m:mrow>
    <m:mn>0</m:mn>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>2</m:mn>
    </m:mrow>
    <m:mo>,</m:mo>
    <m:mi mathvariant="normal">&#x2026;</m:mi>
   </m:mrow>
  </m:mrow>
 </m:math>,
</p>

<a name="equation6"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mi>z</m:mi>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>&#x03B3;</m:mi>
      </m:mrow>
```

```
   <m:mo>-</m:mo>
   <m:mfrac>
    <m:mn>1</m:mn>
    <m:mi>z</m:mi>
   </m:mfrac>
  </m:mrow>
  <m:mo>+</m:mo>
  <m:mrow>
   <m:munderover>
    <m:mo movablelimits="false">&#x2211;</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>=</m:mo>
     <m:mn>1</m:mn>
    </m:mrow>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:munderover>
   <m:mfrac>
    <m:mi>z</m:mi>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>+</m:mo>
       <m:mi>z</m:mi>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mfrac>
  </m:mrow>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:mrow>
  <m:mrow>
   <m:mo>-</m:mo>
   <m:mi>&#x03B3;</m:mi>
  </m:mrow>
  <m:mo>+</m:mo>
  <m:mrow>
   <m:munderover>
    <m:mo movablelimits="false">&#x2211;</m:mo>
    <m:mrow>
     <m:mi>k</m:mi>
     <m:mo>=</m:mo>
     <m:mn>0</m:mn>
    </m:mrow>
    <m:mi mathvariant="normal">&#x221E;</m:mi>
   </m:munderover>
   <m:mo>(</m:mo>
   <m:mrow>
    <m:mfrac>
```

```
        <m:mn>1</m:mn>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
       </m:mfrac>
       <m:mo>-</m:mo>
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>+</m:mo>
         <m:mi>z</m:mi>
        </m:mrow>
       </m:mfrac>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>and
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mfrac>
        <m:mrow>
         <m:mi>z</m:mi>
         <m:mo>+</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mn>2</m:mn>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mfrac>
```

```
        <m:mi>z</m:mi>
        <m:mn>2</m:mn>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>0</m:mn>
       </m:mrow>
       <m:mi mathvariant="normal">&#x221E;</m:mi>
      </m:munderover>
      <m:mfrac>
       <m:msup>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mo>-</m:mo>
          <m:mn>1</m:mn>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
        <m:mi>k</m:mi>
       </m:msup>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>+</m:mo>
        <m:mi>z</m:mi>
       </m:mrow>
      </m:mfrac>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>Also,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x2111;</m:mi>
```

```
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:mfrac>
      <m:mi>y</m:mi>
      <m:mrow>
       <m:msup>
        <m:mi>k</m:mi>
        <m:mn>2</m:mn>
       </m:msup>
       <m:mo>+</m:mo>
       <m:msup>
        <m:mi>y</m:mi>
        <m:mn>2</m:mn>
       </m:msup>
      </m:mrow>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmfsums.xhtml

— dlmfsums.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Sums
  </div>
  <hr/>
<h3>Sums</h3>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mo>-</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
      <m:mi>k</m:mi>
     </m:msup>
     <m:mrow>
      <m:msup>
       <m:mi>&#x03C8;</m:mi>
       <m:mo>&#x2032;</m:mo>
      </m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mi>k</m:mi>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
```

```
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mfrac>
      <m:msup>
       <m:mi>&#x03C0;</m:mi>
       <m:mn>2</m:mn>
      </m:msup>
      <m:mn>8</m:mn>
     </m:mfrac>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:munderover>
      <m:mo movablelimits="false">&#x2211;</m:mo>
      <m:mrow>
       <m:mi>k</m:mi>
       <m:mo>=</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mi mathvariant="normal">&#x221E;</m:mi>
     </m:munderover>
     <m:mfrac>
      <m:mn>1</m:mn>
      <m:mi>k</m:mi>
     </m:mfrac>
     <m:mrow>
      <m:msup>
       <m:mi>&#x03C8;</m:mi>
       <m:mo>&#x2032;</m:mo>
      </m:msup>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>+</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi>&#x03B6;</m:mi>
```

```
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mn>3</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mrow>
       <m:msup>
        <m:mi>&#x03C8;</m:mi>
        <m:mrow>
         <m:mi>&#x2032;</m:mi>
         <m:mi>&#x2032;</m:mi>
        </m:mrow>
       </m:msup>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mn>1</m:mn>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>For further sums involving the psi function see
<a href="http://dlmf.nist.gov/Contents/bib/H#hansen:1975:tsp">
   Hansen(1975)
</a>(pp.360367). For sums of gamma functions see
<a href="http://dlmf.nist.gov/Contents/bib/#andrews:1999:sfu">
   Andrews <em>et.al.</em>(1999)
</a>(Chapters 2 and 3).
</p>

<p>For related sums involving finite field analogs of the gamma and
beta functions (Gauss and Jacobi sums) see
<a href="http://dlmf.nist.gov/Contents/bib/#andrews:1999:sfu">
 Andrews <em>et.al.</em>(1999)
</a>(Chapter 1) and
<a href="http://dlmf.nist.gov/Contents/bib/T#terras:1999:fao">
 Terras(1999)
</a>.
</p>
\getchunk{page foot}
```

——

dlmfsoftware.xhtml

— dlmfsoftware.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Software
  </div>
  <hr/>
\getchunk{page foot}
```

——

dlmfspecialvaluesandextrema.xhtml

— dlmfspecialvaluesandextrema.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Special Values and Extrema
  </div>
  <hr/>
<h3>Special Values and Extrema</h3>
<h6>Contents</h6>
 <ul>
  <li>Gamma Function</li>
  <li>Psi Function</li>
  <li>Extrema</li>
 </ul>

<h4>Gamma Function</h4>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
```

```
      <m:mo>(</m:mo>
      <m:mn>1</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mn>1</m:mn>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mi>n</m:mi>
     <m:mi mathvariant="normal">!</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mo>&#x2223;</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>y</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
```

```
     </m:mrow>
     <m:mo>&#x2223;</m:mo>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mfrac>
       <m:mi>&#x03C0;</m:mi>
       <m:mrow>
        <m:mi>y</m:mi>
        <m:mrow>
         <m:mi>sinh</m:mi>
         <m:mrow>
          <m:mo>(</m:mo>
          <m:mrow>
           <m:mi>&#x03C0;</m:mi>
           <m:mi>y</m:mi>
          </m:mrow>
          <m:mo>)</m:mo>
         </m:mrow>
        </m:mrow>
       </m:mrow>
      </m:mfrac>
      <m:mo>)</m:mo>
     </m:mrow>
     <m:mfrac bevelled="true">
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
    </m:msup>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>2</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mo>+</m:mo>
        <m:mrow>
```

```
      <m:mi mathvariant="normal">&#x2148;</m:mi>
      <m:mi>y</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
  <m:mrow>
   <m:mi mathvariant="normal">&#x0393;</m:mi>
   <m:mrow>
    <m:mo>(</m:mo>
    <m:mrow>
     <m:mstyle displaystyle="false">
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
     </m:mstyle>
     <m:mo>-</m:mo>
     <m:mrow>
      <m:mi mathvariant="normal">&#x2148;</m:mi>
      <m:mi>y</m:mi>
     </m:mrow>
    </m:mrow>
    <m:mo>)</m:mo>
   </m:mrow>
  </m:mrow>
 </m:mrow>
 <m:mo>=</m:mo>
 <m:msup>
  <m:mrow>
   <m:mo>&#x2223;</m:mo>
   <m:mrow>
    <m:mi mathvariant="normal">&#x0393;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:mrow>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mi>y</m:mi>
      </m:mrow>
     </m:mrow>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>&#x2223;</m:mo>
  </m:mrow>
```

```
     <m:mn>2</m:mn>
    </m:msup>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mi>&#x03C0;</m:mi>
     <m:mrow>
      <m:mi>cosh</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>&#x03C0;</m:mi>
        <m:mi>y</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>4</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
         <m:mfrac>
```

```
          <m:mn>3</m:mn>
          <m:mn>4</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mo>-</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mfrac>
     <m:mrow>
      <m:mi>&#x03C0;</m:mi>
      <m:msqrt>
       <m:mn>2</m:mn>
      </m:msqrt>
     </m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mi>cosh</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>&#x03C0;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mo>+</m:mo>
      <m:mrow>
       <m:mi mathvariant="normal">&#x2148;</m:mi>
       <m:mrow>
        <m:mi>sinh</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mrow>
          <m:mi>&#x03C0;</m:mi>
          <m:mi>y</m:mi>
         </m:mrow>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mfrac>
   </m:mrow>
  </m:mrow>
</m:math>
```

```
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:msup>
     <m:mi>&#x03C0;</m:mi>
     <m:mfrac bevelled="true">
      <m:mn>1</m:mn>
      <m:mn>2</m:mn>
     </m:mfrac>
    </m:msup>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>1.77245 38509 05516 02729</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>3</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
```

```
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>2.67893 85347 07747 63365</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>2</m:mn>
        <m:mn>3</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>1.35411 79394 26400 41694</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>4</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
```

```
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>3.62560 99082 21908 31193</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>3</m:mn>
        <m:mn>4</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mn>1.22541 67024 65177 64512</m:mn>
     <m:mi mathvariant="normal">&#x2026;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:msup>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mo>&#x2032;</m:mo>
     </m:msup>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mn>1</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
```

```
     <m:mo>-</m:mo>
     <m:mi>&#x03B3;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<h4>Psi Function</h4>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mn>1</m:mn>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mi>&#x03B3;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<a name="equation13"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mstyle displaystyle="false">
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mn>2</m:mn>
       </m:mfrac>
      </m:mstyle>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>&#x03B3;</m:mi>
```

```
    </m:mrow>
    <m:mo>-</m:mo>
    <m:mrow>
     <m:mn>2</m:mn>
     <m:mrow>
      <m:mi>ln</m:mi>
      <m:mn>2</m:mn>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mn>1</m:mn>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:munderover>
       <m:mo movablelimits="false">&#x2211;</m:mo>
       <m:mrow>
        <m:mi>k</m:mi>
        <m:mo>=</m:mo>
        <m:mn>1</m:mn>
       </m:mrow>
       <m:mi>n</m:mi>
      </m:munderover>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mi>k</m:mi>
      </m:mfrac>
     </m:mrow>
     <m:mo>-</m:mo>
     <m:mi>&#x03B3;</m:mi>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
```

```
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mrow>
       <m:mi>n</m:mi>
       <m:mo>+</m:mo>
       <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mstyle>
      </m:mrow>
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>&#x03B3;</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mrow>
        <m:mi>ln</m:mi>
        <m:mn>2</m:mn>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mn>2</m:mn>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mn>3</m:mn>
         </m:mfrac>
        </m:mstyle>
        <m:mo>+</m:mo>
```

```
        <m:mi mathvariant="normal">&#x2026;</m:mi>
        <m:mo>+</m:mo>
        <m:mstyle displaystyle="false">
         <m:mfrac>
          <m:mn>1</m:mn>
          <m:mrow>
           <m:mrow>
            <m:mn>2</m:mn>
            <m:mi>n</m:mi>
           </m:mrow>
           <m:mo>-</m:mo>
           <m:mn>1</m:mn>
          </m:mrow>
         </m:mfrac>
        </m:mstyle>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
 <m:math display="inline">
  <m:mrow>
   <m:mi>n</m:mi>
   <m:mo>&#x2265;</m:mo>
   <m:mn>1</m:mn>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal"></m:mi>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi mathvariant="normal">&#x2148;</m:mi>
        <m:mi>y</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mn>1</m:mn>
```

```
      <m:mrow>
       <m:mn>2</m:mn>
       <m:mi>y</m:mi>
      </m:mrow>
     </m:mfrac>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mi>&#x03C0;</m:mi>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mrow>
       <m:mi>coth</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>&#x03C0;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal"></m:mi>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mstyle displaystyle="false">
        <m:mfrac>
         <m:mn>1</m:mn>
         <m:mn>2</m:mn>
        </m:mfrac>
       </m:mstyle>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
```

```
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mfrac>
      <m:mi>&#x03C0;</m:mi>
      <m:mn>2</m:mn>
     </m:mfrac>
     <m:mrow>
      <m:mi>tanh</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mi>&#x03C0;</m:mi>
        <m:mi>y</m:mi>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi mathvariant="normal"></m:mi>
     <m:mrow>
      <m:mi>&#x03C8;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:mrow>
        <m:mn>1</m:mn>
        <m:mo>+</m:mo>
        <m:mrow>
         <m:mi mathvariant="normal">&#x2148;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
       </m:mrow>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mrow>
```

```
        <m:mn>2</m:mn>
        <m:mi>y</m:mi>
       </m:mrow>
      </m:mfrac>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mi>&#x03C0;</m:mi>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mrow>
       <m:mi>coth</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mi>&#x03C0;</m:mi>
         <m:mi>y</m:mi>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>

<p>
 <m:math display="inline">
  <m:mrow>
   <m:mn>0</m:mn>
   <m:mo>&lt;</m:mo>
   <m:mi>p</m:mi>
   <m:mo>&lt;</m:mo>
   <m:mi>q</m:mi>
  </m:mrow>
 </m:math> are integers, then
</p>

<a name="equation19"/>
<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:mrow>
     <m:mi>&#x03C8;</m:mi>
     <m:mrow>
      <m:mo>(</m:mo>
      <m:mfrac>
       <m:mi>p</m:mi>
       <m:mi>q</m:mi>
      </m:mfrac>
```

```
      <m:mo>)</m:mo>
     </m:mrow>
    </m:mrow>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mi>&#x03B3;</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mi>ln</m:mi>
       <m:mi>q</m:mi>
      </m:mrow>
      <m:mo>-</m:mo>
      <m:mrow>
       <m:mfrac>
        <m:mi>&#x03C0;</m:mi>
        <m:mn>2</m:mn>
       </m:mfrac>
       <m:mrow>
        <m:mi>cot</m:mi>
        <m:mrow>
         <m:mo>(</m:mo>
         <m:mfrac>
          <m:mrow>
           <m:mi>&#x03C0;</m:mi>
           <m:mi>p</m:mi>
          </m:mrow>
          <m:mi>q</m:mi>
         </m:mfrac>
         <m:mo>)</m:mo>
        </m:mrow>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mn>2</m:mn>
      </m:mfrac>
      <m:mrow>
       <m:munderover>
        <m:mo movablelimits="false">&#x2211;</m:mo>
        <m:mrow>
         <m:mi>k</m:mi>
         <m:mo>=</m:mo>
         <m:mn>1</m:mn>
        </m:mrow>
        <m:mrow>
         <m:mi>q</m:mi>
         <m:mo>-</m:mo>
```

```
        <m:mn>1</m:mn>
       </m:mrow>
      </m:munderover>
      <m:mrow>
       <m:mi>cos</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mfrac>
         <m:mrow>
          <m:mn>2</m:mn>
          <m:mi>&#x03C0;</m:mi>
          <m:mi>k</m:mi>
          <m:mi>p</m:mi>
         </m:mrow>
         <m:mi>q</m:mi>
        </m:mfrac>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
      <m:mrow>
       <m:mi>ln</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mrow>
         <m:mn>2</m:mn>
         <m:mo>-</m:mo>
         <m:mrow>
          <m:mn>2</m:mn>
          <m:mrow>
           <m:mi>cos</m:mi>
           <m:mrow>
            <m:mo>(</m:mo>
            <m:mfrac>
             <m:mrow>
              <m:mn>2</m:mn>
              <m:mi>&#x03C0;</m:mi>
              <m:mi>k</m:mi>
             </m:mrow>
             <m:mi>q</m:mi>
            </m:mfrac>
            <m:mo>)</m:mo>
           </m:mrow>
          </m:mrow>
         </m:mrow>
        </m:mrow>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:mrow>
</m:math>
```

```
</div>

<h4>Extrema</h4>
<div>
 <m:math display="inline">
  <m:mrow>
   <m:mrow>
    <m:msup>
     <m:mi mathvariant="normal">&#x0393;</m:mi>
     <m:mo>&#x2032;</m:mo>
    </m:msup>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:msub>
      <m:mi>x</m:mi>
      <m:mi>n</m:mi>
     </m:msub>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mrow>
    <m:mi>&#x03C8;</m:mi>
    <m:mrow>
     <m:mo>(</m:mo>
     <m:msub>
      <m:mi>x</m:mi>
      <m:mi>n</m:mi>
     </m:msub>
     <m:mo>)</m:mo>
    </m:mrow>
   </m:mrow>
   <m:mo>=</m:mo>
   <m:mn>0</m:mn>
  </m:mrow>
 </m:math>.
</div>
<br/>
<div class="center">
 <table align="center">
  <thead>
   <tr>
    <th align="center" class="b l r t">
     <m:math display="inline">
      <m:mi>n</m:mi>
     </m:math>
    </th>
    <th align="center" class="b r t">
     <m:math display="inline">
      <m:msub>
       <m:mi>x</m:mi>
       <m:mi>n</m:mi>
      </m:msub>
     </m:math>
```

```
   </th>
   <th align="center" class="b r t">
    <m:math display="inline">
     <m:mrow>
      <m:mi mathvariant="normal">&#x0393;</m:mi>
      <m:mrow>
       <m:mo>(</m:mo>
       <m:msub>
        <m:mi>x</m:mi>
        <m:mi>n</m:mi>
       </m:msub>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:math>
   </th>
  </tr>
 </thead>
 <tbody>
  <tr>
   <th align="right" class="l r">0
   </th>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mn>1.46163 21449</m:mn>
    </m:math>
   </td>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mn>0.88560 31944</m:mn>
    </m:math>
   </td>
  </tr>
  <tr>
   <th align="right" class="l r">1
   </th>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>0.50408 30083</m:mn>
     </m:mrow>
    </m:math>
   </td>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>3.54464 36112</m:mn>
     </m:mrow>
    </m:math>
   </td>
  </tr>
  <tr>
```

```
   <th align="right" class="l r">2
   </th>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>1.57349 84732</m:mn>
     </m:mrow>
    </m:math>
   </td>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mn>2.30240 72583</m:mn>
    </m:math>
   </td>
  </tr>
  <tr>
   <th align="right" class="B l r">3
   </th>
   <td align="right" class="B r">
    <m:math display="inline">
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>2.61072 08875</m:mn>
     </m:mrow>
    </m:math>
   </td>
   <td align="right" class="B r">
    <m:math display="inline">
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>0.88813 63584</m:mn>
     </m:mrow>
    </m:math>
   </td>
  </tr>
  <tr>
   <th align="right" class="l r">4
   </th>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mn>3.63529 33665</m:mn>
     </m:mrow>
    </m:math>
   </td>
   <td align="right" class="r">
    <m:math display="inline">
     <m:mn>0.24512 75398</m:mn>
    </m:math>
   </td>
  </tr>
  <tr>
```

```
  <th align="right" class="l r">5
  </th>
  <td align="right" class="r">
   <m:math display="inline">
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>4.65323 77626</m:mn>
    </m:mrow>
   </m:math>
  </td>
  <td align="right" class="r">
   <m:math display="inline">
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>0.05277 96396</m:mn>
    </m:mrow>
   </m:math>
  </td>
 </tr>
 <tr>
  <th align="right" class="B l r">6
  </th>
  <td align="right" class="B r">
   <m:math display="inline">
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>5.66716 24513</m:mn>
    </m:mrow>
   </m:math>
  </td>
  <td align="right" class="B r">
   <m:math display="inline">
    <m:mn>0.00932 45945</m:mn>
   </m:math>
  </td>
 </tr>
 <tr>
  <th align="right" class="l r">7
  </th>
  <td align="right" class="r">
   <m:math display="inline">
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>6.67841 82649</m:mn>
    </m:mrow>
   </m:math>
  </td>
  <td align="right" class="r">
   <m:math display="inline">
    <m:mrow>
     <m:mo>-</m:mo>
     <m:mn>0.00139 73966</m:mn>
    </m:mrow>
   </m:math>
```

```
    </td>
   </tr>
   <tr>
    <th align="right" class="l r">8
    </th>
    <td align="right" class="r">
     <m:math display="inline">
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mn>7.68778 83250</m:mn>
      </m:mrow>
     </m:math>
    </td>
    <td align="right" class="r">
     <m:math display="inline">
      <m:mn>0.00018 18784</m:mn>
     </m:math>
    </td>
   </tr>
   <tr>
    <th align="right" class="l r">9
    </th>
    <td align="right" class="r">
     <m:math display="inline">
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mn>8.69576 41633</m:mn>
      </m:mrow>
     </m:math>
    </td>
    <td align="right" class="r">
     <m:math display="inline">
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mn>0.00002 09253</m:mn>
      </m:mrow>
     </m:math>
    </td>
   </tr>
   <tr>
    <th align="right" class="b l r">10
    </th>
    <td align="right" class="b r">
     <m:math display="inline">
      <m:mrow>
       <m:mo>-</m:mo>
       <m:mn>9.70267 25406</m:mn>
      </m:mrow>
     </m:math>
    </td>
    <td align="right" class="b r">
     <m:math display="inline">
      <m:mn>0.00000 21574</m:mn>
     </m:math>
```

```
    </td>
   </tr>
  </tbody>
 </table>
</div>

<p>As
 <m:math display="inline">
  <m:mrow>
   <m:mi>n</m:mi>
   <m:mi mathvariant="normal">&#x221E;</m:mi>
  </m:mrow>
 </m:math>,
</p>

<div align="center">
 <m:math display="block">
  <m:mrow>
   <m:mrow>
    <m:msub>
     <m:mi>x</m:mi>
     <m:mi>n</m:mi>
    </m:msub>
    <m:mo>=</m:mo>
    <m:mrow>
     <m:mrow>
      <m:mo>-</m:mo>
      <m:mi>n</m:mi>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mfrac>
       <m:mn>1</m:mn>
       <m:mi>&#x03C0;</m:mi>
      </m:mfrac>
      <m:mrow>
       <m:mi>arctan</m:mi>
       <m:mrow>
        <m:mo>(</m:mo>
        <m:mfrac>
         <m:mi>&#x03C0;</m:mi>
         <m:mrow>
          <m:mi>ln</m:mi>
          <m:mi>n</m:mi>
         </m:mrow>
        </m:mfrac>
        <m:mo>)</m:mo>
       </m:mrow>
      </m:mrow>
     </m:mrow>
     <m:mo>+</m:mo>
     <m:mrow>
      <m:mi>O</m:mi>
      <m:mrow>
```

```
       <m:mo>(</m:mo>
       <m:mfrac>
        <m:mn>1</m:mn>
        <m:mrow>
         <m:mi>n</m:mi>
         <m:msup>
          <m:mrow>
           <m:mo>(</m:mo>
           <m:mrow>
            <m:mi>ln</m:mi>
            <m:mi>n</m:mi>
           </m:mrow>
           <m:mo>)</m:mo>
          </m:mrow>
          <m:mn>2</m:mn>
         </m:msup>
        </m:mrow>
       </m:mfrac>
       <m:mo>)</m:mo>
      </m:mrow>
     </m:mrow>
    </m:mrow>
   </m:mrow>
  </m:mrow>
 </m:math>
</div>
\getchunk{page foot}
```

dlmftables.xhtml

— dlmftables.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">
   <a href="http://dlmf.nist.gov">
    Digital Library of Mathematical Functions
   </a><br/>
   The Gamma Function -- Tables
  </div>
  <hr/>
<h3>Tables</h3>

These tables show Axiom's compliance with published standard values.
In all cases shown here Axiom conforms to the accuracy of the published
tables.

<ul>
 <li>The Gamma Function</li>
```

```
 <li>The Psi Function</li>
</ul>

<h4>The Gamma Function</h4>

This table was constructed from the published values in the
Handbook of Mathematical Functions, by Milton Abramowitz
and Irene A. Stegun, by Dover (1965), pp 267-270.

The first column is the point where the Gamma function is evaluated.
The second column is the value reported in the Handbook.
The third column is the actual value computed by Axiom at the given point.
The fourth column is the difference of Axiom's value and the Handbook value.

<table border="1">
 <tr>
  <th>point</th>
  <th>Handbook Value</th>
  <th>Axiom Computed Value</th>
  <th>Difference</th>
 </tr>
 <tr>
  <td>1.000</td>
  <td>1.0000000000</td>
  <td>1.</td>
  <td align="right">0.</td>
 </tr>
 <tr>
  <td>1.005</td>
  <td>0.9971385354</td>
  <td>0.9971385352483757</td>
  <td align="right">-1.51E-10</td>
 </tr>
 <tr>
  <td>1.010</td>
  <td>0.9943258512</td>
  <td>0.99432585118631189</td>
  <td align="right">-2.03E-11</td>
 </tr>
 <tr>
  <td>1.015</td>
  <td>0.9915612888</td>
  <td>0.99156128884131323</td>
  <td align="right">4.14E-11</td>
 </tr>
 <tr>
  <td>1.020</td>
  <td>0.9888442033</td>
  <td>0.9888442032538789</td>
  <td align="right">-4.31E-11</td>
 </tr>
 <tr>
  <td>1.025</td>
  <td>0.9861739633</td>
```

```
 <td>0.98617396313592742</td>
 <td align="right">-1.54E-10</td>
</tr>
<tr>
 <td>1.030</td>
 <td>0.9835499506</td>
 <td>0.98354995053928918</td>
 <td align="right">-7.59E-11</td>
</tr>
<tr>
 <td>1.035</td>
 <td>0.9809715606</td>
 <td>0.98097156056367696</td>
 <td align="right">-4.60E-11</td>
</tr>
<tr>
 <td>1.040</td>
 <td>0.9784382009</td>
 <td>0.9784382009247683</td>
 <td align="right"> 3.00E-11</td>
</tr>
<tr>
 <td>1.045</td>
 <td>0.9759492919</td>
 <td>0.97594929183099266</td>
 <td align="right">-6.55E-11</td>
</tr>
<tr>
 <td>1.050</td>
 <td>0.9735042656</td>
 <td>0.97350426556841785</td>
 <td align="right">-2.72E-11</td>
</tr>
<tr>
 <td>1.055</td>
 <td>0.9711025663</td>
 <td>0.97110256624499502</td>
 <td align="right">-6.77E-11</td>
</tr>
<tr>
 <td>1.060</td>
 <td>0.9687436495</td>
 <td>0.96874364951272707</td>
 <td align="right">-2.36E-12</td>
</tr>
<tr>
 <td>1.065</td>
 <td>0.9664269823</td>
 <td>0.96642698229777113</td>
 <td align="right">-1.37E-11</td>
</tr>
<tr>
 <td>1.070</td>
 <td>0.9641520425</td>
```

```
 <td>0.96415204253821729</td>
 <td align="right"> 4.61E-11</td>
</tr>
<tr>
 <td>1.075</td>
 <td>0.9619183189</td>
 <td>0.96191831892929192</td>
 <td align="right"> 2.31E-11</td>
</tr>
<tr>
 <td>1.080</td>
 <td>0.9597253107</td>
 <td>0.95972531067573963</td>
 <td align="right">-3.00E-11</td>
</tr>
<tr>
 <td>1.085</td>
 <td>0.9575725273</td>
 <td>0.95757252725116249</td>
 <td align="right">-3.68E-11</td>
</tr>
<tr>
 <td>1.090</td>
 <td>0.9554594882</td>
 <td>0.95545948816407866</td>
 <td align="right">-4.24E-11</td>
</tr>
<tr>
 <td>1.095</td>
 <td>0.9533857227</td>
 <td>0.95338572273049704</td>
 <td align="right"> 2.34E-11</td>
</tr>
<tr>
 <td>1.100</td>
 <td>0.9513507699</td>
 <td>0.95135076987625944</td>
 <td align="right">-2.49E-11</td>
</tr>
<tr>
 <td>1.105</td>
 <td>0.9493541778</td>
 <td>0.94935417782771081</td>
 <td align="right"> 2.11E-11</td>
</tr>
<tr>
 <td>1.110</td>
 <td>0.9473955040</td>
 <td>0.94739550404472173</td>
 <td align="right"> 5.80E-11</td>
</tr>
<tr>
 <td>1.115</td>
 <td>0.9454743149</td>
```

```
 <td>0.94547431492209555</td>
 <td align="right"> 1.12E-11</td>
</tr>
<tr>
 <td>1.120</td>
 <td>0.9435901856</td>
 <td>0.94359018561564112</td>
 <td align="right"> 1.06E-11</td>
</tr>
<tr>
 <td>1.125</td>
 <td>0.9417426997</td>
 <td>0.94174269984970138</td>
 <td align="right"> 1.39E-10</td>
</tr>
<tr>
 <td>1.130</td>
 <td>0.9399314497</td>
 <td>0.93993144972988807</td>
 <td align="right"> 1.67E-11</td>
</tr>
<tr>
 <td>1.135</td>
 <td>0.9381560356</td>
 <td>0.93815603556085947</td>
 <td align="right">-5.14E-11</td>
</tr>
<tr>
 <td>1.140</td>
 <td>0.9364160657</td>
 <td>0.93641606566898694</td>
 <td align="right">-2.97E-11</td>
</tr>
<tr>
 <td>1.145</td>
 <td>0.9347111562</td>
 <td>0.93471115622975964</td>
 <td align="right"> 2.05E-11</td>
</tr>
<tr>
 <td>1.150</td>
 <td>0.9330409311</td>
 <td>0.93304093109978414</td>
 <td align="right"> 6.51E-12</td>
</tr>
<tr>
 <td>1.155</td>
 <td>0.9314050217</td>
 <td>0.93140502165323868</td>
 <td align="right">-3.93E-11</td>
</tr>
<tr>
 <td>1.160</td>
 <td>0.9298030666</td>
```

```
 <td>0.92980306664109957</td>
 <td align="right"> 4.51E-11</td>
</tr>
<tr>
 <td>1.165</td>
 <td>0.9282347120</td>
 <td>0.92823471196190366</td>
 <td align="right">-2.59E-11</td>
</tr>
<tr>
 <td>1.170</td>
 <td>0.9266996106</td>
 <td>0.92669961062266581</td>
 <td align="right"> 2.10E-11</td>
</tr>
<tr>
 <td>1.175</td>
 <td>0.9251974225</td>
 <td>0.92519742251686099</td>
 <td align="right"> 1.24E-11</td>
</tr>
<tr>
 <td>1.180</td>
 <td>0.9237278143</td>
 <td>0.92372781430006712</td>
 <td align="right">-1.17E-11</td>
</tr>
<tr>
 <td>1.185</td>
 <td>0.9222904591</td>
 <td>0.92229045925047382</td>
 <td align="right"> 1.49E-10</td>
</tr>
<tr>
 <td>1.190</td>
 <td>0.9208850371</td>
 <td>0.92088503713299241</td>
 <td align="right"> 2.60E-11</td>
</tr>
<tr>
 <td>1.195</td>
 <td>0.9195112341</td>
 <td>0.91951123406686597</td>
 <td align="right">-2.98E-11</td>
</tr>
<tr>
 <td>1.200</td>
 <td>0.9181687424</td>
 <td>0.91816874239667101</td>
 <td align="right">-1.67E-11</td>
</tr>
<tr>
 <td>1.205</td>
 <td>0.9168572606</td>
```

```
 <td>0.91685726056661909</td>
 <td align="right">-3.28E-11</td>
</tr>
<tr>
 <td>1.210</td>
 <td>0.9155764930</td>
 <td>0.91557649299805532</td>
 <td align="right"> 8.85E-12</td>
</tr>
<tr>
 <td>1.215</td>
 <td>0.9143261400</td>
 <td>0.91432614997006778</td>
 <td align="right"> 9.98E-9</td>
</tr>
<tr>
 <td>1.220</td>
 <td>0.9131059475</td>
 <td>0.91310594750311536</td>
 <td align="right"> 1.37E-11</td>
</tr>
<tr>
 <td>1.225</td>
 <td>0.9119156071</td>
 <td>0.91191560725927312</td>
 <td align="right"> 1.49E-10</td>
</tr>
<tr>
 <td>1.230</td>
 <td>0.9107548564</td>
 <td>0.91075485637655895</td>
 <td align="right">-1.50E-11</td>
</tr>
<tr>
 <td>1.235</td>
 <td>0.9096234274</td>
 <td>0.90962342744425173</td>
 <td align="right"> 4.03E-11</td>
</tr>
<tr>
 <td>1.240</td>
 <td>0.9085210583</td>
 <td>0.90852105834198582</td>
 <td align="right"> 4.21E-11</td>
</tr>
<tr>
 <td>1.245</td>
 <td>0.9074474922</td>
 <td>0.90744749215126341</td>
 <td align="right">-5.77E-11</td>
</tr>
<tr>
 <td>1.250</td>
 <td>0.9064024771</td>
```

```
  <td>0.90640247705547716</td>
  <td align="right">-3.68E-11</td>
 </tr>
 <tr>
  <td>1.255</td>
  <td>0.9053857663</td>
  <td>0.90538576624240463</td>
  <td align="right">-5.23E-11</td>
 </tr>
 <tr>
  <td>1.260</td>
  <td>0.9043971178</td>
  <td>0.90439711780910215</td>
  <td align="right"> 2.01E-11</td>
 </tr>
 <tr>
  <td>1.265</td>
  <td>0.9034362946</td>
  <td>0.90343629466913566</td>
  <td align="right"> 5.78E-11</td>
 </tr>
 <tr>
  <td>1.270</td>
  <td>0.9025030645</td>
  <td>0.90250306446208062</td>
  <td align="right">-5.13E-11</td>
 </tr>
 <tr>
  <td>1.275</td>
  <td>0.9015971994</td>
  <td>0.90159719946523187</td>
  <td align="right"> 5.66E-11</td>
 </tr>
 <tr>
  <td>1.280</td>
  <td>0.9007184765</td>
  <td>0.90071847650745973</td>
  <td align="right"> 5.78E-13</td>
 </tr>
 <tr>
  <td>1.285</td>
  <td>0.8998666769</td>
  <td>0.89986667689491762</td>
  <td align="right"> 5.55E-12</td>
 </tr>
 <tr>
  <td>1.290</td>
  <td>0.8990415863</td>
  <td>0.89904158628967101</td>
  <td align="right">-3.93E-12</td>
 </tr>
 <tr>
  <td>1.295</td>
  <td>0.8982429947</td>
```

```
 <td>0.89824299468914737</td>
 <td align="right">-1.72E-11</td>
</tr>
<tr>
 <td>1.300</td>
 <td>0.8974706963</td>
 <td>0.89747069630804477</td>
 <td align="right"> 2.65E-12</td>
</tr>
<tr>
 <td>1.305</td>
 <td>0.8967244895</td>
 <td>0.89672448951215833</td>
 <td align="right"> 2.37E-11</td>
</tr>
<tr>
 <td>1.310</td>
 <td>0.8960041767</td>
 <td>0.89600417674396082</td>
 <td align="right"> 4.53E-11</td>
</tr>
<tr>
 <td>1.315</td>
 <td>0.8953095644</td>
 <td>0.89530956444995535</td>
 <td align="right"> 5.43E-11</td>
</tr>
<tr>
 <td>1.320</td>
 <td>0.8946404630</td>
 <td>0.89464046300975775</td>
 <td align="right"> 1.28E-11</td>
</tr>
<tr>
 <td>1.325</td>
 <td>0.8939966866</td>
 <td>0.89399668666686083</td>
 <td align="right"> 7.95E-11</td>
</tr>
<tr>
 <td>1.330</td>
 <td>0.8933780535</td>
 <td>0.89337805346103716</td>
 <td align="right">-3.97E-11</td>
</tr>
<tr>
 <td>1.335</td>
 <td>0.8927843850</td>
 <td>0.89278438516233538</td>
 <td align="right"> 1.51E-10</td>
</tr>
<tr>
 <td>1.340</td>
 <td>0.8922155072</td>
```

```
 <td>0.89221550720663356</td>
 <td align="right"> 1.43E-11</td>
</tr>
<tr>
 <td>1.345</td>
 <td>0.8916712485</td>
 <td>0.89167124863270442</td>
 <td align="right"> 1.24E-10</td>
</tr>
<tr>
 <td>1.350</td>
 <td>0.8911514420</td>
 <td>0.89115144202666452</td>
 <td align="right"> 3.78E-11</td>
</tr>
<tr>
 <td>1.355</td>
 <td>0.8906559235</td>
 <td>0.89065592343803057</td>
 <td align="right">-5.12E-11</td>
</tr>
<tr>
 <td>1.360</td>
 <td>0.8901845324</td>
 <td>0.8901845323574008</td>
 <td align="right">-5.70E-11</td>
</tr>
<tr>
 <td>1.365</td>
 <td>0.8897371116</td>
 <td>0.88973711163470881</td>
 <td align="right"> 3.11E-11</td>
</tr>
<tr>
 <td>1.370</td>
 <td>0.8893135074</td>
 <td>0.88931350742948501</td>
 <td align="right"> 4.09E-11</td>
</tr>
<tr>
 <td>1.375</td>
 <td>0.8889135692</td>
 <td>0.88891356915622532</td>
 <td align="right">-5.89E-11</td>
</tr>
<tr>
 <td>1.380</td>
 <td>0.8885371494</td>
 <td>0.88853714943101736</td>
 <td align="right"> 2.03E-11</td>
</tr>
<tr>
 <td>1.385</td>
 <td>0.8881841041</td>
```

```
 <td>0.88818410401940351</td>
 <td align="right">-9.53E-11</td>
</tr>
<tr>
 <td>1.390</td>
 <td>0.8878542918</td>
 <td>0.88785429178544073</td>
 <td align="right">-1.00E-11</td>
</tr>
<tr>
 <td>1.395</td>
 <td>0.8875475748</td>
 <td>0.88754757464193323</td>
 <td align="right">-1.49E-10</td>
</tr>
<tr>
 <td>1.400</td>
 <td>0.8872638175</td>
 <td>0.88726381750180738</td>
 <td align="right">-7.13E-12</td>
</tr>
<tr>
 <td>1.405</td>
 <td>0.8870028884</td>
 <td>0.88700288823059736</td>
 <td align="right">-1.66E-10</td>
</tr>
<tr>
 <td>1.410</td>
 <td>0.8867646576</td>
 <td>0.88676465760002188</td>
 <td align="right"> 3.66E-12</td>
</tr>
<tr>
 <td>1.415</td>
 <td>0.8865489993</td>
 <td>0.88654899924499497</td>
 <td align="right">-4.45E-11</td>
</tr>
<tr>
 <td>1.420</td>
 <td>0.8863557896</td>
 <td>0.88635578960951567</td>
 <td align="right">-1.60E-12</td>
</tr>
<tr>
 <td>1.425</td>
 <td>0.8861849081</td>
 <td>0.88618490791840432</td>
 <td align="right">-1.81E-10</td>
</tr>
<tr>
 <td>1.430</td>
 <td>0.8860362361</td>
```

```
 <td>0.88603623612466142</td>
 <td align="right"> 2.35E-11</td>
</tr>
<tr>
 <td>1.435</td>
 <td>0.8859096587</td>
 <td>0.88590965887072826</td>
 <td align="right"> 1.59E-10</td>
</tr>
<tr>
 <td>1.440</td>
 <td>0.8858050635</td>
 <td>0.88580506344804788</td>
 <td align="right">-5.45E-11</td>
</tr>
<tr>
 <td>1.445</td>
 <td>0.8857223397</td>
 <td>0.88572233975753722</td>
 <td align="right"> 5.12E-11</td>
</tr>
<tr>
 <td>1.450</td>
 <td>0.8856613803</td>
 <td>0.88566138027095553</td>
 <td align="right">-3.63E-11</td>
</tr>
<tr>
 <td>1.455</td>
 <td>0.8856220700</td>
 <td>0.88562207999314335</td>
 <td align="right"> 9.99E-9</td>
</tr>
<tr>
 <td>1.460</td>
 <td>0.8856043364</td>
 <td>0.88560433642511449</td>
 <td align="right"> 3.29E-11</td>
</tr>
<tr>
 <td>1.465</td>
 <td>0.8856080495</td>
 <td>0.88560804952797856</td>
 <td align="right"> 4.00E-11</td>
</tr>
<tr>
 <td>1.470</td>
 <td>0.8856331217</td>
 <td>0.88563312168767672</td>
 <td align="right">-2.25E-11</td>
</tr>
<tr>
 <td>1.475</td>
 <td>0.8856794575</td>
```

```
 <td>0.88567945767984679</td>
 <td align="right"> 1.68E-10</td>
</tr>
<tr>
 <td>1.480</td>
 <td>0.8857469646</td>
 <td>0.88574696463853297</td>
 <td align="right"> 3.58E-11</td>
</tr>
<tr>
 <td>1.485</td>
 <td>0.8858355520</td>
 <td>0.88583555202000774</td>
 <td align="right"> 1.39E-11</td>
</tr>
<tr>
 <td>1.490</td>
 <td>0.8859451316</td>
 <td>0.885945131572484</td>
 <td align="right">-2.22E-11</td>
</tr>
<tr>
 <td>1.495</td>
 <td>0.8860756174</td>
 <td>0.88607561730422169</td>
 <td align="right">-9.20E-11</td>
</tr>
<tr>
 <td>1.500</td>
 <td>0.8862269255</td>
 <td>0.88622692545275816</td>
 <td align="right">-5.14E-11</td>
</tr>
<tr>
 <td>1.505</td>
 <td>0.8863989744</td>
 <td>0.88639897445482596</td>
 <td align="right"> 5.62E-11</td>
</tr>
<tr>
 <td>1.510</td>
 <td>0.8865916850</td>
 <td>0.88659168491694862</td>
 <td align="right">-8.75E-11</td>
</tr>
<tr>
 <td>1.515</td>
 <td>0.8868049797</td>
 <td>0.88680497958669369</td>
 <td align="right">-1.15E-10</td>
</tr>
<tr>
 <td>1.520</td>
 <td>0.8870387833</td>
```

```
 <td>0.88703878332457031</td>
 <td align="right"> 3.78E-11</td>
</tr>
<tr>
 <td>1.525</td>
 <td>0.8872930231</td>
 <td>0.88729302307655866</td>
 <td align="right">-3.89E-11</td>
</tr>
<tr>
 <td>1.530</td>
 <td>0.8875676278</td>
 <td>0.88756762784725507</td>
 <td align="right"> 5.05E-11</td>
</tr>
<tr>
 <td>1.535</td>
 <td>0.8878625287</td>
 <td>0.88786252867361892</td>
 <td align="right">-2.97E-11</td>
</tr>
<tr>
 <td>1.540</td>
 <td>0.8881776586</td>
 <td>0.88817765859552456</td>
 <td align="right">-1.03E-11</td>
</tr>
<tr>
 <td>1.545</td>
 <td>0.8885129527</td>
 <td>0.88851295264558472</td>
 <td align="right">-4.41E-11</td>
</tr>
<tr>
 <td>1.550</td>
 <td>0.8888683478</td>
 <td>0.88886834780261559</td>
 <td align="right"> 2.74E-12</td>
</tr>
<tr>
 <td>1.555</td>
 <td>0.8892437830</td>
 <td>0.88924378298210571</td>
 <td align="right">-1.06E-11</td>
</tr>
<tr>
 <td>1.560</td>
 <td>0.8896391990</td>
 <td>0.88963919900923583</td>
 <td align="right">-3.65E-12</td>
</tr>
<tr>
 <td>1.565</td>
 <td>0.8900545387</td>
```

```
 <td>0.89005453859597561</td>
 <td align="right">-1.04E-10</td>
</tr>
<tr>
 <td>1.570</td>
 <td>0.8904897463</td>
 <td>0.89048974631869759</td>
 <td align="right"> 2.61E-11</td>
</tr>
<tr>
 <td>1.575</td>
 <td>0.8909447686</td>
 <td>0.89094476859629979</td>
 <td align="right"> 8.93E-12</td>
</tr>
<tr>
 <td>1.580</td>
 <td>0.8914195537</td>
 <td>0.89141955366882042</td>
 <td align="right">-2.38E-11</td>
</tr>
<tr>
 <td>1.585</td>
 <td>0.8919140515</td>
 <td>0.8919140515765388</td>
 <td align="right"> 8.47E-11</td>
</tr>
<tr>
 <td>1.590</td>
 <td>0.8924282141</td>
 <td>0.8924282141395512</td>
 <td align="right"> 3.07E-11</td>
</tr>
<tr>
 <td>1.595</td>
 <td>0.8929619949</td>
 <td>0.89296199493781103</td>
 <td align="right"> 4.74E-11</td>
</tr>
<tr>
 <td>1.600</td>
 <td>0.8935153493</td>
 <td>0.89351534928506793</td>
 <td align="right">-2.24E-11</td>
</tr>
<tr>
 <td>1.605</td>
 <td>0.8940882342</td>
 <td>0.89408823423580575</td>
 <td align="right"> 3.63E-11</td>
</tr>
<tr>
 <td>1.610</td>
 <td>0.8946806085</td>
```

```
 <td>0.89468060852796683</td>
 <td align="right"> 2.74E-11</td>
</tr>
<tr>
 <td>1.615</td>
 <td>0.8952924327</td>
 <td>0.89529243259029823</td>
 <td align="right">-9.74E-11</td>
</tr>
<tr>
 <td>1.620</td>
 <td>0.8959236685</td>
 <td>0.89592366851824745</td>
 <td align="right"> 2.86E-11</td>
</tr>
<tr>
 <td>1.625</td>
 <td>0.8965742800</td>
 <td>0.89657428005659789</td>
 <td align="right"> 6.46E-11</td>
</tr>
<tr>
 <td>1.630</td>
 <td>0.8972442326</td>
 <td>0.89724423258250552</td>
 <td align="right">-7.80E-12</td>
</tr>
<tr>
 <td>1.635</td>
 <td>0.8979334930</td>
 <td>0.89793349308892934</td>
 <td align="right"> 9.89E-11</td>
</tr>
<tr>
 <td>1.640</td>
 <td>0.8986420302</td>
 <td>0.89864203016845012</td>
 <td align="right">-2.68E-11</td>
</tr>
<tr>
 <td>1.645</td>
 <td>0.8993698138</td>
 <td>0.89936981399746452</td>
 <td align="right"> 2.04E-10</td>
</tr>
<tr>
 <td>1.650</td>
 <td>0.9001168163</td>
 <td>0.9001168163207548</td>
 <td align="right"> 1.21E-11</td>
</tr>
<tr>
 <td>1.655</td>
 <td>0.9008830104</td>
```

```
 <td>0.90088301043641827</td>
 <td align="right"> 2.24E-11</td>
</tr>
<tr>
 <td>1.660</td>
 <td>0.9016683712</td>
 <td>0.90166837118115595</td>
 <td align="right">-1.49E-11</td>
</tr>
<tr>
 <td>1.665</td>
 <td>0.9024728748</td>
 <td>0.90247287490643413</td>
 <td align="right"> 1.16E-10</td>
</tr>
<tr>
 <td>1.670</td>
 <td>0.9032964995</td>
 <td>0.9032964995021503</td>
 <td align="right">-1.09E-11</td>
</tr>
<tr>
 <td>1.675</td>
 <td>0.9041392243</td>
 <td>0.90413922432675797</td>
 <td align="right"> 3.24E-11</td>
</tr>
<tr>
 <td>1.680</td>
 <td>0.9050010302</td>
 <td>0.90500103023115419</td>
 <td align="right"> 4.40E-11</td>
</tr>
<tr>
 <td>1.685</td>
 <td>0.9058818996</td>
 <td>0.90588189953639731</td>
 <td align="right">-7.63E-11</td>
</tr>
<tr>
 <td>1.690</td>
 <td>0.9067818160</td>
 <td>0.90678181602099839</td>
 <td align="right"> 9.93E-12</td>
</tr>
<tr>
 <td>1.695</td>
 <td>0.9077007650</td>
 <td>0.90770076490852225</td>
 <td align="right">-9.63E-11</td>
</tr>
<tr>
 <td>1.700</td>
 <td>0.9086387329</td>
```

```
  <td>0.90863873285549646</td>
  <td align="right">-5.97E-11</td>
 </tr>
 <tr>
  <td>1.705</td>
  <td>0.9095957079</td>
  <td>0.90959570793962097</td>
  <td align="right"> 4.25E-11</td>
 </tr>
 <tr>
  <td>1.710</td>
  <td>0.9105716796</td>
  <td>0.9105716796482709</td>
  <td align="right"> 5.89E-11</td>
 </tr>
 <tr>
  <td>1.715</td>
  <td>0.9115666390</td>
  <td>0.91156663886729161</td>
  <td align="right">-1.31E-10</td>
 </tr>
 <tr>
  <td>1.720</td>
  <td>0.9125805779</td>
  <td>0.91258057787007674</td>
  <td align="right">-1.93E-11</td>
 </tr>
 <tr>
  <td>1.725</td>
  <td>0.9136134904</td>
  <td>0.91361349029479011</td>
  <td align="right">-1.16E-10</td>
 </tr>
 <tr>
  <td>1.730</td>
  <td>0.9146653712</td>
  <td>0.91466537118231861</td>
  <td align="right">-2.63E-11</td>
 </tr>
 <tr>
  <td>1.735</td>
  <td>0.9157362171</td>
  <td>0.9157362168940244</td>
  <td align="right">-2.15E-10</td>
 </tr>
 <tr>
  <td>1.740</td>
  <td>0.9168260252</td>
  <td>0.91682602514979106</td>
  <td align="right">-5.47E-11</td>
 </tr>
 <tr>
  <td>1.745</td>
  <td>0.9179347950</td>
```

```
 <td>0.91793479500653363</td>
 <td align="right"> 8.97E-12</td>
</tr>
<tr>
 <td>1.750</td>
 <td>0.9190625268</td>
 <td>0.91906252684888312</td>
 <td align="right"> 3.95E-11</td>
</tr>
<tr>
 <td>1.755</td>
 <td>0.9202092224</td>
 <td>0.92020922238011904</td>
 <td align="right">-3.48E-11</td>
</tr>
<tr>
 <td>1.760</td>
 <td>0.9213748846</td>
 <td>0.92137488461334993</td>
 <td align="right"> 4.68E-12</td>
</tr>
<tr>
 <td>1.765</td>
 <td>0.9225595178</td>
 <td>0.92255951786293755</td>
 <td align="right"> 4.88E-11</td>
</tr>
<tr>
 <td>1.770</td>
 <td>0.9237631277</td>
 <td>0.9237631277361581</td>
 <td align="right"> 2.96E-11</td>
</tr>
<tr>
 <td>1.775</td>
 <td>0.9249857211</td>
 <td>0.92498572112510025</td>
 <td align="right"> 2.89E-11</td>
</tr>
<tr>
 <td>1.780</td>
 <td>0.9262273062</td>
 <td>0.92622730619879157</td>
 <td align="right"> 8.37E-12</td>
</tr>
<tr>
 <td>1.785</td>
 <td>0.9274878926</td>
 <td>0.92748789239555507</td>
 <td align="right">-1.97E-10</td>
</tr>
<tr>
 <td>1.790</td>
 <td>0.9287674904</td>
```

```
 <td>0.92876749040057904</td>
 <td align="right">-3.84E-12</td>
</tr>
<tr>
 <td>1.795</td>
 <td>0.9300661123</td>
 <td>0.93006611219852275</td>
 <td align="right">-1.13E-10</td>
</tr>
<tr>
 <td>1.800</td>
 <td>0.9313837710</td>
 <td>0.93138377097715253</td>
 <td align="right">-2.97E-11</td>
</tr>
<tr>
 <td>1.805</td>
 <td>0.9327204811</td>
 <td>0.93272048117993289</td>
 <td align="right"> 8.20E-11</td>
</tr>
<tr>
 <td>1.810</td>
 <td>0.9340762585</td>
 <td>0.93407625848467779</td>
 <td align="right">-2.05E-11</td>
</tr>
<tr>
 <td>1.815</td>
 <td>0.9354511198</td>
 <td>0.93545111979719375</td>
 <td align="right"> 8.27E-12</td>
</tr>
<tr>
 <td>1.820</td>
 <td>0.9368450832</td>
 <td>0.93684508324512517</td>
 <td align="right"> 4.80E-11</td>
</tr>
<tr>
 <td>1.825</td>
 <td>0.9382581682</td>
 <td>0.93825816817200214</td>
 <td align="right">-2.82E-11</td>
</tr>
<tr>
 <td>1.830</td>
 <td>0.9396903951</td>
 <td>0.93969039513148056</td>
 <td align="right"> 1.86E-11</td>
</tr>
<tr>
 <td>1.835</td>
 <td>0.9411417859</td>
```

```
 <td>0.94114178588178177</td>
 <td align="right">-2.64E-11</td>
</tr>
<tr>
 <td>1.840</td>
 <td>0.9426123634</td>
 <td>0.94261236338031951</td>
 <td align="right">-2.35E-11</td>
</tr>
<tr>
 <td>1.845</td>
 <td>0.9441021519</td>
 <td>0.94410215177851575</td>
 <td align="right">-1.22E-10</td>
</tr>
<tr>
 <td>1.850</td>
 <td>0.9456111764</td>
 <td>0.94561117639912362</td>
 <td align="right">-2.02E-12</td>
</tr>
<tr>
 <td>1.855</td>
 <td>0.9471394637</td>
 <td>0.94713946380190617</td>
 <td align="right"> 9.43E-11</td>
</tr>
<tr>
 <td>1.860</td>
 <td>0.9486870417</td>
 <td>0.94868704167359708</td>
 <td align="right">-2.86E-11</td>
</tr>
<tr>
 <td>1.865</td>
 <td>0.9502539389</td>
 <td>0.95025393889348797</td>
 <td align="right">-1.33E-11</td>
</tr>
<tr>
 <td>1.870</td>
 <td>0.9518401855</td>
 <td>0.95184018551169203</td>
 <td align="right"> 9.61E-12</td>
</tr>
<tr>
 <td>1.875</td>
 <td>0.9534458127</td>
 <td>0.95344581274503493</td>
 <td align="right"> 5.77E-11</td>
</tr>
<tr>
 <td>1.880</td>
 <td>0.9550708530</td>
```

```
 <td>0.95507085297311556</td>
 <td align="right">-2.73E-11</td>
</tr>
<tr>
 <td>1.885</td>
 <td>0.9567153398</td>
 <td>0.95671533973453671</td>
 <td align="right">-6.02E-11</td>
</tr>
<tr>
 <td>1.890</td>
 <td>0.9583793077</td>
 <td>0.95837930772329927</td>
 <td align="right"> 1.97E-11</td>
</tr>
<tr>
 <td>1.895</td>
 <td>0.9600627927</td>
 <td>0.960062792785362</td>
 <td align="right"> 8.60E-11</td>
</tr>
<tr>
 <td>1.900</td>
 <td>0.9617658319</td>
 <td>0.96176583191536336</td>
 <td align="right"> 2.60E-11</td>
</tr>
<tr>
 <td>1.905</td>
 <td>0.9634884632</td>
 <td>0.96348846325350124</td>
 <td align="right"> 5.75E-11</td>
</tr>
<tr>
 <td>1.910</td>
 <td>0.9652307261</td>
 <td>0.96523072608257054</td>
 <td align="right">-3.05E-11</td>
</tr>
<tr>
 <td>1.915</td>
 <td>0.9669926608</td>
 <td>0.96699266080453206</td>
 <td align="right"> 5.78E-13</td>
</tr>
<tr>
 <td>1.920</td>
 <td>0.9687743090</td>
 <td>0.96877430902013406</td>
 <td align="right"> 1.66E-11</td>
</tr>
<tr>
 <td>1.925</td>
 <td>0.9705757134</td>
```

```
 <td>0.97057571340334281</td>
 <td align="right">-3.67E-12</td>
</tr>
<tr>
 <td>1.930</td>
 <td>0.9723969178</td>
 <td>0.9723969177808085</td>
 <td align="right">-5.87E-12</td>
</tr>
<tr>
 <td>1.935</td>
 <td>0.9742379672</td>
 <td>0.97423796710926569</td>
 <td align="right">-8.59E-11</td>
</tr>
<tr>
 <td>1.940</td>
 <td>0.9760989075</td>
 <td>0.97609890747347727</td>
 <td align="right">-2.67E-11</td>
</tr>
<tr>
 <td>1.945</td>
 <td>0.9779797861</td>
 <td>0.97797978608432246</td>
 <td align="right">-2.76E-11</td>
</tr>
<tr>
 <td>1.950</td>
 <td>0.9798806513</td>
 <td>0.9798806512770295</td>
 <td align="right">-3.65E-11</td>
</tr>
<tr>
 <td>1.955</td>
 <td>0.9818015524</td>
 <td>0.98180155250954815</td>
 <td align="right"> 1.02E-10</td>
</tr>
<tr>
 <td>1.960</td>
 <td>0.9837425404</td>
 <td>0.98374254036106346</td>
 <td align="right">-5.01E-11</td>
</tr>
<tr>
 <td>1.965</td>
 <td>0.9857036664</td>
 <td>0.985703666530647</td>
 <td align="right"> 1.27E-10</td>
</tr>
<tr>
 <td>1.970</td>
 <td>0.9876849838</td>
```

```
  <td>0.98768498383604675</td>
  <td align="right"> 4.68E-11</td>
 </tr>
 <tr>
  <td>1.975</td>
  <td>0.9896865462</td>
  <td>0.98968654618919183</td>
  <td align="right">-1.77E-11</td>
 </tr>
 <tr>
  <td>1.980</td>
  <td>0.9917084087</td>
  <td>0.99170840868869103</td>
  <td align="right">-3.22E-12</td>
 </tr>
 <tr>
  <td>1.985</td>
  <td>0.9937506274</td>
  <td>0.9937506274792185</td>
  <td align="right"> 6.46E-11</td>
 </tr>
 <tr>
  <td>1.990</td>
  <td>0.9958132598</td>
  <td>0.99581325984380575</td>
  <td align="right"> 4.71E-11</td>
 </tr>
 <tr>
  <td>1.995</td>
  <td>0.9978963643</td>
  <td>0.99789636418011041</td>
  <td align="right">-1.27E-10</td>
 </tr>
</table>


<h4>The Psi Function</h4>

This table was constructed from the published values in the
Handbook of Mathematical Functions, by Milton Abramowitz
and Irene A. Stegun, by Dover (1965), pp 267-270.

Axiom implements the polygamma function which allows for multiple
derivatives. The Psi function is a special case of the polygamma
function for zero derivatives. For the purpose of this table it
is defined as:
<pre>
   Psi(x) == polygamma(0,x)
</pre>

The first column is the point where the Gamma function is evaluated.
The second column is the value reported in the Handbook.
The third column is the actual value computed by Axiom at the given point.
The fourth column is the difference of Axiom's value and the Handbook value.
```

```
<table border="1">
 <tr>
  <th>point</th>
  <th>Handbook Value</th>
  <th>Axiom Computed Value</th>
  <th>Difference</th>
 </tr>
 <tr>
  <td>1.000</td>
  <td>-0.5772156649</td>
  <td>-0.57721566490153275</td>
  <td align="right">-1.53E-12</td>
 </tr>
 <tr>
  <td>1.005</td>
  <td>-0.5690209113</td>
  <td>-0.56902091134438304</td>
  <td align="right"> -4.43E-11</td>
 </tr>
 <tr>
  <td>1.010</td>
  <td>-0.5608854579</td>
  <td>-0.56088545786867472</td>
  <td align="right"> 3.13E-11</td>
 </tr>
 <tr>
  <td>1.015</td>
  <td>-0.5528085156</td>
  <td>-0.55280851559434629</td>
  <td align="right"> 5.65E-12</td>
 </tr>
 <tr>
  <td>1.020</td>
  <td>-0.5447893105</td>
  <td>-0.54478931045617984</td>
  <td align="right"> 4.38E-11</td>
 </tr>
 <tr>
  <td>1.025</td>
  <td>-0.5368270828</td>
  <td>-0.53682708284938863</td>
  <td align="right"> -4.93E-11</td>
 </tr>
 <tr>
  <td>1.030</td>
  <td>-0.5289210873</td>
  <td>-0.5289210872854303</td>
  <td align="right"> 1.45E-11</td>
 </tr>
 <tr>
  <td>1.035</td>
  <td>-0.5210705921</td>
  <td>-0.52107059205771</td>
```

```
  <td align="right"> 4.22E-11</td>
 </tr>
 <tr>
  <td>1.040</td>
  <td>-0.5132748789</td>
  <td>-0.51327487891683021</td>
  <td align="right"> -1.68E-11</td>
 </tr>
 <tr>
  <td>1.045</td>
  <td>-0.5055332428</td>
  <td>-0.50553324275508449</td>
  <td align="right"> 4.49E-11</td>
 </tr>
 <tr>
  <td>1.050</td>
  <td>-0.4978449913</td>
  <td>-0.49784499129987031</td>
  <td align="right"> 1.29E-13</td>
 </tr>
 <tr>
  <td>1.055</td>
  <td>-0.4902094448</td>
  <td>-0.49020944481574569</td>
  <td align="right"> -1.57E-11</td>
 </tr>
 <tr>
  <td>1.060</td>
  <td>-0.4826259358</td>
  <td>-0.48262593581482538</td>
  <td align="right"> -1.48E-11</td>
 </tr>
 <tr>
  <td>1.065</td>
  <td>-0.4750938088</td>
  <td>-0.47509380877526647</td>
  <td align="right"> 2.47E-11</td>
 </tr>
 <tr>
  <td>1.070</td>
  <td>-0.4676124199</td>
  <td>-0.46761241986755342</td>
  <td align="right"> 3.24E-11</td>
 </tr>
 <tr>
  <td>1.075</td>
  <td>-0.4601811367</td>
  <td>-0.4601811366883593</td>
  <td align="right"> 1.16E-11</td>
 </tr>
 <tr>
  <td>1.080</td>
  <td>-0.4527993380</td>
  <td>-0.45279933800171246</td>
```

```
 <td align="right"> -1.71E-12</td>
</tr>
<tr>
 <td>1.085</td>
 <td>-0.4454664135</td>
 <td>-0.44546641348725191</td>
 <td align="right"> 1.27E-11</td>
</tr>
<tr>
 <td>1.090</td>
 <td>-0.4381817635</td>
 <td>-0.43818176349533489</td>
 <td align="right"> 4.66E-12</td>
</tr>
<tr>
 <td>1.095</td>
 <td>-0.4309447988</td>
 <td>-0.43094479880878706</td>
 <td align="right"> -8.78E-12</td>
</tr>
<tr>
 <td>1.100</td>
 <td>-0.4237549404</td>
 <td>-0.42375494041107653</td>
 <td align="right"> -1.10E-11</td>
</tr>
<tr>
 <td>1.105</td>
 <td>-0.4166116193</td>
 <td>-0.41661161926071655</td>
 <td align="right"> 3.92E-11</td>
</tr>
<tr>
 <td>1.110</td>
 <td>-0.4095142761</td>
 <td>-0.40951427607169383</td>
 <td align="right"> 2.83E-11</td>
</tr>
<tr>
 <td>1.115</td>
 <td>-0.4024623611</td>
 <td>-0.40246236109974648</td>
 <td align="right"> 2.53E-13</td>
</tr>
<tr>
 <td>1.120</td>
 <td>-0.3954553339</td>
 <td>-0.39545533393429283</td>
 <td align="right"> -3.42E-11</td>
</tr>
<tr>
 <td>1.125</td>
 <td>-0.3884926633</td>
 <td>-0.38849266329585463</td>
```

```
 <td align="right"> 4.14E-12</td>
</tr>
<tr>
 <td>1.130</td>
 <td>-0.3815738268</td>
 <td>-0.38157382683879215</td>
 <td align="right"> -3.87E-11</td>
</tr>
<tr>
 <td>1.135</td>
 <td>-0.3746983110</td>
 <td>-0.37469831095919082</td>
 <td align="right"> 4.08E-11</td>
</tr>
<tr>
 <td>1.140</td>
 <td>-0.3678656106</td>
 <td>-0.36786561060774969</td>
 <td align="right"> -7.74E-12</td>
</tr>
<tr>
 <td>1.145</td>
 <td>-0.3610752291</td>
 <td>-0.361075229107509</td>
 <td align="right"> -7.50E-12</td>
</tr>
<tr>
 <td>1.150</td>
 <td>-0.3543266780</td>
 <td>-0.35432667797627904</td>
 <td align="right"> 2.37E-11</td>
</tr>
<tr>
 <td>1.155</td>
 <td>-0.3476194768</td>
 <td>-0.34761947675362337</td>
 <td align="right"> 4.63E-11</td>
</tr>
<tr>
 <td>1.160</td>
 <td>-0.3409531528</td>
 <td>-0.34095315283226135</td>
 <td align="right"> -3.22E-11</td>
</tr>
<tr>
 <td>1.165</td>
 <td>-0.3343272413</td>
 <td>-0.3343272412937619</td>
 <td align="right"> 6.23E-12</td>
</tr>
<tr>
 <td>1.170</td>
 <td>-0.3277412847</td>
 <td>-0.3277412847483927</td>
```

```
 <td align="right"> -4.83E-11</td>
</tr>
<tr>
 <td>1.175</td>
 <td>-0.3211948332</td>
 <td>-0.3211948331790081</td>
 <td align="right"> 2.09E-11</td>
</tr>
<tr>
 <td>1.180</td>
 <td>-0.3146874438</td>
 <td>-0.31468744378886082</td>
 <td align="right"> 1.11E-11</td>
</tr>
<tr>
 <td>1.185</td>
 <td>-0.3082186809</td>
 <td>-0.30821868085320625</td>
 <td align="right"> 4.67E-11</td>
</tr>
<tr>
 <td>1.190</td>
 <td>-0.3017881156</td>
 <td>-0.30178811557461016</td>
 <td align="right"> 2.53E-11</td>
</tr>
<tr>
 <td>1.195</td>
 <td>-0.2953953259</td>
 <td>-0.2953953259418296</td>
 <td align="right"> -4.18E-11</td>
</tr>
<tr>
 <td>1.200</td>
 <td>-0.2890398966</td>
 <td>-0.28903989659218843</td>
 <td align="right"> 7.81E-12</td>
</tr>
<tr>
 <td>1.205</td>
 <td>-0.2827214187</td>
 <td>-0.28272141867731704</td>
 <td align="right"> 2.26E-11</td>
</tr>
<tr>
 <td>1.210</td>
 <td>-0.2764394897</td>
 <td>-0.2764394897321919</td>
 <td align="right"> -3.21E-11</td>
</tr>
<tr>
 <td>1.215</td>
 <td>-0.2701937135</td>
 <td>-0.27019371354735244</td>
```

```
 <td align="right"> -4.73E-11</td>
</tr>
<tr>
 <td>1.220</td>
 <td>-0.2639837000</td>
 <td>-0.26398370004422023</td>
 <td align="right"> -4.42E-11</td>
</tr>
<tr>
 <td>1.225</td>
 <td>-0.2578090652</td>
 <td>-0.25780906515343338</td>
 <td align="right"> 4.65E-11</td>
</tr>
<tr>
 <td>1.230</td>
 <td>-0.2516694307</td>
 <td>-0.25166943069609982</td>
 <td align="right"> 3.90E-12</td>
</tr>
<tr>
 <td>1.235</td>
 <td>-0.2455644243</td>
 <td>-0.24556442426789726</td>
 <td align="right"> 3.21E-11</td>
</tr>
<tr>
 <td>1.240</td>
 <td>-0.2394936791</td>
 <td>-0.23949367912593666</td>
 <td align="right"> -2.59E-11</td>
</tr>
<tr>
 <td>1.245</td>
 <td>-0.2334568341</td>
 <td>-0.23345683407831253</td>
 <td align="right"> 2.16E-11</td>
</tr>
<tr>
 <td>1.250</td>
 <td>-0.2274535334</td>
 <td>-0.22745353337626528</td>
 <td align="right"> 2.37E-11</td>
</tr>
<tr>
 <td>1.255</td>
 <td>-0.2214834266</td>
 <td>-0.22148342660888165</td>
 <td align="right"> -8.88E-12</td>
</tr>
<tr>
 <td>1.260</td>
 <td>-0.2155461686</td>
 <td>-0.21554616860026521</td>
```

```
 <td align="right"> -2.65E-13</td>
</tr>
<tr>
 <td>1.265</td>
 <td>-0.2096414193</td>
 <td>-0.20964141930911384</td>
 <td align="right"> -9.11E-12</td>
</tr>
<tr>
 <td>1.270</td>
 <td>-0.2037688437</td>
 <td>-0.20376884373062343</td>
 <td align="right"> -3.06E-11</td>
</tr>
<tr>
 <td>1.275</td>
 <td>-0.1979281118</td>
 <td>-0.19792811180067393</td>
 <td align="right"> -6.73E-13</td>
</tr>
<tr>
 <td>1.280</td>
 <td>-0.1921188983</td>
 <td>-0.19211889830222173</td>
 <td align="right"> -2.22E-12</td>
</tr>
<tr>
 <td>1.285</td>
 <td>-0.1863408828</td>
 <td>-0.18634088277384209</td>
 <td align="right"> 2.61E-11</td>
</tr>
<tr>
 <td>1.290</td>
 <td>-0.1805937494</td>
 <td>-0.1805937494203691</td>
 <td align="right"> -2.03E-11</td>
</tr>
<tr>
 <td>1.295</td>
 <td>-0.1748771870</td>
 <td>-0.17487718702556942</td>
 <td align="right"> -2.55E-11</td>
</tr>
<tr>
 <td>1.300</td>
 <td>-0.1691908889</td>
 <td>-0.16919088886679934</td>
 <td align="right"> 3.32E-11</td>
</tr>
<tr>
 <td>1.305</td>
 <td>-0.1635345526</td>
 <td>-0.163534552631597</td>
```

```
 <td align="right"> -3.15E-11</td>
</tr>
<tr>
 <td>1.310</td>
 <td>-0.1579078803</td>
 <td>-0.15790788033614178</td>
 <td align="right"> -3.61E-11</td>
</tr>
<tr>
 <td>1.315</td>
 <td>-0.1523105782</td>
 <td>-0.15231057824555994</td>
 <td align="right"> -4.55E-11</td>
</tr>
<tr>
 <td>1.320</td>
 <td>-0.1467423568</td>
 <td>-0.1467423567959959</td>
 <td align="right"> 4.00E-12</td>
</tr>
<tr>
 <td>1.325</td>
 <td>-0.1412029305</td>
 <td>-0.14120293051842803</td>
 <td align="right"> -1.84E-11</td>
</tr>
<tr>
 <td>1.330</td>
 <td>-0.1356920180</td>
 <td>-0.13569201796416941</td>
 <td align="right"> 3.58E-11</td>
</tr>
<tr>
 <td>1.335</td>
 <td>-0.1302093416</td>
 <td>-0.13020934163201769</td>
 <td align="right"> -3.20E-11</td>
</tr>
<tr>
 <td>1.340</td>
 <td>-0.1247546279</td>
 <td>-0.12475462789700376</td>
 <td align="right"> 2.99E-12</td>
</tr>
<tr>
 <td>1.345</td>
 <td>-0.1193276069</td>
 <td>-0.11932760694070754</td>
 <td align="right"> -4.07E-11</td>
</tr>
<tr>
 <td>1.350</td>
 <td>-0.1139280127</td>
 <td>-0.11392801268308839</td>
```

```
 <td align="right"> 1.69E-11</td>
</tr>
<tr>
 <td>1.355</td>
 <td>-0.1085555827</td>
 <td>-0.10855558271580501</td>
 <td align="right"> -1.58E-11</td>
</tr>
<tr>
 <td>1.360</td>
 <td>-0.1032100582</td>
 <td>-0.10321005823697738</td>
 <td align="right"> -3.69E-11</td>
</tr>
<tr>
 <td>1.365</td>
 <td>-0.0978911840</td>
 <td>-0.097891183987354968</td>
 <td align="right"> 1.26E-11</td>
</tr>
<tr>
 <td>1.370</td>
 <td>-0.0925987082</td>
 <td>-0.092598708187860979</td>
 <td align="right"> 1.21E-11</td>
</tr>
<tr>
 <td>1.375</td>
 <td>-0.0873323825</td>
 <td>-0.087332382478473081</td>
 <td align="right"> 2.15E-11</td>
</tr>
<tr>
 <td>1.380</td>
 <td>-0.0820919619</td>
 <td>-0.082091961858406615</td>
 <td align="right"> 4.15E-11</td>
</tr>
<tr>
 <td>1.385</td>
 <td>-0.0768772046</td>
 <td>-0.076877204627574525</td>
 <td align="right"> -2.75E-11</td>
</tr>
<tr>
 <td>1.390</td>
 <td>-0.0716878723</td>
 <td>-0.071687872329281643</td>
 <td align="right"> -2.92E-11</td>
</tr>
<tr>
 <td>1.395</td>
 <td>-0.0665237297</td>
 <td>-0.066523729694132228</td>
```

```
 <td align="right"> 5.86E-12</td>
</tr>
<tr>
 <td>1.400</td>
 <td>-0.0613845446</td>
 <td>-0.061384544585116108</td>
 <td align="right"> 1.48E-11</td>
</tr>
<tr>
 <td>1.405</td>
 <td>-0.0562700879</td>
 <td>-0.056270087943841696</td>
 <td align="right"> -4.38E-11</td>
</tr>
<tr>
 <td>1.410</td>
 <td>-0.0511801337</td>
 <td>-0.051180133737897426</td>
 <td align="right"> -3.78E-11</td>
</tr>
<tr>
 <td>1.415</td>
 <td>-0.0461144589</td>
 <td>-0.04.6114458909301992</td>
 <td align="right"> -9.30E-12</td>
</tr>
<tr>
 <td>1.420</td>
 <td>-0.0410728433</td>
 <td>-0.041072843324024277</td>
 <td align="right"> -2.40E-11</td>
</tr>
<tr>
 <td>1.425</td>
 <td>-0.0360550697</td>
 <td>-0.036055069722547906</td>
 <td align="right"> -2.25E-11</td>
</tr>
<tr>
 <td>1.430</td>
 <td>-0.0310609237</td>
 <td>-0.031060923671447194</td>
 <td align="right"> 2.85E-11</td>
</tr>
<tr>
 <td>1.435</td>
 <td>-0.0260901935</td>
 <td>-0.02609019351596098</td>
 <td align="right"> -1.59E-11</td>
</tr>
<tr>
 <td>1.440</td>
 <td>-0.0211426703</td>
 <td>-0.021142670333530678</td>
```

```
 <td align="right"> -3.35E-11</td>
</tr>
<tr>
 <td>1.445</td>
 <td>-0.0162181479</td>
 <td>-0.016218147888283685</td>
 <td align="right"> 1.17E-11</td>
</tr>
<tr>
 <td>1.450</td>
 <td>-0.0113164226</td>
 <td>-0.011316422586445718</td>
 <td align="right"> 1.35E-11</td>
</tr>
<tr>
 <td>1.455</td>
 <td>-0.0064372934</td>
 <td>-0.0064372934326406561</td>
 <td align="right"> -3.26E-11</td>
</tr>
<tr>
 <td>1.460</td>
 <td>-0.0015805620</td>
 <td>-0.0015805619870833398</td>
 <td align="right"> 1.29E-11</td>
</tr>
<tr>
 <td>1.465</td>
 <td>0.0032539677</td>
 <td>0.0032539676763745362</td>
 <td align="right"> -2.36E-11</td>
</tr>
<tr>
 <td>1.470</td>
 <td>0.0080664890</td>
 <td>0.0080664890113649745</td>
 <td align="right"> 1.13E-11</td>
</tr>
<tr>
 <td>1.475</td>
 <td>0.0128571930</td>
 <td>0.012857193039295334</td>
 <td align="right"> 3.92E-11</td>
</tr>
<tr>
 <td>1.480</td>
 <td>0.0176262684</td>
 <td>0.017626268388849287</td>
 <td align="right"> -1.11E-11</td>
</tr>
<tr>
 <td>1.485</td>
 <td>0.0223739013</td>
 <td>0.022373901334705404</td>
```

```
 <td align="right"> 3.47E-11</td>
</tr>
<tr>
 <td>1.490</td>
 <td>0.0271002758</td>
 <td>0.027100275835486465</td>
 <td align="right"> 3.54E-11</td>
</tr>
<tr>
 <td>1.495</td>
 <td>0.0318055736</td>
 <td>0.031805573570971468</td>
 <td align="right"> -2.90E-11</td>
</tr>
<tr>
 <td>1.500</td>
 <td>0.0364899740</td>
 <td>0.036489973978576673</td>
 <td align="right"> -2.14E-11</td>
</tr>
<tr>
 <td>1.505</td>
 <td>0.0411536543</td>
 <td>0.041153654289123542</td>
 <td align="right"> -1.08E-11</td>
</tr>
<tr>
 <td>1.510</td>
 <td>0.0457967896</td>
 <td>0.045796789561914686</td>
 <td align="right"> -3.80E-11</td>
</tr>
<tr>
 <td>1.515</td>
 <td>0.0504195527</td>
 <td>0.050419552719128236</td>
 <td align="right"> 1.91E-11</td>
</tr>
<tr>
 <td>1.520</td>
 <td>0.0550221146</td>
 <td>0.055022114579551307</td>
 <td align="right"> -2.04E-11</td>
</tr>
<tr>
 <td>1.525</td>
 <td>0.0596046439</td>
 <td>0.05960464389166209</td>
 <td align="right"> -8.33E-12</td>
</tr>
<tr>
 <td>1.530</td>
 <td>0.0641673074</td>
 <td>0.064167307366077231</td>
```

```
  <td align="right"> -3.39E-11</td>
 </tr>
 <tr>
  <td>1.535</td>
  <td>0.0687102697</td>
  <td>0.068710269707385141</td>
  <td align="right"> 7.38E-12</td>
 </tr>
 <tr>
  <td>1.540</td>
  <td>0.0732336936</td>
  <td>0.073233693645366138</td>
  <td align="right"> 4.53E-11</td>
 </tr>
 <tr>
  <td>1.545</td>
  <td>0.0777377300</td>
  <td>0.077737739965624497</td>
  <td align="right"> 9.96E-9</td>
 </tr>
 <tr>
  <td>1.550</td>
  <td>0.0822225675</td>
  <td>0.082222567539644631</td>
  <td align="right"> 3.96E-11</td>
 </tr>
 <tr>
  <td>1.555</td>
  <td>0.0866883334</td>
  <td>0.086688333354268288</td>
  <td align="right"> -4.57E-11</td>
 </tr>
 <tr>
  <td>1.560</td>
  <td>0.0911351925</td>
  <td>0.091135192540635401</td>
  <td align="right"> 4.06E-11</td>
 </tr>
 <tr>
  <td>1.565</td>
  <td>0.0955632984</td>
  <td>0.095563298402570163</td>
  <td align="right"> 2.57E-12</td>
 </tr>
 <tr>
  <td>1.570</td>
  <td>0.0999728024</td>
  <td>0.099972802444444731</td>
  <td align="right"> 4.44E-11</td>
 </tr>
 <tr>
  <td>1.575</td>
  <td>0.1043638544</td>
  <td>0.10436385439851947</td>
```

```
 <td align="right"> -1.48E-12</td>
</tr>
<tr>
 <td>1.580</td>
 <td>0.1087366023</td>
 <td>0.10873660225178161</td>
 <td align="right"> -4.82E-11</td>
</tr>
<tr>
 <td>1.585</td>
 <td>0.1130911923</td>
 <td>0.11309119227228603</td>
 <td align="right"> -2.77E-11</td>
</tr>
<tr>
 <td>1.590</td>
 <td>0.1174277690</td>
 <td>0.11742776903501095</td>
 <td align="right"> 3.50E-11</td>
</tr>
<tr>
 <td>1.595</td>
 <td>0.1217464754</td>
 <td>0.12174647544723916</td>
 <td align="right"> 4.72E-11</td>
</tr>
<tr>
 <td>1.600</td>
 <td>0.1260474528</td>
 <td>0.12604745277347584</td>
 <td align="right"> -2.65E-11</td>
</tr>
<tr>
 <td>1.605</td>
 <td>0.1303308407</td>
 <td>0.13033084065991318</td>
 <td align="right"> -4.00E-11</td>
</tr>
<tr>
 <td>1.610</td>
 <td>0.1345967772</td>
 <td>0.13459677715844587</td>
 <td align="right"> -4.15E-11</td>
</tr>
<tr>
 <td>1.615</td>
 <td>0.1388453988</td>
 <td>0.13884539875025736</td>
 <td align="right"> -4.97E-11</td>
</tr>
<tr>
 <td>1.620</td>
 <td>0.1430768404</td>
 <td>0.14307684036898005</td>
```

```
 <td align="right"> -3.10E-11</td>
</tr>
<tr>
 <td>1.625</td>
 <td>0.1472912354</td>
 <td>0.14729123542343325</td>
 <td align="right"> 2.34E-11</td>
</tr>
<tr>
 <td>1.630</td>
 <td>0.1514887158</td>
 <td>0.15148871581995815</td>
 <td align="right"> 1.99E-11</td>
</tr>
<tr>
 <td>1.635</td>
 <td>0.1556694120</td>
 <td>0.15566941198435302</td>
 <td align="right"> -1.56E-11</td>
</tr>
<tr>
 <td>1.640</td>
 <td>0.1598334529</td>
 <td>0.15983345288341522</td>
 <td align="right"> -1.65E-11</td>
</tr>
<tr>
 <td>1.645</td>
 <td>0.1639809660</td>
 <td>0.16398096604610457</td>
 <td align="right"> 4.61E-11</td>
</tr>
<tr>
 <td>1.650</td>
 <td>0.1681120776</td>
 <td>0.16811207758432767</td>
 <td align="right"> -1.56E-11</td>
</tr>
<tr>
 <td>1.655</td>
 <td>0.1722269122</td>
 <td>0.17222691221335784</td>
 <td align="right"> 1.33E-11</td>
</tr>
<tr>
 <td>1.660</td>
 <td>0.1763255933</td>
 <td>0.17632559327189457</td>
 <td align="right"> -2.81E-11</td>
</tr>
<tr>
 <td>1.665</td>
 <td>0.1804082427</td>
 <td>0.18040824274177392</td>
```

```
 <td align="right"> 4.17E-11</td>
</tr>
<tr>
 <td>1.670</td>
 <td>0.1844749813</td>
 <td>0.1844749812673292</td>
 <td align="right"> -3.26E-11</td>
</tr>
<tr>
 <td>1.675</td>
 <td>0.1885259282</td>
 <td>0.18852592817442249</td>
 <td align="right"> -2.55E-11</td>
</tr>
<tr>
 <td>1.680</td>
 <td>0.1925612015</td>
 <td>0.19256120148913258</td>
 <td align="right"> -1.08E-11</td>
</tr>
<tr>
 <td>1.685</td>
 <td>0.1965809180</td>
 <td>0.19658091795613342</td>
 <td align="right"> -4.38E-11</td>
</tr>
<tr>
 <td>1.690</td>
 <td>0.2005851931</td>
 <td>0.20058519305674649</td>
 <td align="right"> -4.32E-11</td>
</tr>
<tr>
 <td>1.695</td>
 <td>0.2045741410</td>
 <td>0.20457414102668603</td>
 <td align="right"> 2.66E-11</td>
</tr>
<tr>
 <td>1.700</td>
 <td>0.2085478749</td>
 <td>0.20854787487349435</td>
 <td align="right"> -2.65E-11</td>
</tr>
<tr>
 <td>1.705</td>
 <td>0.2125065064</td>
 <td>0.21250650639368796</td>
 <td align="right"> -6.31E-12</td>
</tr>
<tr>
 <td>1.710</td>
 <td>0.2164501462</td>
 <td>0.21645014618960501</td>
```

```
 <td align="right"> -1.03E-11</td>
</tr>
<tr>
 <td>1.715</td>
 <td>0.2203789037</td>
 <td>0.2203789036859658</td>
 <td align="right"> -1.40E-11</td>
</tr>
<tr>
 <td>1.720</td>
 <td>0.2242928871</td>
 <td>0.22429288714615725</td>
 <td align="right"> 4.61E-11</td>
</tr>
<tr>
 <td>1.725</td>
 <td>0.2281922037</td>
 <td>0.22819220368823745</td>
 <td align="right"> -1.17E-11</td>
</tr>
<tr>
 <td>1.730</td>
 <td>0.2320769593</td>
 <td>0.23207695930067274</td>
 <td align="right"> 6.72E-13</td>
</tr>
<tr>
 <td>1.735</td>
 <td>0.2359472589</td>
 <td>0.23594725885781176</td>
 <td align="right"> -4.21E-11</td>
</tr>
<tr>
 <td>1.740</td>
 <td>0.2398032061</td>
 <td>0.23980320613509676</td>
 <td align="right"> 3.50E-11</td>
</tr>
<tr>
 <td>1.745</td>
 <td>0.2436449038</td>
 <td>0.24364490382402559</td>
 <td align="right"> 2.40E-11</td>
</tr>
<tr>
 <td>1.750</td>
 <td>0.2474724535</td>
 <td>0.2474724535468612</td>
 <td align="right"> 4.68E-11</td>
</tr>
<tr>
 <td>1.755</td>
 <td>0.2512859559</td>
 <td>0.25128595587109781</td>
```

```
 <td align="right"> -2.89E-11</td>
</tr>
<tr>
 <td>1.760</td>
 <td>0.2550855103</td>
 <td>0.25508551032368809</td>
 <td align="right"> 2.36E-11</td>
</tr>
<tr>
 <td>1.765</td>
 <td>0.2588712154</td>
 <td>0.25887121540503744</td>
 <td align="right"> 5.03E-12</td>
</tr>
<tr>
 <td>1.770</td>
 <td>0.2626431686</td>
 <td>0.26264316860276249</td>
 <td align="right"> 2.76E-12</td>
</tr>
<tr>
 <td>1.775</td>
 <td>0.2664014664</td>
 <td>0.2664014664052331</td>
 <td align="right"> 5.23E-12</td>
</tr>
<tr>
 <td>1.780</td>
 <td>0.2701462043</td>
 <td>0.27014620431488368</td>
 <td align="right"> 1.48E-11</td>
</tr>
<tr>
 <td>1.785</td>
 <td>0.2738774769</td>
 <td>0.27387747686131236</td>
 <td align="right"> -3.86E-11</td>
</tr>
<tr>
 <td>1.790</td>
 <td>0.2775953776</td>
 <td>0.27759537761416786</td>
 <td align="right"> 1.41E-11</td>
</tr>
<tr>
 <td>1.795</td>
 <td>0.2812999992</td>
 <td>0.2812999991958266</td>
 <td align="right"> -4.17E-12</td>
</tr>
<tr>
 <td>1.800</td>
 <td>0.2849914333</td>
 <td>0.2849914332938619</td>
```

```
 <td align="right"> -6.13E-12</td>
</tr>
<tr>
 <td>1.805</td>
 <td>0.2886697707</td>
 <td>0.28866977067331689</td>
 <td align="right"> -2.66E-11</td>
</tr>
<tr>
 <td>1.810</td>
 <td>0.2923351012</td>
 <td>0.29233510118877948</td>
 <td align="right"> -1.12E-11</td>
</tr>
<tr>
 <td>1.815</td>
 <td>0.2959875138</td>
 <td>0.29598751379626109</td>
 <td align="right"> -3.73E-12</td>
</tr>
<tr>
 <td>1.820</td>
 <td>0.2996270966</td>
 <td>0.29962709656488773</td>
 <td align="right"> -3.51E-11</td>
</tr>
<tr>
 <td>1.825</td>
 <td>0.3032539367</td>
 <td>0.30325393668840539</td>
 <td align="right"> -1.15E-11</td>
</tr>
<tr>
 <td>1.830</td>
 <td>0.3068681205</td>
 <td>0.30686812049650136</td>
 <td align="right"> -3.49E-12</td>
</tr>
<tr>
 <td>1.835</td>
 <td>0.3104697335</td>
 <td>0.31046973346594764</td>
 <td align="right"> -3.40E-11</td>
</tr>
<tr>
 <td>1.840</td>
 <td>0.3140588602</td>
 <td>0.31405886023156859</td>
 <td align="right"> 3.15E-11</td>
</tr>
<tr>
 <td>1.845</td>
 <td>0.3176355846</td>
 <td>0.31763558459703256</td>
```

```
 <td align="right"> -2.96E-12</td>
</tr>
<tr>
 <td>1.850</td>
 <td>0.3211999895</td>
 <td>0.32119998954547946</td>
 <td align="right"> 4.54E-11</td>
</tr>
<tr>
 <td>1.855</td>
 <td>0.3247521572</td>
 <td>0.32475215724997797</td>
 <td align="right"> 4.99E-11</td>
</tr>
<tr>
 <td>1.860</td>
 <td>0.3282921691</td>
 <td>0.32829216908382075</td>
 <td align="right"> -1.61E-11</td>
</tr>
<tr>
 <td>1.865</td>
 <td>0.3318201056</td>
 <td>0.33182010563065989</td>
 <td align="right"> 3.06E-11</td>
</tr>
<tr>
 <td>1.870</td>
 <td>0.3353360467</td>
 <td>0.33533604669448569</td>
 <td align="right"> -5.51E-12</td>
</tr>
<tr>
 <td>1.875</td>
 <td>0.3388400713</td>
 <td>0.33884007130944738</td>
 <td align="right"> 9.44E-12</td>
</tr>
<tr>
 <td>1.880</td>
 <td>0.3423322577</td>
 <td>0.34233225774952925</td>
 <td align="right"> 4.95E-11</td>
</tr>
<tr>
 <td>1.885</td>
 <td>0.3458126835</td>
 <td>0.34581268353806771</td>
 <td align="right"> 3.80E-11</td>
</tr>
<tr>
 <td>1.890</td>
 <td>0.3492814255</td>
 <td>0.34928142545713492</td>
```

```
 <td align="right"> -4.28E-11</td>
</tr>
<tr>
 <td>1.895</td>
 <td>0.3527385596</td>
 <td>0.35273855955676792</td>
 <td align="right"> -4.32E-11</td>
</tr>
<tr>
 <td>1.900</td>
 <td>0.3561841612</td>
 <td>0.35618416116406026</td>
 <td align="right"> -3.59E-11</td>
</tr>
<tr>
 <td>1.905</td>
 <td>0.3596183049</td>
 <td>0.35961830489211799</td>
 <td align="right"> -7.88E-12</td>
</tr>
<tr>
 <td>1.910</td>
 <td>0.3630410646</td>
 <td>0.36304106464888108</td>
 <td align="right"> 4.88E-11</td>
</tr>
<tr>
 <td>1.915</td>
 <td>0.3664525136</td>
 <td>0.36645251364580167</td>
 <td align="right"> 4.58E-11</td>
</tr>
<tr>
 <td>1.920</td>
 <td>0.3698527244</td>
 <td>0.36985272440640171</td>
 <td align="right"> 6.40E-12</td>
</tr>
<tr>
 <td>1.925</td>
 <td>0.3732417688</td>
 <td>0.37324176877469795</td>
 <td align="right"> -2.53E-11</td>
</tr>
<tr>
 <td>1.930</td>
 <td>0.3766197179</td>
 <td>0.37661971792349891</td>
 <td align="right"> 2.34E-11</td>
</tr>
<tr>
 <td>1.935</td>
 <td>0.3799866424</td>
 <td>0.37998664236258128</td>
```

```
 <td align="right"> -3.74E-11</td>
</tr>
<tr>
 <td>1.940</td>
 <td>0.3833426119</td>
 <td>0.38334261194674013</td>
 <td align="right"> 4.67E-11</td>
</tr>
<tr>
 <td>1.945</td>
 <td>0.3866876959</td>
 <td>0.38668769588372298</td>
 <td align="right"> -1.62E-11</td>
</tr>
<tr>
 <td>1.950</td>
 <td>0.3900219627</td>
 <td>0.39002196274204304</td>
 <td align="right"> 4.20E-11</td>
</tr>
<tr>
 <td>1.955</td>
 <td>0.3933454805</td>
 <td>0.39334548045868012</td>
 <td align="right"> -4.13E-11</td>
</tr>
<tr>
 <td>1.960</td>
 <td>0.3966583163</td>
 <td>0.39665831634666171</td>
 <td align="right"> 4.66E-11</td>
</tr>
<tr>
 <td>1.965</td>
 <td>0.3999605371</td>
 <td>0.39996053710254509</td>
 <td align="right"> 2.54E-12</td>
</tr>
<tr>
 <td>1.970</td>
 <td>0.4032522088</td>
 <td>0.40325220881377177</td>
 <td align="right"> 1.37E-11</td>
</tr>
<tr>
 <td>1.975</td>
 <td>0.4065333970</td>
 <td>0.40653339696592627</td>
 <td align="right"> -3.40E-11</td>
</tr>
<tr>
 <td>1.980</td>
 <td>0.4098041664</td>
 <td>0.40980416644989071</td>
```

```
  <td align="right"> 4.98E-11</td>
 </tr>
 <tr>
  <td>1.985</td>
  <td>0.4130645816</td>
  <td>0.41306458156888626</td>
  <td align="right"> -3.11E-11</td>
 </tr>
 <tr>
  <td>1.990</td>
  <td>0.4163147060</td>
  <td>0.41631470604541487</td>
  <td align="right"> 4.54E-11</td>
 </tr>
 <tr>
  <td>1.995</td>
  <td>0.4195546030</td>
  <td>0.41955460302810832</td>
  <td align="right"> 2.81E-11</td>
 </tr>
 <tr>
  <td>2.000</td>
  <td>0.4227843351</td>
  <td>0.42278433509846725</td>
  <td align="right"> -1.53E-12</td>
 </tr>
</table>
\getchunk{page foot}
```

draw.xhtml

— draw.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
      What would you like to draw?
    </td>
   </tr>
   <tr>
    <td>
     <center>
      <b>Two Dimensional Plots</b>
     </center>
    </td>
   </tr>
   <tr>
```

```
 <td>
  <a href="draw2donevariable.xhtml">
   A function of one variable
  </a>
 </td>
 <td>
  y = f(x)
 </td>
</tr>
<tr>
 <td>
  <a href="draw2ddefinedcurve.xhtml">
   A parametrically defined curve
  </a>
 </td>
 <td>
  (x(t), y(t))
 </td>
</tr>
<tr>
 <td>
  <a href="draw2dpolynomialequation.xhtml">
   A solution to a polynomial equation
  </a>
 </td>
 <td>
  p(x,y) = 0
 </td>
</tr>
<tr>
 <td>
 </td>
</tr>
<tr>
 <td>
  <center>
   <b>Three Dimensional Plots</b>
  </center>
 </td>
</tr>
<tr>
 <td>
  <a href="draw3dtwovariable.xhtml">
   A function of two variable
  </a>
 </td>
 <td>
  y = f(x,y)
 </td>
</tr>
<tr>
 <td>
  <a href="draw3ddefinedtube.xhtml">
   A parametrically defined tube
```

```
     </a>
    </td>
    <td>
     (x(t), y(t), z(t))
    </td>
   </tr>
   <tr>
    <td>
     <a href="draw3ddefinedsurface.xhtml">
      A parametrically defined surface
     </a>
    </td>
    <td>
     (x(u,v), y(u,v), z(u,v))
    </td>
   </tr>
   <tr>
    <td>
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

draw2donevariable.xhtml

— draw2donevariable.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function').value;
    myvar = document.getElementById('var').value;
    myfrom = document.getElementById('range1').value;
    myto = document.getElementById('range2').value;
    mytitle = document.getElementById('title1').value;
    if (mytitle == "") {
     ans = 'draw('+myfunc+','+myvar+'='+myfrom+'..'+myto+')';
    } else {
     ans =
      'draw('+myfunc+','+myvar+'='+myfrom+'..'+myto+',title=="'+mytitle+'")';
    }
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <center>
```

```
  Drawing y=f(x)<br/>
  where y is the dependent variable and<br/>
  where x is the independent variable
 </center>
  <table>
   <tr>
    <td>
      What function f would you like to draw?
    </td>
   </tr>
   <tr>
    <td>
     <input type="text" id="function" size="80" tabindex="10"
       value="x*cos(x)"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter independent variable and range:<br/>
     Variable:
     <input type="text" id="var" size="10" tabindex="30" value="x"/>
     ranges from:
     <input type="text" id="range1" size="10" tabindex="40" value="0"/>
     to:
     <input type="text" id="range2" size="10" tabindex="45" value="30"/>
    </td>
   </tr>
   <tr>
    <td>
     Optionally enter a title for your curve:
     <input type="text" id="title1" size="20" tabindex="50"
       value="y=x*cos(x)"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

draw2ddefinedcurve.xhtml

— draw2ddefinedcurve.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc1 = document.getElementById('function1').value;
    myfunc2 = document.getElementById('function2').value;
    myvar = document.getElementById('var').value;
    myfrom = document.getElementById('range1').value;
    myto = document.getElementById('range2').value;
```

```
    mytitle = document.getElementById('title1').value;
    if (mytitle == "") {
     ans=
      'draw(curve('+myfunc1+','+myfunc2+'),'+myvar+'='+myfrom+'..'+myto+')';
    } else {
     ans =
      'draw(curve('+myfunc1+','+myfunc2+'),'+myvar+'='+myfrom+'..'+myto+
            ',title=="'+mytitle+'")';
    }
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <center>
  Drawing a parametrically defined curve<br/>
  (f1(t),f2(t))<br/>
  in terms of two functions f1 and f2<br/>
  and an independent variable t
 </center>
  <table>
   <tr>
    <td>
     Enter the two functions:<br/>
     Function 1:<br/>
     <input type="text" id="function1" size="80" tabindex="10"
       value="-9*sin(4*t/5)"/><br/>
     Function 2:<br/>
     <input type="text" id="function2" size="80" tabindex="20"
       value="8*sin(t)"/><br/>
    </td>
   </tr>
   <tr>
    <td>
    Enter the independent variable and range:<br/>
    Variable:
     <input type="text" id="var" size="10" tabindex="30" value="t"/>
     ranges from:
     <input type="text" id="range1" size="10" tabindex="40" value="-5*%pi"/>
     to:
     <input type="text" id="range2" size="10" tabindex="45" value="5*%pi"/>
    </td>
   </tr>
   <tr>
    <td>
     Optionally enter a title for your curve:
     <input type="text" id="title1" size="20" tabindex="50"
       value="Lissajous"/>
    </td>
   </tr>
```

```
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

————-

draw2dpolynomialequation.xhtml

— draw2dpolynomialequation.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function1').value;
    myvar1 = document.getElementById('var1').value;
    myfrom1 = document.getElementById('range11').value;
    myto1 = document.getElementById('range21').value;
    myvar2 = document.getElementById('var2').value;
    myfrom2 = document.getElementById('range12').value;
    myto2 = document.getElementById('range22').value;
    mytitle = document.getElementById('title1').value;
    if (mytitle == "") {
     ans=
      'draw('+myfunc+'=0,'+myvar1+','+myvar2+',range==['+
            myfrom1+'..'+myto1+','+myfrom2+'..'+myto2+'])';
    } else {
     ans=
      'draw('+myfunc+'=0,'+myvar1+','+myvar2+',range==['+
         myfrom1+'..'+myto1+','+myfrom2+'..'+myto2+'],title=="'+mytitle+'")';
    }
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <center>
  Plotting the solution to p(x,y)=0, where<br/>
  p is a polynomial in two variables x and y
 </center>
  <table>
   <tr>
    <td>
     Enter the polynomial p:<br/>
     <input type="text" id="function1" size="80" tabindex="10"
       value="y^2+7*x*y-(x^3+16*x)"/>
    </td>
   </tr>
   <tr>
```

```
    <td>
    Enter the variables:<br/>
    Variable 1:
     <input type="text" id="var1" size="10" tabindex="30" value="x"/>
     ranges from:
     <input type="text" id="range11" size="10" tabindex="40" value="-15"/>
     to:
     <input type="text" id="range21" size="10" tabindex="45" value="10"/><br/>
    Variable 2:
     <input type="text" id="var2" size="10" tabindex="46" value="y"/>
     ranges from:
     <input type="text" id="range12" size="10" tabindex="47" value="-10"/>
     to:
     <input type="text" id="range22" size="10" tabindex="48" value="50"/><br/>
    </td>
   </tr>
   <tr>
    <td>
     Optionally enter a title for your curve:
     <input type="text" id="title1" size="20" tabindex="50"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

draw3dtwovariable.xhtml

— draw3dtwovariable.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function1').value;
    myvar1 = document.getElementById('var1').value;
    myfrom1 = document.getElementById('range11').value;
    myto1 = document.getElementById('range21').value;
    myvar2 = document.getElementById('var2').value;
    myfrom2 = document.getElementById('range12').value;
    myto2 = document.getElementById('range22').value;
    mytitle = document.getElementById('title1').value;
    if (mytitle == "") {
     ans=
      'draw('+myfunc+','+myvar1+'='+myfrom1+'..'+myto1+','+
                        myvar2+'='+myfrom2+'..'+myto2+')';
    } else {
     ans=
      'draw('+myfunc+','+myvar1+'='+myfrom1+'..'+myto1+','+
                        myvar2+'='+myfrom2+'..'+myto2+
                        ',title=="'+mytitle+'")';
```

```
    }
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <center>
  Drawing z=f(x,y)<br/>
  where z is the dependent variable and<br/>
  where x, y are the independent variables
 </center>
  <table>
   <tr>
    <td>
     What function f which you like to draw?<br/>
     <input type="text" id="function1" size="80" tabindex="10"
       value="exp(cos(x-y)-sin(x*y))-2"/>
    </td>
   </tr>
   <tr>
    <td>
    Enter the independent variables and ranges:<br/>
    Variable 1:
     <input type="text" id="var1" size="10" tabindex="30" value="x"/>
     ranges from:
     <input type="text" id="range11" size="10" tabindex="40" value="-5"/>
     to:
     <input type="text" id="range21" size="10" tabindex="45" value="5"/><br/>
    Variable 2:
     <input type="text" id="var2" size="10" tabindex="46" value="y"/>
     ranges from:
     <input type="text" id="range12" size="10" tabindex="47" value="-5"/>
     to:
     <input type="text" id="range22" size="10" tabindex="48" value="5"/><br/>
    </td>
   </tr>
   <tr>
    <td>
     Optionally enter a title for your curve:
     <input type="text" id="title1" size="20" tabindex="50"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

draw3ddefinedtube.xhtml

— draw3ddefinedtube.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc1 = document.getElementById('function1').value;
    myfunc2 = document.getElementById('function2').value;
    myfunc3 = document.getElementById('function3').value;
    myvar1 = document.getElementById('var1').value;
    myfrom1 = document.getElementById('range1').value;
    myto1 = document.getElementById('range2').value;
    mytitle = document.getElementById('title1').value;
    if (mytitle == "") {
     ans=
      'draw(curve('+myfunc1+','+myfunc2+','+myfunc3+'),'+myvar1+'='+
        myfrom1+'..'+myto1+',tubeRadius==.25,tubePoints==16)';
    } else {
     ans=
      'draw(curve('+myfunc1+','+myfunc2+','+myfunc3+'),'+myvar1+'='+
        myfrom1+'..'+myto1+',tubeRadius==.25,tubePoints==16,title=="'+
        mytitle+'")';
    }
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <center>
  Drawing a parmetrically defined curve: (f1(t), f2(t), f3(t))<br/>
  in terms of three functions f1, f2, and f3<br/>
  and an independent variable t
 </center>
  <table>
   <tr>
    <td>
     Enter the three functions of the independent variable:<br/>
     Function f1:
     <input type="text" id="function1" size="70" tabindex="10"
       value="1.3*cos(2*t)*cos(4*t)+sin(4*t)*cos(t)"/><br/>
     Function f2:
     <input type="text" id="function2" size="70" tabindex="20"
       value="1.3*sin(2*t)*cos(4*t)-sin(4*t)*sin(t)"/><br/>
     Function f3:
     <input type="text" id="function3" size="70" tabindex="30"
       value="2.5*cos(4*t)"/><br/>
    </td>
   </tr>
```

```
   <tr>
    <td>
    Enter the independent variable and range:<br/>
    Variable:
     <input type="text" id="var1" size="10" tabindex="40" value="t"/>
     ranges from:
     <input type="text" id="range1" size="10" tabindex="50" value="0"/>
     to:
     <input type="text" id="range2" size="10" tabindex="60" value="4*%pi"/>
    </td>
   </tr>
   <tr>
    <td>
     Optionally enter a title for your surface:
     <input type="text" id="title1" size="20" tabindex="70" value="knot"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

draw3ddefinedsurface.xhtml

— draw3ddefinedsurface.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc1 = document.getElementById('function1').value;
    myfunc2 = document.getElementById('function2').value;
    myfunc3 = document.getElementById('function3').value;
    myvar1 = document.getElementById('var1').value;
    myfrom1 = document.getElementById('range1').value;
    myto1 = document.getElementById('range2').value;
    myvar2 = document.getElementById('var11').value;
    myfrom2 = document.getElementById('range11').value;
    myto2 = document.getElementById('range21').value;
    mytitle = document.getElementById('title1').value;
    if (mytitle == "") {
     ans=
      'draw(surface('+myfunc1+','+myfunc2+','+myfunc3+'),'+
        myvar1+'='+myfrom1+'..'+myto1+','+
        myvar2+'='+myfrom2+'..'+myto2+')';
    } else {
     ans=
      'draw(surface('+myfunc1+','+myfunc2+','+myfunc3+'),'+
        myvar1+'='+myfrom1+'..'+myto1+','+
        myvar2+'='+myfrom2+'..'+myto2+',title=="'+mytitle+'")';
    }
    alert(ans);
```

```
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <center>
  Drawing a parametrically defined surface<br/>
  (f1(u,v), f2(u,v), f3(u,v))<br/>
  in terms of three functions f1, f2, and f3<br/>
  and two independent variables u and v
 </center>
  <table>
   <tr>
    <td>
     Enter the three functions of the independent variable:<br/>
     Function f1:
     <input type="text" id="function1" size="70" tabindex="10"
       value="u*sin(v)"/><br/>
     Function f2:
     <input type="text" id="function2" size="70" tabindex="20"
       value="v*cos(u)"/><br/>
     Function f3:
     <input type="text" id="function3" size="70" tabindex="30"
       value="u*cos(v)"/><br/>
    </td>
   </tr>
   <tr>
    <td>
    Enter the independent variables and range:<br/>
    Variable 1:
     <input type="text" id="var1" size="10" tabindex="40" value="u"/>
     ranges from:
     <input type="text" id="range1" size="10" tabindex="50" value="-%pi"/>
     to:
     <input type="text" id="range2" size="10" tabindex="60" value="%pi"/>
    <br/>
    Variable 2:
     <input type="text" id="var11" size="10" tabindex="70" value="v"/>
     ranges from:
     <input type="text" id="range11" size="10" tabindex="80" value="-%pi/2"/>
     to:
     <input type="text" id="range21" size="10" tabindex="90" value="%pi/2"/>
    </td>
   </tr>
   <tr>
    <td>
     Optionally enter a title for your surface:
     <input type="text" id="title1" size="20" tabindex="100"/>
    </td>
   </tr>
  </table>
```

```
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———

equdifferential.xhtml

— equdifferential.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Solution of Differential Equations</div>
  <hr/>
In this section we discuss Axiom's facilities for solving differential
equations in closed-form and in series.

Axiom provides facilities for closed-form solution of single differential
equations of the following kinds:
<ul>
 <li>linear ordinary differential equations
 </li>
 <li>non-linear first order ordinary differential equations when integrating
     factors can be found just by integration
 </li>
</ul>

For a discussion of the solution of systems of linear and polynomial
equations, see <a href="axbook/section-8.5.xhtml">Solution of Linear
and Polynomial Equations</a>.
<ul>
 <li>
  <a href="equdifferentiallinear.xhtml">
   Closed-Form Solutions of Linear Differential Equations
  </a>
 </li>
 <li>
  <a href="equdifferentialnonlinear.xhtml">
   Closed-Form Solutions of Non-Linear Differential Equations
  </a>
 </li>
 <li>
  <a href="equdifferentialpowerseries.xhtml">
   Power Series Solutions of Differential Equations
  </a>
 </li>
</ul>
```

```
\getchunk{page foot}
```

equdifferentiallinear.xhtml

— equdifferentiallinear.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">
   Closed-Form Solutions of Linear Differential Equations
  </div>
  <hr/>
A differential equation is an equation involving an unknown function and
one or more of its derivatives. The equation is called ordinary if
derivatives with respect to only one dependent variable appear in the
equation (it is called partial otherwise). The package
<a href="db.xhtml?ElementaryFunctionODESolver">ElementaryFunctionODESolver</a>
provides the top-level operation
<a href="dbopsolve.xhtml">solve</a> for finding closed-form solutions of
ordinary differential equations.

To solve a differential equation, you must first create an operator for the
unknown function. We let y be the unknown function in terms of x.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="y:=operator 'y" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You then type the equation using <a href="dbopd.xhtml">D</a> to create the
derivatives of the unknown function y(x) where x is any symbol you choose
(the so-called dependent variable). This is how you enter the equation
<pre>
    y'' + y' + y = 0
</pre>
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="deq:=D(y x,x,2)+D(y x,x)+y x=0" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
```

```
The simplest way to invoke the <a href="dbopsolve.xhtml">solve</a> command
is with three arguments,
<ul>
 <li>the differential equation</li>
 <li>the operator representing the unknown function</li>
 <li>the dependent variable</li>
</ul>
So, to solve the above equation, we enter this.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="solve(deq,y,x)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Since linear ordinary differential equations have infinitely many solutions,
<a href="dbopsolve.xhtml">solve</a> returns a particular solution f_p and
a basis f1,..fn for the solutions of the corresponding homogeneous equation.
Any expression of the form fp+c1 f1+...+cn fn where the ci do not involve the
dependent variable is also a solution. This is similar to what you get when
you solve systems of linear algebraic equations.

A way to select a unique solution is to specify initial conditions: choose a
value a for the dependent variable and specify the values of the unknown
function and its derivatives at a. If the number of initial conditions is
equal to the order of the equation, then the solution is unique (if it exists
in closed form) and <a href="dbopsolve.xhtml">solve</a> tries to find it. To
specify initial conditions to <a href="dbopsolve.xhtml">solve</a>, use an
<a href="db.xhtml?Equation">Equation</a> of the form x=a for the third
parameter instead of the dependent variable, and add a fourth parameter
consisting of the list of values y(a), y'(a), ...

To find the solution of y''+y=0 satisfying y(0)=y'(0)=1, do this.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="deq:=D(y x,x,2)+y x" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
You can omit the "=0" when you enter the equation to be solved.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p4','p5']);"
    value="solve(deq,y,x=0,[1,1])" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Axiom is not limited to linear differential equations with constant
coefficients. It can also find solutions when the coefficients are
rational or algebraic functions of the dependent variable. Furthermore,
```

```
Axiom is not limited by the order of the equation. Axiom can solve the
following thrid order equations with polynomial coefficients.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p6']);"
    value="deq:=x^3*D(y x,x,3)+x^2*D(y x,x,2)-2*x*D(y x,x)+2*yx=2*x^4" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p6','p7']);"
    value="solve(deq,y,x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
On the other hand, and in contrast with the operation
<a href="dbopintegrate.xhtml">integrate</a> it can happen that Axiom finds
no solution and that some closed-form solution still exists. While it is
mathematically complicated to describe exactly when the solutions are
guaranteed to be found, the following statements are correct and form
good guidelines for linear ordinary differential equations.
<ul>
 <li>If the coefficients are constants, Axiom finds a complete basis of
     solutions (i.e. all solutions).
 </li>
 <li>If the coefficients are rational functions in the dependent variable,
     Axiom ast least finds all solutions that do not involve algebraic
     functions.
 </li>
</ul>
Note that this last statement does not mean that Axiom does not find the
solutions that are algebraic functions. It means that it is not guaranteed
that the algebraic function solutions will be found. This is an example
where all the algebraic solutions are found.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p8']);"
    value="deq:=(x^2+1)*D(y x,x,2)+3*x*D(y x,x)+y x=0" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p8','p9']);"
    value="solve(deq,y,x)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>

\getchunk{page foot}
```

equdifferentialnonlinear.xhtml

— equdifferentialnonlinear.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">
   Closed-Form Solutions of Non-Linear Differential Equations
  </div>
  <hr/>
This is an example that shows how to solve a non-linear first order
ordinary differential equation manually when an integrating factor can
be found just by integration. At the end, we show you how to solve it
directly.

Let's solve the differential equation
<pre>
  y' = y/(x + y log y)
</pre>
Using the notation
<pre>
  m(x,y)+n(x,y)y' = 0
</pre>
we have m=-y and n=x+y*log y
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="m:=-y" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="n:=x+y*log y" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
We first check for exactness, that is, does dm/dy=dn/dx?
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="D(m,y)-D(n,x)" />
  <div id="ansp3"><div></div></div>
```

```
 </li>
</ul>
This is not zero, so the equation is not exact. Therefore we must look
for an integrating factor, that is, a function mu(x,y) such that
d(mu m)/dy=d(mu n)/dx. Normally, we first search for mu(x,y) depending only
on x or only on y. Let’s search for such a mu(x) first.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest(’p4’);"
    value="mu:=operator ’mu" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’,’p5’]);"
    value="a:=D(mu(x)*m,y)-D(mu(x)*n,x)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
If the above is zero for a function mu that does not depend on y, then
mu(x) is an integrating factor.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’,’p5’,’p6’]);"
    value="solve(a=0,mu,x)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The solution depends on y, so there is no integrating factor that depends
on x only. Let’s look for on that depends on y only.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’,’p7’]);"
    value="b:=D(mu(y)*m,y)-D(mu(y)*n,x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’,’p7’,’p8’]);"
    value="sb:=solve(b=0,mu,y)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
We’ve found one. The above mu(y) is an integrating factor. We must multiply
our initial equation (that is, m and n) by the integrating factor.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’,’p7’,’p8’,’p9’]);"
```

```
    value="intFactor:=sb.basis.1" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p2','p4','p7','p8','p9','p10']);"
    value="m:=intFactor*m" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p2','p4','p7','p8','p9','p11']);"
    value="n:=intFactor*n" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
Let's check for exactness.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p2','p4','p7','p8','p9','p10','p11','p12']);"
    value="D(m,y)-D(n,x)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
We must solve the exact equation, that is, find a function s(x,y) such that
ds/dx=m and ds/dy=n. We start by writing
<pre>
  s(x,y) = h(y) + integrate(m,x)
</pre>
where h(y) is an unknown function of y. This guarantees that ds/dx=m.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="h:=operator 'h" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p1','p2','p4','p7','p8','p9','p10','p13','p14']);"
    value="sol:=h y+integrate(m,x)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
All we want is to find h(y) such that ds/dy=n.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick=
     "handleFree(['p1','p2','p4','p7','p8','p9','p10','p13','p14','p15']);"
    value="dsol:=D(sol,y)" />
```

```
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick=
   "handleFree(['p1','p2','p4','p7','p8','p9','p10','p13','p14','p15','p16']);"
    value="nsol:=solve(dsol=n,h,y)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
The above particular solution is the h(y) we want, so we just replace h(y)
by it in the implicit solution.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick=
"handleFree(['p1','p2','p4','p7','p8','p9','p10','p13','p14','p15','p16','p17']);"
    value="eval(sol,h y=nsol.particular)" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
A first integral of the initial equation is obtained by setting this result
equal to an arbitrary constant.

Now that we've seen how to solve the equation "by hand" we show you how to
do it with the <a href="dbopsolve.xhtml">solve</a> operation. First define
y to be an operator.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest('p18');"
    value="y:=operator 'y" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
Next we create the differential equation.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p18','p19']);"
    value="deq:=D(y x,x)=y(x)/(x+y(x)*log y x)" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
Finally, we solve it.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p18','p19','p20']);"
    value="solve(deq,y,x)" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———-

equdifferentialpowerseries.xhtml

— equdifferentialpowerseries.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">
   Power Series Solutions of Differential Equations
  </div>
  <hr/>
The command to solve differential equations in power series around a
particular initial point with specific initial conditions is called
<a href="dbopseriessolve.xhtml">seriesSolve</a>. It can take a variety of
parameters, so we illustrate its use with some examples.

Since the coefficients of some solutions are quite large, we reset the
default to compute only seven terms.
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value=")set streams calculate 7" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can solve a single nonlinear equation of any order. For example, we
solve
<pre>
  y''' = sin(y'') * exp(y) + cos(x)
</pre>
subject to y(0)=1, y'(0)=0, y''(0)=0

We first tell Axiom that the symbol 'y denotes a new operator.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="y:=operator 'y" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Enter the differential equation using y like any system function.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
```

```
    onclick="handleFree(['p1','p2','p3']);"
    value="eq:=D(y(x),x,3)-sin(D(y(x),x,2))*exp(y(x))=cos(x)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Solve it around x=0 with the initial conditions y(0)=1, y'(0)=y''(0)=0.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="seriesSolve(eq,y,x=0,[1,0,0])" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
You can also solve a system of nonlinear first order equations. For
example, we solve a system that has tan(t) and sec(t) as solutions.

We tell Axiom that x is also an operator.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="x:=operator 'x" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Enter the two equations forming our system.
<ul>
 <li> <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p5','p6']);"
    value="eq1:=D(x(t),t)=1+x(t)^2" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p2','p5','p7']);"
    value="eq2:=D(y(t),t)=x(t)*y(t)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Solve the system around t=0 with the initial conditions x(0)=0 and y(0)=1.
Notice that since we give the unknowns in the order [x,y], the answer is a
list of two series in the order [series for x(t), series for y(t)].
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p5','p6','p7','p8']);"
    value="seriesSolve([eq2,eq1],[x,y],t=0,[y(0)=1,x(0)=0])" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
The order in which we give the equations and the initial conditions has no
effect on the order of the solution.
\getchunk{page foot}
```

———-

equationpage.xhtml

— equationpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
Axiom lets you solve equations of various types:
  <table>
   <tr>
    <td>
     <a href="equsystemlinear.xhtml">
      Solution of Systems of Linear Equations
     </a>
    </td>
    <td>
     Solve systems of linear equations
    </td>
   </tr>
   <tr>
    <td>
     <a href="polyroots3.xhtml">
      Solution of a Single Polynomial Equation
     </a>
    </td>
    <td>
     Find roots of polynomials
    </td>
   </tr>
   <tr>
    <td>
     <a href="polyroots4.xhtml">
      Solution of a System of Polynomial Equations
     </a>
    </td>
    <td>
     Solve systems of polynomial equations
    </td>
   </tr>
   <tr>
    <td>
     <a href="equdifferential.xhtml">
      Solution of Differential Equations
     </a>
    </td>
    <td>
     Closed form and series solutions of differential equations
    </td>
   </tr>
  </table>
```

```
\getchunk{page foot}
```

equsystemlinear.xhtml

— equsystemlinear.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Solution of Systems of Linear Equations</div>
  <hr/>
You can use the operation <a href="dbopsolve.xhtml">solve</a> to solve
systems of linear equations.

The operation <a href="dbopsolve.xhtml">solve</a> takes two arguments, the
list of equations and the list of the unknowns to be solved for. A system
of linear equations need not have a unique solution.

To solve the linear system:
<pre>
        x + y + x = 8
    3*x - 2*y + z = 0
    x + 2*y + 2*z = 17
</pre>
evaluate this expression.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="solve([x+y+x=8,3*x-2*y+z=0,x+2*y+2*z=17],[x,y,z])" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Parameters are given as new variables starting with a percent sign and
"%" and the variables are expressed in terms of the parameters. If the system
has no solutions then the empty list is returned.

When you solve the linear system
<pre>
      x + 2*y + 3*z = 2
    2*x + 3*y + 4*z = 2
    3*x + 4*y + 5*z = 2
</pre>
with this expression you get a solution involving a parameter.
<ul>
 <li>
```

```
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="solve([x+2*y+3*z=2,2*x+3*y+4*z=2,3*x+4*y+5*z=2],[x,y,z])" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The system can also be presented as a matrix and a vector. The matrix
contains the coefficients of the linear equations and the vector contains
the numbers appearing on the right-hand sides of the equations. You may
input the matrix as a list of rows and the vector as a list of its elements.

To solve the system:
<pre>
       x + y + z = 8
   2*x - 2*y + z = 0
   x + 2*y + 2*z = 17
</pre>
in matrix form you would evaluate this expression.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="solve([[1,1,1],[3,-2,1],[1,2,2]],[8,0,17])" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The solutions are presented as a Record with two components: the component
particular contains a particular solution of the given system or the item
"failed" if there are no solutions, the component basis contains a list of
vectors that are a basis for the space of solutions of the corresponding
homogeneous system. If the system of linear equations does not have a unique
solution, then the basis component contains non-trivial vectors.

This happens when you solve the linear system
<pre>
    x + 2*y + 3*z = 2
  2*x + 3*y + 4*z = 2
  3*x + 4*y + 5*z = 2
</pre>
with this command.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="solve([[1,2,3],[2,3,4],[3,4,5]],[2,2,2])" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
All solutions of this system are obtained by adding the particular solution
with a linear combination of the basis vectors.

When no solution exists then "failed" is returned as the particular
component, as follows:
<ul>
```

```
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="solve([[1,2,3],[2,3,4],[3,4,5]],[2,3,2])" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
When you want to solve a system of homogeneous equations (that is, a system
where the numbers on the right-hand sides of the equations are all zero)
in the matrix form you can omit the second argument and use the
<a href="dbopnullspace.xhtml">nullSpace</a> operation.

This computes the solutions of the following system of equations:
<pre>
    x + 2*y + 3*z = 0
  2*x + 3*y + 4*z = 0
  3*x + 4*y + 5*z = 0
</pre>
The result is given as a list of vectors and these vectors form a basis for
the solution space.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="nullSpace([[1,2,3],[2,3,4],[3,4,5]])" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>

\getchunk{page foot}
```

examplesexposedpage.xhtml

— examplesexposedpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      examplesexposedpage not implemented
\getchunk{page foot}
```

factored.xhtml

— factored.xhtml —

```
\getchunk{standard head}
 </head>
```

```
 <body>
\getchunk{page head}
factored not implemented
\getchunk{page foot}
```

foundationlibrarydocpage.xhtml

— foundationlibrarydocpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      foundationlibrarydocpage not implemented
\getchunk{page foot}
```

funalgebraicfunctions.xhtml

— funalgebraicfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Algebraic Functions</div>
  <hr/>
Algebraic functions are functions defined by algebraic equations. There are
two ways of constructing them, either by using rational powers or implicitly.
For rational powers, use <a href="dbopstarstar.xhtml">**</a> or the system
functions <a href="dbopsqrt.xhtml">sqrt</a> and
<a href="dbopnthroot.xhtml">nthRoot</a> for square and nth roots.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="f:=sqrt(1+x^(1/3))" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
To define an algebraic function implicitly use
<a href="dboprootof.xhtml">rootOf</a>. The following line defines a function
y of x satisfying the equation
<pre>
  y^3 = x*y-y^2-x^3+1
```

```
</pre>
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="y:=rootOf(y^3+y^2-x*y+x^3-1,y)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
You can manipulate, differentiate or integrate an implicitly defined
algebraic function like any other Axiom function.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="differentiate(y,x)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Higher powers of algebraic functions are automatically reduced during
calculations.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p2','p4']);"
    value="(y+1)^3" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
But denominators are not automatically rationalized.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="g:=inv f" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Use <a href="dbopratdenom.xhtml">ratDenom</a> to remove the algebraic
quantities from the denominator.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p5','p6']);"
    value="ratDenom g" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

funelementaryfunctions.xhtml

— funelementaryfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Elementary Functions</div>
  <hr/>
Axiom has most of the usual functions from calculus built-in.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="f:=x*log y * sin(1/(x+y))" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can substitute values or another elementary function for variables
with the function eval.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="eval(f,[x=y,y=x])" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
As you can see, the substitutions are made "in parallel" as in the case
of polynomials. It's also possible to substitute expressions for kernels
instead of variables.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="eval(f,log y = acosh(x+sqrt y))" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———

funoperatoralgebra.xhtml

— funoperatoralgebra.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Operator</div>
  <hr/>
Given any ring R, the ring of the <a href="db.xhtml?Integer">Integer</a>
linear operators over R is called <a href="db.xhtml?Operator">Operator(R)</a>.
To create an operator over R, first create a basic operator using the
operation <a href="dbopoperator.xhtml">operator</a>, and then convert it
to <a href="db.xhtml?Operator">Operator(R)</a> for the R you want. We choose R
to be the two by two matrices over the integers.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="R:=SQMATRIX(2,INT)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Create the operator tilde on R
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value='t:=operator("tilde")::OP(R)' />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Since  <a href="db.xhtml?Operator">Operator</a> is unexposed we must either
package-call operations from it, or expose it explicitly. For convenience
we will do the latter.
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value=")set expose add constructor Operator" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
To attach an evaluation function (from R to R) to an operator over R, use
evaluate(op,f) where op is an operator over R and f is a function R->R.
This needs to be done only once when the operator is defined. Note that f
must be <a href="db.xhtml?Integer">Integer</a> linear (that is,
<pre>
  f(ax+y) = a f(x) + f(y)
</pre>
for any integer a and any x and y in R). We now attach the transpose map
to the above operator t.
<ul>
```

```
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="evaluate(t,m+->transpose m)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Operators can be manipulated formally as in any ring:
<a href="dbopplus.xhtml">+</a> is the pointwise addition and
<a href="dbopstar.xhtml">*</a> is composition. Any element x of R can
be converted to an operator op_x over R, and the evaluation function of
op_x is left-multiplication by x. Multiplying on the left by this matrix
swaps the two rows.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5']);"
    value="s:R:=matrix [[0,1],[1,0]]" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Can you guess what is the action of the following operator?
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6']);"
    value="rho:=t*s" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Hint: applying rho four times gives the identity, so rho^4-1
should return 0 when applied to any two by two matrix.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7']);"
    value="z:=rho^4-1" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Now check with this matrix
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p8']);"
    value="m:R:=matrix [[1,2],[3,4]]" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p9']);"
    value="z m" />
  <div id="ansp9"><div></div></div>
 </li>
```

```
</ul>
As you have probably guessed by now, rho acts on matrices by rotating
the elements clockwise.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p8’,’p10’]);"
    value="rho m" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p8’,’p11’]);"
    value="rho rho m" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p8’,’p12’]);"
    value="(rho**3) m" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
Do the swapping of rows and transposition commute? We can check by computing
their bracket.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p8’,’p13’]);"
    value="b:=t*s-s*t" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
Now apply it to m.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’,’p8’,’p13’,’p14’]);"
    value="b m" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>

Next we demonstrate how to define a differential operator on a polynomial
ring. This is the recursive definition of the nth Legendre polynomial.
<ul>
 <li>
  <input type="submit" id="p15" class="noresult"
    onclick="makeRequest(’p15’);"
    value="L n==( n=0 => 1 ; n=1 => x ; (2*n-1)/n*x*L(n-1)-(n-1)/n*L(n-2) )" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
Create the differential operator d/dx on polynomials in x over the rational
```

```
numbers.
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p1','p2','p3','p16']);"
    value='dx:=operator("D")::OP(POLY FRAC INT)' />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Now attach a map to it.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p1','p2','p3','p16','p17']);"
    value="evaluate(dx,p+->D(p,'x))" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
This is the differential equation satisfied by the nth Legendre polynomial.
<ul>
 <li>
  <input type="submit" id="p18" class="noresult"
    onclick="handleFree(['p1','p2','p3','p16','p17','p18']);"
    value="E n == (1-x^2)*dx^2-2*x*dx+n*(n+1)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
Now we verify this for n=15. Here is the polynomial.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p1','p2','p3','p15','p19']);"
    value="L 15" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
Here is the operator.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p1','p2','p3','p16','p17','p18','p20']);"
    value="E 15" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
Here is the evaluation.
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick=
 "handleFree(['p1','p2','p3','p15','p16','p17','p18','p19','p20','p21']);"
    value="(E 15)(L 15)" />
  <div id="ansp21"><div></div></div>
 </li>
```

```
</ul>
\getchunk{page foot}
```

———

functionpage.xhtml

— functionpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Functions in Axiom</div>
  <hr/>
In Axiom, a function is an expression in one or more variables.
(Think of it as a function of those variables.) You can also
define a function by rules or use a built-in function. Axiom lets
you convert expressions to compiled functions.
  <table>
   <tr>
    <td>
     <a href="funrationalfunctions.xhtml">Rational Functions</a>
    </td>
    <td>
     Quotients of polynomials
    </td>
   </tr>
   <tr>
    <td>
     <a href="funalgebraicfunctions.xhtml">Algebraic Functions</a>
    </td>
    <td>
     Those defined by polynomial
    </td>
   </tr>
   <tr>
    <td>
     <a href="funelementaryfunctions.xhtml">Elementary Functions</a>
    </td>
    <td>
     The elementary functions of calculus
    </td>
   </tr>
   <tr>
    <td>
     <a href="funsimplification.xhtml">Simplification</a>
    </td>
    <td>
     How to simplify expressions
    </td>
   </tr>
   <tr>
```

```
    <td>
     <a href="funpatternmatching.xhtml">Pattern Matching</a>
    </td>
    <td>
     How to use the pattern matcher
    </td>
   </tr>
   <tr>
    <td>
     <a href="funoperatoralgebra.xhtml">Operator Algebra</a>
    </td>
    <td>
     The operator algebra facility
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

———-

funpatternmatching.xhtml

— funpatternmatching.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Rules and Pattern Matching</div>
  <hr/>
A common mathematical formula is
<pre>
log(x)+log(y)==log(x*y)
</pre>
for any x and y. The presence of the word "any" indicates that x and y
can stand for arbitrary mathematical expressions in the above formula. You
can use such mathematical formulas in Axiom to specify "rewrite rules".
Rewrite rules are objects in Axiom that can be assigned to variables for
later use, often for the purpose of simplification. Rewrite rules look like
ordinary function definitions except that they are preceded by the reserved
word rule. For example, a rewrite rule for the above formula is:
<pre>
  rule log(x) + log(y) == log(x * y)
</pre>
Like function definitions, no action is taken when a rewrite rule is issued.
Think of rewrite rules as functions that take one argument. When a rewrite
rule A=B is applied to an argument f, its meaning is "rewrite every
subexpressions of f that matches A by B". The left-and side of a rewrite rule
is called a <a href="glossarypage.xhtml#p38600">pattern</a>;
```

```
its right-hand side is
called its <a href="glossarypage.xhtml#p49000">substitution</a>.

Create a rewrite rule named logrule. The generated symbol begins with a
"%" and is a place holder for any other terms that might occur in the sum.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="logrule:=rule log(x)+log(y)==log(x*y)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Create an expression with logarithms.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="f:=log sin x + log x" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Apply logrule to f.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="logrule f" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The meaning of our example rewrite rule is "for all expressions x and y,
rewrite log(x) and log(y) by log(x*y)". Patterns generally have both operation
names
(here, <a href="dboplog.xhtml">log</a> and <a href="dbopplus.xhtml">+</a>)
and variables (here, x and y). By default, every operation name stands for
itself. The <a href="dboplog.xhtml">log</a> matches only "log" and not
any other operation such as <a href="dbopsin.xhtml">sin</a>. On the other
hand, variables do not stand for themselves. Rather, a variable denotes a
<a href="glossarypage.xhtml#p39400">pattern variable</a>
that is free to match any expression whatsoever.

When a rewrite rule is applied, a process called
<a href="glossarypage.xhtml#p38661">pattern matching</a>
goes to work by systematically
scanning the subexpressions of the argument. When a subexpression is found
that "matches" the pattern, the subexpression is replaced by the right hand
side of the rule. The details of what happens will be covered later.

The customary Axiom notation for patterns is actually a shorthand for a
longer, more general notation. Pattern variables can be made explicit
by using a percent ("%") as the first character of the variable name. To
say that a name stands for itself, you can prefix that name with a quote
operator ("'"). Although the current Axiom parser does not let you quote
```

```
an operation name, this more general notation gives you an alternative way
of giving the same rewrite rule:
<pre>
  rule log(%x) + log(%y) == log(x*y)
</pre>
This longer notation gives you patterns that the standard notation won't
handle. For example, the rule
<pre>
  rule %f(c * 'x) == c*%f(x)
</pre>
means "for all f and c, replace f(y) by c*f(x) when y is the product
of c and the explicit variable x".

Thus the pattern can have several adornments on the names that appear there.
Normally, all of these adornments are dropped in the substitution on the
right hand side. To summarize:
<hr/>
To enter a single rule in Axiom, use the following syntax:
<pre>
  rule lefthandside == righthandside
</pre>
The lefthandside is a pattern to be matched and the righthandside is its
substitution. The rule is an object of type
<a href="db.xhtml?RewriteRule">RewriteRule</a> that can be assigned to a
variable and applied to expressions to transform them.
<hr/>
Rewrite rules can be collected into rulesets so that a set of rules can be
applied at once. Here is another simplification rule for logarithms.
<pre>
  rule y*log(x) == log(x**y)
</pre>
for any x and y. If instead of giving a single rule following the reserved
word rule you give a "pile" of rules, you create what is called a
ruleset. Like rules, rulesets are objects in Axiom and can be assigned to
variables. You will find it useful to group commonly used rules into
input files, and read them in as needed. Create a ruleset named logrules.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="logrules:=rule (log(x)+log(y)==log(x*y) ; y*log(x)==log(x^y))" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Again, create an expression f containing logarithms.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="f:=a*log(sin x)-2*log x" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Apply the ruleset logrules to f.
```

```
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p4','p5','p6']);"
    value="logrules f" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
We have allowed pattern variables to match arbitrary expressions in the
above examples. Often you want a variable to match onlyh expressions
satisfying some predicate. For example, you may want to apply the
transformation
<pre>
  y*log(x) == log(x^y)
</pre>
only when y is an integer. The way to restrict a pattern variable y by a
predicate f(y) is by using a vertical bar "|", which means "such that",
in much the same way it is used in function definitions. You do this only
once but at the earliest (meaning deepest and leftmost) part of the pattern.
This restricts the logarithmic rule to create integer exponents only.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="logrules2:=rule (log(x)+log(y)==log(x*y) ; (y | integer? y)*log(x)==log(x^y))" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Compare this with the result of applying the previous set of rules.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p5','p8']);"
    value="f" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p5','p7','p9']);"
    value="logrules2 f" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
You should be aware that you might need to apply a function like
<a href="dbopinteger.xhtml">integer</a> within your predicate expression
to actually apply the test function. Here we use
<a href="dbopinteger.xhtml">integer</a> because n has type
<a href="dbexpressioninteger.xhtml">Expression Integer</a> but
<a href="dbopevenq.xhtml">even?</a> is an operation defined on the integers.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="evenRule:=rule cos(x)^(n | integer? n and even? integer n)==(1-sin(x)^2)^(n/2)" />
```

```
  <div id="ansp10"><div></div></div>
 </li>
</ul>
```

Here is the application of the rule.

```
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p10','p11']);"
    value="evenRule(cos(x)^2)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
```

This is an example of some of the usual identities involving products of sines and cosines.

```
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="makeRequest('p12');"
    value="sinCosProducts:=rule (sin(x)*sin(y)==(cos(x-y)-cos(x+y))/2 ; cos(x)*cos(y)==(cos(x-y)+cos(x+y))/2 ; s
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="g:=sin(a)*sin(b)+cos(b)*cos(a)+sin(2*z)*cos(2*a)" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p12','p13','p14']);"
    value="sinCosProducts g" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
```

Another qualification you will often want to use is to allow a pattern to match an identity element. Using the pattern x+y, for example, neither x nor y matches the expression 0. Similarly, if a pattern contains a product x*y or an exponentiation x^y, then neither x nor y matches 1. If identical elements were matched, pattern matching would generally loop. Here is an expansion rule for exponentials.

```
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="makeRequest('p15');"
    value="exprule:=rule exp(a+b)==exp(a)*exp(b)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
```

This rule would cause infinite rewriting on this if either a or b were allowed to match 0.

```
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p15','p16']);"
    value="exprule exp x" />
```

```
  <div id="ansp16"><div></div></div>
 </li>
</ul>
There are occasions when you do want a pattern variable in a sum or product
to match 0 or 1. If so, prefix its name with a "?" whenever it appears in
a left-hand side of a rule. For example, consider the following rule for the
exponential integral
<pre>
  integral((y+exp x)/x,x) == integral(y/x,x)+Ei x
</pre>
for any x and y. This rule is valid if y=0. One solution is to create a
<a href="db.xhtml?Ruleset">Ruleset</a> with two rules, one with and one
without y. A better solution is to use an "optional" pattern variable.
Define rule eirule with a pattern variable ?y to indicate that an
expression may or may not occur.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="makeRequest('p17');"
    value="eirule:=rule integral((?y+exp x)/x,x)==integral(y/x,x)+Ei x" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
Apply rule eirule to an integral without this term.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="handleFree(['p17','p18']);"
    value="eirule integral (exp m/m,m)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
Apply rule eirule to an integral with this term.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p17','p19']);"
    value="eirule integral(sin m+exp m/m,m)" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
Here is one final adornment you will find useful. When matching a pattern
of the form x+y to an expression containing a long sum of the form
a+...+b, there is no way to predict in advance which subset of the sum
matches x and which matches y. Aside from efficiency, this is generally
unimportant since the rule holds for any possible combination of matches
for x and y. In some situations, however, you many want to say which
pattern variable is a sum (or product) of several terms, and which should
match only a single term. To do this, put a prefix colon (":") before the
pattern variable that you want to match mutliple terms. The remaining rules
involve operators u and v.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
```

```
    onclick="makeRequest('p20');"
    value="u:=operator 'u" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
These definitions tell Axiom that u and v are formal operators to be
used in expressions.
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="makeRequest('p21');"
    value="v:=operator 'v" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>
First define myRule with no restrictions on the pattern variables x and y.
<ul>
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="makeRequest('p22');"
    value="myRule:=rule u(x+y)==u x + v y" />
  <div id="ansp22"><div></div></div>
 </li>
</ul>
Apply myRule to an expression.
<ul>
 <li>
  <input type="submit" id="p23" class="subbut"
    onclick="handleFree(['p20','p21','p22','p23']);"
    value="myRule u(a+b+c+d)" />
  <div id="ansp23"><div></div></div>
 </li>
</ul>
Define myOtherRule to match several terms so that the rule gets applied
recursively.
<ul>
 <li>
  <input type="submit" id="p24" class="subbut"
    onclick="makeRequest('p24');"
    value="myOtherRule:=rule u(:x+y)==u x + v y" />
  <div id="ansp24"><div></div></div>
 </li>
</ul>
Apply myOtherRule to the same expression
<ul>
 <li>
  <input type="submit" id="p25" class="subbut"
    onclick="handleFree(['p20','p21','p24','p25']);"
    value="myOtherRule u(a+b+c+d)" />
  <div id="ansp25"><div></div></div>
 </li>
</ul>
Here are some final remarks on pattern matching. Pattern matching provides
a very useful paradigm for solving certain classes of problems, namely,
```

```
those that involve transformations of one form to another and back. However,
it is important to recognize its limitations.

First, pattern matching slows down as the number of rules you have to
apply increases. Thus it is good practice to organize the sets of rules
you use optimally so that irrelevant rules are never included.

Second, careless use of pattern matching can lead to wrong answers. You
should avoid pattern matching to handle hidden algebraic relationships
that can go undetected by other programs. As a simple example, a symbol
such as "J" can easily be used to represent the square root of -1 or some
other important algebraic quantity. Many algorithms branch on whether an
expression is zero or not, then divide by that expression if it is not. If
you fail to simplify an expresison involving powers of J to -1, algorithms
may incorrectly assume an expression is no-zero, take a wrong branch, and
produce a meaningless result.

Pattern matching should also not be used as a substitute for a domain. In
Axiom, objects of one domain are transformed to objects of other domains
using well-defined <a href="dbopcoerce.xhtml">coerce</a> operations.
Pattern matching should be used on objects that are all of the same type.
Thus if your application can be handled by type
<a href="db.xhtml?Expression">Expression</a> in Axiom and you think you
need pattern matching consider this choice carefully. You may well be
better served by extending an existing domain or by building a new domain
of objects for your application.
\getchunk{page foot}
```

funrationalfunctions.xhtml

— funrationalfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Rational Functions</div>
  <hr/>
To create a rational function, just compute the quotient of two
polynomials.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="f:=(x-y)/(x+y)" />
  <div id="ansp1"><div></div></div>
 </li>
```

```
</ul>
Use the functions
<a href="dbopnumer.xhtml">numer</a> and
<a href="dbopdenom.xhtml">denom</a> to recover the numerator and
denominator of a fraction:
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="numer f" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="denom f" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Since these are polynomials, you can apply all of the
<a href="polynomialpage.xhtml">polynomial operations</a> to them.
You can substitute values or other rational functions for the variables
using the function <a href="dbopeval.xhtml">eval</a>. The syntax for
<a href="dbopeval.xhtml">eval</a> is similar to the one for polynomials:
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="eval(f,x=1/x)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="eval(f,[x=y,y=x])" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

funsimplification.xhtml

— funsimplification.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
```

```
\getchunk{page head}
  <div align="center">Simplification</div>
  <hr/>
Simplifying an expression often means different things at different times.
Axiom offers a large number of "simplification" functions. The most common
one, which performs the usual trigonometric simplifications is
<a href="dbopsimplify.xhtml">simplify</a>.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="f:=cos(x)/sin(x)*log(sin(x)^2/(cos(x)^2+sin(x)^2))" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="g:=simplify f" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
If the result of <a href="dbopsimplify.xhtml">simplify</a> is not
satisfactory, specific transformations are available. For example, to
rewrite g in terms of secants and cosecants instead of sines and cosines,
issues:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="h:=sin2csc cos2sec g" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
To apply the logarithm simplification rules to h, issue:
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="h:=expandLog h" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Since the square root of x^2 is the absolute value of x and not x itself,
algebraic radicals are not automatically simplified, but you can
specifically request it by calling
<a href="dboprootsimp.xhtml">rootSimp</a>:
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="f1:=sqrt((x+1)^3)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="rootSimp f1" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
There are other transformations which are sometimes useful. Use the
functions
<a href="dbopcomplexelementary.xhtml">complexElementary</a> and
<a href="dboptrigs.xhtml">trigs</a> to go back and forth between
the complex exponential and trigonometric forms of an elementary function.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="g1:=sin(x+cos x)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p7','p8']);"
    value="g2:=complexElementary g1" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p7','p8','p9']);"
    value="trigs g2" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Similarly, the functions
<a href="dboprealelementary.xhtml">realElementary</a> and
<a href="dbophtrigs.xhtml">htrigs</a> convert hyperbolic functions in
and out of their exponential form.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="h1:=sinh(x+cosh x)" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p10','p11']);"
    value="h2:=realElementary h1" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p10','p11','p12']);"
    value="htrigs h2" />
  <div id="ansp12"><div></div></div>
 </li>
```

```
</ul>
Axiom has other transformations, most of which are in the packages
<a href="db.xhtml?ElementaryFunctionStructurePackage">
ElementaryFunctionStructurePackage</a>,
<a href="db.xhtml?TrigonometricManipulations">
TrigonometricManipulations</a>,
<a href="db.xhtml?AlgebraicManipulations">AlgebraicManipulations</a>, and
<a href="db.xhtml?TranscendentalManipulations">
TranscendentalManipulations</a>. If you need to apply a simplification
rule not built into the system you can use Axiom's
<a href="funpatternmatching.xhtml">pattern matcher</a>.
\getchunk{page foot}
```

glossarypage.xhtml

— glossarypage.xhtml —

```
\getchunk{standard head}
 <style>
  div.glabel    { color:blue; }
  div.gsyntax   { color:blue; }
  div.gspad     { color:blue; }
  div.gfunction { color:blue; }
  div.gtype     { color:blue; }
  div.gcmd      { color:blue; }
 </style>
 </head>
 <body>
\getchunk{page head}
<ul>
 <li><a name="p0" class="glabel"/><b>!</b>
  <div class="gsyntax">(syntax)</div> Suffix character for
  <a href="#p14365">destructive operations</a>.
 </li>
 <li><a name="p74" class="glabel"/><b>,</b>
  <div class="gsyntax">(syntax)</div> a separator for items in a
  <a href="#p50262">tuple</a>,  e.g. to separate arguments of a function
  <div class="gspad">f(x, y)</div>.
 </li>
 <li><a name="p210" class="glabel"/><b>=></b>
  <div class="gsyntax">(syntax)</div> the expression
  <div class="gspad">a => b</div> is equivalent to
  <div class="gspad">if a then</div> <a href="#p19348">exit</a>
  <div class="gspad">b</div>.
 </li>
 <li><a name="p317" class="glabel"/><b>?</b>
  <ol>
   <li>
    <div class="gsyntax">(syntax)</div> a suffix character for
    Boolean-valued <div class="gfunction">function</div> names,
    e.g. <div class="gfunction">odd?</div>.
```

```
  </li>
  <li>
   Suffix character for pattern variables.
  </li>
  <li> The special type <div class="gspad">?</div> means
   <div class="gsyntax">don't care</div>. For example,  the declaration
   <div align="center" class="gspad">x : Polynomial ?</div>  means that
   values assigned to <div class="gspad">x</div> must be polynomials over
   an arbitrary <a href="#p51532">underlying domain</a>.
  </li>
 </ol>
</li>
<li><a name="p725" class="glabel"/><b>abstract datatype</b>
 a programming language principle used in Axiom  where a datatype is
 defined in two parts: (1) a <div class="gsyntax">public</div> part
 describing a set of <a href="#p20171">exports</a>, principally operations
 that apply to objects of that type,  and (2) a
 <div class="gsyntax">private</div> part describing the implementation of
 the datatype usually in terms of a <a href="#p44277">representation</a> for
 objects of the type. Programs which create and otherwise manipulate objects
 of the type may only do so through its exports. The representation and
 other implementation information is specifically hidden.
</li>
<li><a name="p1287" class="glabel"/><b>abstraction</b>
 described functionally or conceptually without regard to implementation
</li>
<li><a name="p1362" class="glabel"/><b>accuracy</b>
 the degree of exactness of an approximation or measurement. In computer
 algebra systems,  computations are typically carried out with complete
 accuracy using integers or rational numbers of indefinite size. Domain
 <div class="gtype">Float</div> provides a function
 <div class="gfunction">precision</div> from
 <div class="gtype">Float</div> to change the precision for floating point
 computations. Computations using <div class="gtype">DoubleFloat</div>
 have a fixed precision but uncertain accuracy.
</li>
<li><a name="p1794" class="glabel"/><b>add-chain</b>
 a hierarchy formed by <a href="#p16819">domain extensions</a>. If domain
 <div class="gspad">A</div> extends domain <div class="gspad">B</div> and
 domain <div class="gspad">B</div> extends domain <div class="gspad">C</div>,
 then <div class="gspad">A</div> has <div class="gsyntax">add-chain</div>
 <div class="gspad">B</div>-<div class="gspad">C</div>.
</li>
<li><a name="p1993" class="glabel"/><b>aggregate</b>
 a data structure designed to hold multiple values. Examples of aggregates
 are <div class="gtype">List</div>,  <div class="gtype">Set</div>,
 <div class="gtype">Matrix</div> and <div class="gtype">Bits</div>.
</li>
<li><a name="p2150" class="glabel"/><b>AKCL</b>
 Austin Kyoto Common LISP,  a version of
 <a href="#p30645"><div class="gspad">KCL</div></a> produced by
 William Schelter, Austin, Texas.
</li>
<li><a name="p2267" class="glabel"/><b>algorithm</b>
```

```
 a step-by-step procedure for a solution of a problem; a program
</li>
<li><a name="p2335" class="glabel"/><b>ancestor</b>
 (of a domain) a category which is a <a href="#p38095">parent</a> of the
 domain,  or a <a href="#p38095">parent</a> of a
 <a href="#p38095">parent</a> and so on.
</li>
<li><a name="p2473" class="glabel"/><b>application</b>
 <div class="gsyntax">(syntax)</div> an expression denoting "application"
 of a function to a set of <a href="#p2885">argument</a> parameters.
 Applications are written as a <a href="#p38004">parameterized form</a>.
 For example,  the form <div class="gspad">f(x, y)</div> indicates the
 "application of the function <div class="gspad">f</div> to the tuple of
 arguments <div class="gspad">x</div> and <div class="gspad">y</div>".
 See also <a href="#p19167">evaluation</a> and
 <a href="#p29675">invocation</a>.
</li>
<li><a name="p2852" class="glabel"/><b>apply</b>
 See <a href="#p2473">application</a>.
</li>
<li><a name="p2885" class="glabel"/><b>argument</b>
 <ol>
  <li>
   (actual argument) a value passed to a function at the time of a
   <a href="#p22911">function</a> call application; also called an
   <div class="gsyntax">actual parameter</div>.
  </li>
  <li>
   (formal argument) a variable used in the definition of a function
   to denote the actual argument passed when the function is called.
  </li>
 </ol>
</li>
<li><a name="p3173" class="glabel"/><b>arity</b>
 <ol>
  <li>
   (function) the number of arguments.
  </li>
  <li>
   (operator or operation) corresponds to the arity of a function
   implementing the operator or operation.
  </li>
 </ol>
</li>
<li><a name="p3322" class="glabel"/><b>assignment</b>
 <div class="gsyntax">(syntax)</div> an expression of the form
 <div class="gspad">x := e</div>,  meaning "assign the value of
 <div class="gspad">e</div> to <div class="gspad">x"</div>. After
 <a href="#p19167">evaluation</a>,  the <a href="#p52894">variable</a>
 <div class="gspad">x</div> <a href="#p39600">pointer</a> to an object
 obtained by evaluating the expression <div class="gspad">e</div>. If
 <div class="gspad">x</div> has a <a href="#p50664">type</a> as a result
 of a previous <a href="#p12903">declaration</a>, the object assigned to
 <div class="gspad">x</div> must have that type. An interpreter must often
```

```
 <a href="#p9572">coercion</a> the value of <div class="gspad">e</div>
 to make that happen. For example,  in the interpreter,
 <div align="center" class="gspad">x : Float := 11</div> first
 <a href="#p12903">declaration</a> <div class="gspad">x</div> to be a float.
 This declaration causes the interpreter to coerce 11 to 11.0 in order to
 assign a floating point value to <div class="gspad">x</div>.
</li>
<li><a name="p4093" class="glabel"/><b>attribute</b>
 a name or functional form denoting <div class="gsyntax">any</div> useful
 computational property. For example,
 <div class="gfunction">commutative(<div class="gspad">"*"</div>)</div>
 asserts that "<div class="gfunction">*</div> is commutative". Also,
 <div class="gfunction">finiteAggregate</div> is used to assert that an
 aggregate has a finite number of immediate components.
</li>
<li><a name="p4380" class="glabel"/><b>basis</b>
 <div class="gsyntax">(algebra)</div> <div class="gspad">S</div> is a
 basis of a module <div class="gspad">M</div> over a
 <a href="#p45405">ring</a> if <div class="gspad">S</div> generates
 <div class="gspad">M</div>,  and <div class="gspad">S</div> is linearly
 independent
</li>
<li><a name="p4536" class="glabel"/><b>benefactor</b>
 (of a given domain) a domain or package that the given domain explicitly
  references (for example,  calls functions from) in its implementation
</li>
<li><a name="p4684" class="glabel"/><b>binary</b>
 operation or function with <a href="#p3173">arity</a> 2
</li>
<li><a name="p4735" class="glabel"/><b>binding</b>
 the association of a variable with properties such as
 <a href="#p52710">value</a> and <a href="#p50664">type</a>. The
 top-level <a href="#p19131">environment</a> in the interpreter consists
 of bindings for all user variables and functions. Every
 <a href="#p22911">function</a> has an associated set of bindings,  one
 for each formal <a href="#p2885">argument</a> and
 <a href="#p32278">local variable</a>.
</li>
<li><a name="p5086" class="glabel"/><b>block</b>
 <div class="gsyntax">(syntax)</div> a control structure where
 expressions are sequentially <a href="#p19167">evaluation</a>.
</li>
<li><a name="p5198" class="glabel"/><b>body</b>
 a <a href="#p23911">function body</a> or <a href="#p33300">loop body</a>.
</li>
<li><a name="p5256" class="glabel"/><b>boolean</b>
 objects denoted by the <a href="#p31774">literals</a>
 <div class="gspad">true</div> and <div class="gspad">false</div>;
 elements of domain <div class="gtype">Boolean</div>.
 See also <div class="gtype">Bits</div>.
</li>
<li><a name="p5399" class="glabel"/><b>built-in function</b>
 a <a href="#p22911">function</a> in the standard Axiom  library.
 Contrast <a href="#p52526">user function</a>.
```

```
</li>
<li><a name="p5499" class="glabel"/><b>cache</b>
 <ol>
  <li>
   (noun) a mechanism for immediate retrieval of previously computed data.
   For example,  a function which does a lengthy computation might store
   its values in a <a href="#p25428">hash table</a> using argument as a
   key. The hash table then serves as a cache for the function (see also
   <div class="gcmd">)set function cache</div>). Also,  when
   <a href="#p43448">recurrence relations</a> which depend upon
   <div class="gspad">n</div> previous values are compiled,  the previous
   <div class="gspad">n</div> values are normally cached
   (use <div class="gcmd">)set functions recurrence</div> to change this).
  </li>
  <li>
   (verb) to save values in a cache.
  </li>
 </ol>
</li>
<li><a name="p6070" class="glabel"/><b>capsule</b>
 the part of the <a href="#p23911">function body</a> of a
 <a href="#p16173">domain constructor</a> that defines the functions
 implemented by the constructor.
</li>
<li><a name="p6220" class="glabel"/><b>case</b>
 <div class="gsyntax">(syntax)</div> an operator used to
 conditionally evaluate code based on the branch of a
 <a href="#p51780">Union</a>. For example,  if value
 <div class="gspad">u</div> is
 <div class="gspad">Union(Integer, "failed")</div>,  the conditional
 expression <div class="gspad">if u case Integer then A else B</div>
 evaluate <div class="gspad">A</div> if <div class="gspad">u</div> is
 an integer and <div class="gspad">B</div> otherwise.
</li>
<li><a name="p6537" class="glabel"/><b>Category</b>
 the distinguished object denoting the type of a category; the class of
 all categories.
</li>
<li><a name="p6628" class="glabel"/><b>category</b>
 <div class="gsyntax">(basic concept)</div> second-order types which
 serve to define useful "classification worlds" for domains, such as
 algebraic constructs (e.g. groups, rings, fields), and data structures
 (e.g. homogeneous aggregates, collections, dictionaries). Examples of
 categories are <div class="gtype">Ring</div> ("the class of all
 rings") and <div class="gtype">Aggregate</div> ("the class of all
 aggregates"). The categories of a given world are arranged in a
 hierarchy (formally, a directed acyclic graph). Each category inherits
 the properties of all its ancestors. Thus, for example, the category
 of ordered rings (<div class="gtype">OrderedRing</div>) inherits the
 properties of the category of rings (<div class="gtype">Ring</div>)
 and those of the ordered sets
 (<div class="gtype">OrderedSet</div>). Categories provide a database of
 algebraic knowledge and ensure mathematical correctness, e.g. that
 "matrices of polynomials" is correct but "polynomials of hash tables"
```

```
 is not, that the multiply operation for "polynomials of continued
 fractions" is commutative, but that for "matrices of power series" is
 not. optionally provide "default definitions" for operations they
 export. Categories are defined in Axiom by functions called
 <a href="#p8355">category constructors</a>. Technically, a category
 designates a class of domains with common
 <a href="#p36041">operations</a> and <a href="#p4093">attributes</a> but
 usually with different <a href="#p22911">functions</a> and
 <a href="#p44277">representations</a> for its constituent
 <a href="#p35301">objects</a>. Categories are always defined using the
 Axiom library language (see also
 <a href="#p8634">category extension</a>).
 See also file <div class="gsyntax">catdef.spad</div>
 for definitions of basic algebraic categories in Axiom .
</li>
<li><a name="p8355" class="glabel"/><b>category constructor</b>
 a function that creates categories, described by an abstract
 datatype in the Axiom programming language. For example, the category
 constructor <div class="gtype">Module</div> is a function which takes
 a domain parameter <div class="gspad">R</div> and creates the category
 "modules over <div class="gspad">R</div>".
</li>
<li><a name="p8634" class="glabel"/><b>category extension</b>
 created by a category definition, an expression usually of the form
 <div class="gspad">A == B with ...</div>. In English, this means
 "category A is a <div class="gspad">B</div> with the new operations
 and attributes as given by ... . See, for example, file
 <div class="gsyntax">catdef.spad</div> for a definitions of the algebra
 categories in Axiom , <div class="gsyntax">aggcat.spad</div> for data
 structure categories.
</li>
<li><a name="p8996" class="glabel"/><b>category hierarchy</b>
 hierarchy formed by category extensions. The root category is
 <div class="gtype">Object</div>. A category can be defined as a
 <a href="#p30459">Join</a> of two or more categories so as to have
 multiple <a href="#p38095">parents</a>. Categories may also have
 parameterized so as to allow conditional inheritance.
</li>
<li><a name="p9278" class="glabel"/><b>character</b>
 <ol>
  <li>
   an element of a character set,  as represented by a keyboard key.
  </li>
  <li>
   a component of a string. For example, the 0th element of the string
   <div class="gspad">"hello there"</div> is the character
   <div class="gsyntax">h</div>.
  </li>
 </ol>
</li>
<li><a name="p9472" class="glabel"/><b>client</b>
 (of a given domain) any domain or package that explicitly calls
 functions from the given domain
</li>
```

```
<li><a name="p9572" class="glabel"/><b>coercion</b>
 an automatic transformation of an object of one
 <a href="#p50664">type</a> to an object of a similar or desired target
 type. In the interpreter, coercions and
 <a href="#p45044">retractions</a> are done automatically by the
 interpreter when a type mismatch occurs. Compare
 <a href="#p12242">conversion</a>.
</li>
<li><a name="p9854" class="glabel"/><b>comment</b>
 textual remarks imbedded in code. Comments are preceded by a double
 dash (<div class="gsyntax">--</div>). For Axiom library code,
 stylized comments for on-line documentation are preceded by a two plus
 signs (<div class="gsyntax">++</div>).
</li>
<li><a name="p10064" class="glabel"/><b>Common LISP</b>
 A version of <a href="#p31518">LISP</a> adopted as an informal
 standard by major users and suppliers of LISP
</li>
<li><a name="p10167" class="glabel"/><b>compile-time</b>
 the time when category or domain constructors are compiled. Contrast
 <a href="#p45818">run-time</a>.
</li>
<li><a name="p10262" class="glabel"/><b>compiler</b>
 a program that generates low-level code from a higher-level source
 language. Axiom has three compilers.
 <ol>
  <li>
   A <div class="gsyntax">graphics
   compiler</div> converts graphical formulas to a compiled subroutine so
   that points can be rapidly produced for graphics commands.
  </li>
  <li>
   An <div class="gsyntax">interpreter compiler</div> optionally compiles
   <a href="#p52526">user functions</a> when first
   <a href="#p29675">invocation</a>
   (use <div class="gcmd">)set functions compile</div>
   to turn this feature on).
  </li>
  <li>
   A <div class="gsyntax">library compiler</div> compiles all
   constructors.
  </li>
 </ol>
</li>
<li><a name="p10792" class="glabel"/><b>computational object</b>
  In Axiom , domains are objects. This term is used to distinquish the
 objects which are members of domains rather than domains themselves.
</li>
<li><a name="p10941" class="glabel"/><b>conditional</b>
 a <a href="#p12001">control structure</a> of the form
 <div class="gspad">if A then B else C</div>; The
 <a href="#p19167">evaluation</a> of <div class="gspad">A</div> produces
 <div class="gspad">true</div> or <div class="gspad">false</div>. If
 <div class="gspad">true</div>, <div class="gspad">B</div> evaluates to
```

```
 produce a value; otherwise <div class="gspad">C</div> evaluates to
 produce a value. When the value is not used,
 <div class="gspad">else C</div> part can be omitted.
</li>
<li><a name="p11264" class="glabel"/><b>constant</b>
 <div class="gsyntax">(syntax)</div> a reserved word used in
 <a href="#p46813">signatures</a> in Axiom programming language to signify
 that mark an operation always returns the same value. For example, the
 signature <div class="gspad">0: constant -> $</div> in the source code
 of <div class="gtype">AbelianMonoid</div> tells the Axiom compiler
 that <div class="gspad">0</div> is a constant so that suitable
 optimizations might be performed.
</li>
<li><a name="p11642" class="glabel"/><b>constructor</b>
 a <a href="#p22911">function</a> which creates a
 <a href="#p6628">category</a>, <a href="#p15041">domain</a>, or
 <a href="#p36778">package</a>.
</li>
<li><a name="p11755" class="glabel"/><b>continuation</b>
 when a line of a program is so long that it must be broken into
 several lines, then all but the first line are called
 <div class="gsyntax">continuation lines</div>. If such a line is given
 interactively, then each incomplete line must end with an underscore.
</li>
<li><a name="p12001" class="glabel"/><b>control structure</b>
 program structures which can specify a departure from normal
 sequential execution. Axiom has four kinds of control structures:
 <a href="#p5086">blocks</a>, <a href="#p6220">case</a> statements,
 <a href="#p10941">conditionals</a>, and <a href="#p33121">loops</a>.
</li>
<li><a name="p12242" class="glabel"/><b>conversion</b>
 the transformation of an object on one <a href="#p50664">type</a> to
 one of another type. Conversions performed automatically are called
 <a href="#p9572">coercions</a>. These happen when the interpreter has a
 type mismatch and a similar or declared target type is needed. In
 general, the user must use the infix operation
 <div class="gspad">::</div> to cause this transformation.
</li>
<li><a name="p12604" class="glabel"/><b>copying semantics</b>
 the programming language semantics used in Pascal but
 <div class="gsyntax">not</div> in Axiom . See also
 <a href="#p39949">pointer semantics</a> for details.
</li>
<li><a name="p12740" class="glabel"/><b>data structure</b>
 a structure for storing data in the computer. Examples are
 <a href="#p31730">lists</a> and <a href="#p25428">hash tables</a>.
</li>
<li><a name="p12850" class="glabel"/><b>datatype</b>
 equivalent to <a href="#p15041">domain</a> in Axiom .
</li>
<li><a name="p12903" class="glabel"/><b>declaration</b>
 <div class="gsyntax">(syntax)</div> an expression of the form
 <div class="gspad">x : T</div> where <div class="gspad">T</div> is some
 <div class="gspad">type</div>. A declaration forces all values
```

```
 <a href="#p3322">assigned</a> to <div class="gspad">T</div> to be of that
 type. If a value is of a different type, the interpreter will try to
 <a href="#p9572">coerce</a> the value to type
 <div class="gspad">T</div>. Declarations are necessary in case of
 ambiguity or when a user wants to introduce an an
 <a href="#p20259">unexposed</a> domain.
</li>
<li><a name="p13351" class="glabel"/><b>default definition</b>
 a function defined by a <a href="#p6628">category</a>. Such
 definitions appear category definitions of the form
 <div class="gspad">C: Category == T add I</div> in an optional
 implmentation part <div class="gspad">I</div> to the right of the
 keyword <div class="gspad">add</div>.
</li>
<li><a name="p13571" class="glabel"/><b>default package</b>
 a optional <a href="#p36778">package</a> of
 <a href="#p22911">functions</a> associated with a category. Such
 functions are necessarily defined in terms over other functions
 exported by the category.
</li>
<li><a name="p13754" class="glabel"/><b>definition</b>
 <div class="gsyntax">(syntax)</div>
 <ol>
  <li>
   An expression of the form
   <div class="gspad">f(a) == b</div> defining function
   <div class="gspad">f</div> with <a href="#p21594">formal arguments</a>
   <div class="gspad">a</div> and <a href="#p5198">body</a>
   <div class="gspad">b</div>; equivalent to the statement
   <div class="gspad">f == (a) +-> b</div>.
  </li>
  <li>
   An expression of the form
   <div class="gspad">a == b</div> where <div class="gspad">a</div> is a
   <a href="#p49347">symbol</a>, equivalent to
   <div class="gspad">a() == b</div>.
   See also <a href="#p33585">macro</a> where a similar
   substitution is done at <a href="#p38242">parse</a> time.
  </li>
 </ol>
</li>
<li><a name="p14178" class="glabel"/><b>delimiter</b>
 a <a href="#p9278">character</a> which marks the beginning or end of
 some syntactically correct unit in the language, e.g. " for strings,
 blanks for identifiers.
</li>
<li><a name="p14365" class="glabel"/><b>destructive operation</b>
 An operation which changes a component or structure of a value. In
 Axiom , all destructive operations have names which end with an
 exclamation mark (<div class="gsyntax">!</div>). For example, domain
 <div class="gtype">List</div> has two operations to reverse the
 elements of a list, one named <div class="gfunction">reverse</div>
 from <div class="gtype">List</div> which returns a copy of the
 original list with the elements reversed, another named
```

```
 <div class="gfunction">reverse!</div> from <div class="gtype">List</div>
 which reverses the elements <div class="gsyntax">in place</div> thus
 destructively changing the original list.
</li>
<li><a name="p14877" class="glabel"/><b>documentation</b>
 <ol>
  <li>
   on-line or hard copy descriptions of Axiom;
  </li>
  <li>
   text in library code preceded by
   <div class="gsyntax">++</div> comments as opposed to general comments
   preceded by <div class="gsyntax">--</div>.
  </li>
 </ol>
</li>
<li><a name="p15041" class="glabel"/><b>domain</b>
 <div class="gsyntax">(basic concept)</div> a domain corresponds to
 the usual notion of abstract datatypes: that of a set of values and a
 set of "exported operations" for the creation and manipulation of
 these values. Datatypes are parameterized, dynamically constructed,
 and can combine with others in any meaningful way, e.g. "lists of
 floats" (<div class="gtype">List Float</div>), "fractions of
 polynomials with integer coefficients"
 (<div class="gtype">Fraction Polynomial Integer</div>),
 "matrices of infinite <a href="#p47825">streams</a> of cardinal numbers"
 (<div class="gtype">Matrix Stream CardinalNumber</div>). The term
 <div class="gsyntax">domain</div> is actually abbreviates
 <div class="gsyntax">domain of computation</div>. Technically, a domain
 denotes a class of objects, a class of
 <a href="#p36041">operations</a> for creating and other manipulating
 these objects, and a class of <a href="#p4093">attributes</a>
 describing computationally useful properties. Domains also provide
 <a href="#p22911">functions</a> for each operation often in terms of some
 <a href="#p44277">representation</a> for the objects. A domain itself
 is an <a href="#p35301">object</a> created by a
 <a href="#p22911">function</a> called a <a href="#p16173">domain
 constructor</a>.
</li>
<li><a name="p16173" class="glabel"/><b>domain constructor</b>
  a function that creates domains, described by an abstract datatype in
 the Axiom programming language. Simple domains like
 <div class="gtype">Integer</div> and <div class="gtype">Boolean</div> are
 created by domain constructors with no arguments. Most domain
 constructors take one or more parameters, one usually denoting an
 <a href="#p51532">underlying domain</a>. For example, the domain
 <div class="gtype">Matrix(R)</div> denotes "matrices over
 <div class="gspad">R"</div>. Domains <div class="gsyntax">Mapping</div>,
 <div class="gsyntax">Record</div>, and
 <div class="gsyntax">Union</div> are primitive domains. All other domains
 are written in the Axiom programming language and can be modified by
 users with access to the library source code.
</li>
<li><a name="p16819" class="glabel"/><b>domain extension</b>
```

```
 a domain constructor <div class="gspad">A</div> is said to
 <div class="gsyntax">extend</div> a domain constructor
 <div class="gspad">B</div> if <div class="gspad">A</div>
 <div class="gspad">'s</div> definition has the form
 <div class="gspad">A == B add ...</div>.
 This intuitively means "functions not defined by <div
 class="gspad">A</div> are assumed to come from
 <div class="gspad">B</div>". Successive domain extensions form
 <a href="#p1794">add-chains</a> affecting the the
 <a href="#p46200">search order</a> for functions not implemented directly
 by the domain during <a href="#p17853">dynamic lookup</a>.
</li>
<li><a name="p17269" class="glabel"/><b>dot notation</b>
 using an infix dot (<div class="gsyntax">.</div>) for function
 application. If <div class="gspad">u</div> is the list
 <div class="gspad">[7, 4, -11]</div> then both
 <div class="gspad">u(2)</div> and <div class="gspad">u.2</div> return
 4. Dot notation nests to left. Thus <div class="gspad">f . g . h</div>
 is equivalent to <div class="gspad">(f . g) . h</div>.
</li>
<li><a name="p17507" class="glabel"/><b>dynamic</b>
 that which is done at <a href="#p45818">run-time</a> as opposed to
 <a href="#p10167">compile-time</a>. For example, the interpreter will
 build the domain "matrices over integers" dynamically in response to
 user input. However, the compilation of all functions for matrices and
 integers is done during <a href="#p10167">compile-time</a>. Constrast
 <a href="#p47594">static</a>.
</li>
<li><a name="p17853" class="glabel"/><b>dynamic lookup</b>
 In Axiom , a <a href="#p17507">domain</a> may or may not explicitly
 provide <a href="#p22911">function</a> definitions for all of its
 exported <a href="#p36041">operations</a>. These definitions may
 instead come from domains in the <a href="#p1794">add-chain</a> or
 from <a href="#p13571">default packages</a>. When a
 <a href="#p2400">function call</a> is made for an operation in the
 domain, up to five steps are carried out.
 <ol>
  <li>
   If the domain itself implements a function for the operation,
   that function is returned.
  </li>
  <li>
   Each of the domains in the <a href="#p1794">add-chain</a> are searched
   for one which implements the function; if found, the function is returned.
  </li>
  <li>
   Each of the <a href="#p13571">default packages</a> for the domain are
   searched in order of the <a href="#p30933">lineage</a>. If any of the
   default packages implements the function, the first one found is
   returned.
  </li>
  <li>
   Each of the <a href="#p13571">default packages</a> for each of the
   domains in the <a href="#p1794">add-chain</a> are searched in the
```

```
   order of their <a href="#p30933">lineage</a>. If any of the default
   packages implements the function, the first one found is returned.
  </li>
  <li> If all of the above steps fail,  an error message is reported.
  </li>
 </ol>
</li>
<li><a name="p19071" class="glabel"/><b>empty</b>
 the unique value of objects with type <div class="gtype">Void</div>.
</li>
<li><a name="p19131" class="glabel"/><b>environment</b>
 a set of <a href="#p4735">bindings</a>.
</li>
<li><a name="p19167" class="glabel"/><b>evaluation</b>
 a systematic process which transforms an
 <a href="#p20659">expression</a> into an object called the
 <a href="#p52710">value</a> of the expression. Evaluation may produce
 <a href="#p46699">side effects</a>.
</li>
<li><a name="p19348" class="glabel"/><b>exit</b>
 <div class="gsyntax">(reserved word)</div> an
 <a href="#p36278">operator</a> which forces an exit from the current
 block. For example, the <a href="#p5086">block</a>
 <div class="gspad">(a := 1; if i > 0 then exit a; a := 2)</div> will
 prematurely exit at the second statement with value 1 if the value of
 <div class="gspad">i</div> is greater than 0. See
 <a href="#p210"><div class="gspad">=></div></a> for an alternate syntax.
</li>
<li><a name="p19681" class="glabel"/><b>explicit export</b>
 <ol>
  <li>
   (of a domain <div class="gspad">D</div>) any
   <a href="#p4093">attribute</a>, <a href="#p36041">operation</a>, or
   <a href="#p6628">category</a> explicitly mentioned in the
   <a href="#p50664">type</a> specification part <div class="gspad">T</div>
   for the domain constructor definition  <div class="gspad">D: T == I</div>
  </li>
  <li>
   (of a category <div class="gspad">C</div>) any
   <a href="#p4093">attribute</a>, <a href="#p36041">operation</a>, or
   <a href="#p6628">category</a> explicitly mentioned in the
   <a href="#p50664">type</a> specification part <div class="gspad">T</div>
   for the domain constructor definition
   <div class="gspad">C: <a href="#p6537">Category</a> == T</div>
  </li>
 </ol>
</li>
<li><a name="p20171" class="glabel"/><b>export</b>
  <a href="#p19681">explicit export</a> or <a href="#p27325">implicit
 export</a> of a domain or category
</li>
<li><a name="p20259" class="glabel"/><b>expose</b>
 some constructors are <div class="gsyntax">exposed</div>, others
 <div class="gsyntax">unexposed</div>. Exposed domains and packages
```

```
 are recognized by the interpreter. Use
 <div class="gcmd">)set expose</div>
 to control change what is exposed. To see both exposed
 and unexposed constructors, use the browser with give the system
 command <div class="gcmd">)set hyperdoc browse exposure
 on</div>. Unexposed constructors will now appear prefixed by star
 (<div class="gspad">*</div>).
</li>
<li><a name="p20659" class="glabel"/><b>expression</b>
 <ol>
  <li> any syntactically correct program fragment.
  </li>
  <li> an element of domain <div class="gtype">Expression</div>
  </li>
 </ol>
</li>
<li><a name="p20757" class="glabel"/><b>extend</b>
 see <a href="#p8634">category extension</a> or <a href="#p16819">domain
 extension</a>
</li>
<li><a name="p20829" class="glabel"/><b>field</b>
 <div class="gsyntax">(algebra)</div> a <a href="#p17507">domain</a>
 which is <a href="#p45405">ring</a> where every non-zero element is
 invertible and where <div class="gspad">xy=yx</div>; a member of
 category <div class="gtype">Field</div>. For a complete list of
 fields, click on <div class="gsyntax">Domains</div> under
 <div class="gsyntax">Cross Reference</div> for
 <div class="gtype">Field</div>.
</li>
<li><a name="p21109" class="glabel"/><b>file</b>
 a program or collection of data stored on disk,  tape or other medium.
</li>
<li><a name="p21186" class="glabel"/><b>float</b>
 a floating-point number with user-specified precision; an element of
 domain <div class="gtype">Float</div>. Floats are
 <a href="#p31774">literals</a> which are written two ways: without an
 exponent (e.g. <div class="gspad">3.1416</div>), or with an exponent
 (e.g. <div class="gspad">3.12E-12</div>). Use function
 <a href="#p42318">precision</a> to change the precision of the mantissage
 (20 digits by default). See also <a href="#p47066">small float</a>.
</li>
<li><a name="p21594" class="glabel"/><b>formal parameter</b>
 (of a function) an identifier <a href="#p4735">bound</a> to the value
 of an actual <a href="#p2885">argument</a> on
 <a href="#p29675">invocation</a>. In the function definition
 <div class="gspad">f(x, y) == u</div>, for example,
 <div class="gspad">x</div> and <div class="gspad">y</div> are the formal
 parameter.
</li>
<li><a name="p21847" class="glabel"/><b>frame</b>
 the basic unit of an interactive session; each frame has its own
 <a href="#p47691">step number</a>, <a href="#p19131">environment</a>, and
 <a href="#p26034">history</a>. In one interactive session, users can
 can create and drop frames, and have several active frames simultaneously.
```

```
</li>
<li><a name="p22113" class="glabel"/><b>free</b>
  <div class="gsyntax">(syntax)</div> A keyword used in user-defined
 functions to declare that a variable is a
 <a href="#p22739">free variable</a> of that function.
 For example, the statement
 <div class="gspad">free x</div> declares the variable
 <div class="gspad">x</div> within the body of a function
 <div class="gspad">f</div> to be a free variable in
 <div class="gspad">f</div>. Without such a declaration, any variable
 <div class="gspad">x</div> which appears on the left hand side of an
 assignment is regarded as a <a href="#p32278">local variable</a> of
 that function. If the intention of the assignment is to give an value
 to a <a href="#p24833">global variable</a> <div class="gspad">x</div>,
 the body of that function must contain the statement
 <div class="gspad">free x</div>.
</li>
<li><a name="p22739" class="glabel"/><b>free variable</b>
  (of a function) a variable which appears in a body of a function but
 is not <a href="#p4735">bound</a> by that function. See
 <a href="#p32278">local variable</a> by default.
</li>
<li><a name="p22911" class="glabel"/><b>function</b>
  implementation of <a href="#p36041">operation</a>; it takes zero or
 more <a href="#p2885">argument</a> parameters and produces zero or
 more values. Functions are objects which can be passed as parameters
 to functions and can be returned as values of functions. Functions can
 also create other functions (see also
 <div class="gtype">InputForm</div>). See also
 <a href="#p2473">application</a> and
 <a href="#p29675">invocation</a>. The terms
 <div class="gsyntax">operation</div> and
 <div class="gsyntax">function</div> are distinct notions in Axiom . An
 operation is an abstraction of a function, described by declaring a
 <a href="#p46813">signature</a>. A function is created by providing an
 implementation of that operation by some piece of Axiom code. Consider
 the example of defining a user-function <div class="gspad">fact</div>
 to compute the <div class="gfunction">factorial</div> of a nonnegative
 integer. The Axiom statement
 <div class="gspad">fact: Integer -> Integer</div>
 describes the operation, whereas the statement
 <div class="gspad">fact(n) = reduce(*, [1..n])</div> defines the
 functions. See also <a href="#p24495">generic function</a>.
</li>
<li><a name="p23911" class="glabel"/><b>function body</b>
  the part of a <a href="#p22911">function</a>
 <div class="gspad">'s</div> definition which is evaluated when the function
 is called at <a href="#p45818">run-time</a>; the part of the function
 definition to the right of the <div class="gspad">==</div>.
</li>
<li><a name="p2400" class="glabel"/><b>function call</b>
  <div class="gsyntax">(syntax)</div> an expression denoting
 "application" of a function to a set of <a href="#p2885">argument</a>
 parameters. Applications are written as a
```

```
 <a href="#p38004">parameterized form</a>. For example, the form
 <div class="gspad">f(x, y)</div> indicates the "application of the function
 <div class="gspad">f</div> to the tuple of arguments
 <div class="gspad">x</div> and <div class="gspad">y</div>". See also
 <a href="#p19167">evaluation</a> and <a href="#p29675">invocation</a>.
</li>
<li><a name="p24123" class="glabel"/><b>garbage collection</b>
  a system function that automatically recycles memory cells from the
 <a href="#p25771">heap</a>. Axiom is built upon
 <a href="#p10064">Common LISP</a> which provides this facility.
</li>
<li><a name="p24294" class="glabel"/><b>garbage collector</b>
 a mechanism for reclaiming storage in the <a href="#p25771">heap</a>.
</li>
<li><a name="p24359" class="glabel"/><b>Gaussian</b>
  a complex-valued expression, e.g. one with both a real and imaginary
 part; a member of a <div class="gtype">Complex</div> domain.
</li>
<li><a name="p24495" class="glabel"/><b>generic function</b>
  the use of one function to operate on objects of different types; One
 might regard Axiom as supporting generic
 <a href="#p36041">operations</a> but not generic functions. One operation
 <div class="gspad">+: (D, D) -> D</div> exists for adding elements in
 a ring; each ring however provides its own type-specific function for
 implementing this operation.
</li>
<li><a name="p24833" class="glabel"/><b>global variable</b>
  A variable which can be referenced freely by functions. In Axiom ,
 all top-level user-defined variables defined during an interactive
 user session are global variables. Axiom does not allow <div
 class="gsyntax">fluid variables</div>, that is, variables
 <a href="#p4735">bound</a> by functions which can be referenced by
 functions those functions call.
</li>
<li><a name="p25189" class="glabel"/><b>Groebner basis</b>
  <div class="gsyntax">(algebra)</div> a special basis for a
 polynomial ideal that allows a simple test for membership. It is
 useful in solving systems of polynomial equations.
</li>
<li><a name="p25348" class="glabel"/><b>group</b>
  <div class="gsyntax">(algebra)</div> a <a href="#p34266">monoid</a>
 where every element has a multiplicative inverse.
</li>
<li><a name="p25428" class="glabel"/><b>hash table</b>
  A data structure that efficiency maps a given object to another. A
 hash table consists of a set of <div class="gsyntax">entries</div>,
 each of which associates a <div class="gsyntax">key</div> with a
 <div class="gsyntax">value</div>. Finding the object stored under a key
 can be very fast even if there are a large number of entries since
 keys are <div class="gsyntax">hashed</div> into numerical codes for
 fast lookup.
</li>
<li><a name="p25771" class="glabel"/><b>heap</b>
  an area of storage used by data in programs. For example, AXIOM will
```

```
 use the heap to hold the partial results of symbolic
 computations. When cancellations occur, these results remain in the
 heap until <a href="#p24294">garbage collected</a>.
</li>
<li><a name="p26034" class="glabel"/><b>history</b>
  a mechanism which records the results for an interactive
 computation. Using the history facility, users can save computations,
 review previous steps of a computation, and restore a previous
 interactive session at some later time. For details, issue the system
 command <div class="gsyntax">)history ?</div> to the interpreter. See
 also <a href="#p21847">frame</a>.
</li>
<li><a name="p26380" class="glabel"/><b>ideal</b>
  <div class="gsyntax">(algebra)</div> a subset of a ring that is
 closed under addition and multiplication by arbitrary ring elements,
 i.e. it<div class="gspad">'s</div> a module over the ring.
</li>
<li><a name="p26553" class="glabel"/><b>identifier</b>
  <div class="gsyntax">(syntax)</div> an Axiom name; a
 <a href="#p31774">literal</a> of type <div class="gtype">Symbol</div>. An
 identifier begins with an alphabetical character or % and may be
 followed by alphabetic characters, digits, ? or !. Certain
 distinquished <a href="#p44698">reserved words</a> are not allowed as
 identifiers but have special meaning in the Axiom .
</li>
<li><a name="p26892" class="glabel"/><b>immutable</b>
  an object is immutable if it cannot be changed by an
 <a href="#p36041">operation</a>; not a <a href="#p34398">mutable
 object</a>. Algebraic objects generally immutable: changing an
 algebraic expression involves copying parts of the original
 object. One exception is a matrix object of type
 <div class="gtype">Matrix</div>. Examples of mutable objects are data
 structures such as those of type <div class="gtype">List</div>. See
 also <a href="#p39949">pointer semantics</a>.
</li>
<li><a name="p27325" class="glabel"/><b>implicit export</b>
  (of a domain or category) any <a href="#p4093">attribute</a> or
 <a href="#p36041">operation</a> which is either an explicit export or
 else an explicit export of some category which an explicit category
 export <a href="#p20757">extends</a>.
</li>
<li><a name="p27564" class="glabel"/><b>index</b>
 <ol>
  <li>
   a variable that counts the number of times a
   <a href="#p33121">loop</a> is repeated.
  </li>
  <li>
   the "address" of an element in a data structure (see also category
   <div class="gtype">LinearAggregate</div>).
  </li>
 </ol>
</li>
<li><a name="p27746" class="glabel"/><b>infix</b>
```

```
  <div class="gsyntax">(syntax)</div> an
 <a href="#p36278">operator</a> placed between two
 <a href="#p35946">operands</a>; also called a
 <div class="gsyntax">binary operator</div>, e.g.
 <div class="gspad">a + b</div>. An infix operator may also be used as a
 <a href="#p42559">prefix</a>, e.g. <div class="gspad">+(a, b)</div> is
 also permissable in the Axiom language. Infix operators have a
 relative <a href="#p42098">precedence</a>.
</li>
<li><a name="p28103" class="glabel"/><b>input area</b>
 a rectangular area on a screen into which users can enter text.
</li>
<li><a name="p28185" class="glabel"/><b>instantiate</b>
 to build a <a href="#p6628">category</a>,  <a href="#p17507">domain</a>,
 or <a href="#p36778">package</a> at run-time
</li>
<li><a name="p28282" class="glabel"/><b>integer</b>
  a <a href="#p31774">literal</a> object of domain
 <div class="gtype">Integer</div>, the class of integers with an unbounded
 number of digits. Integer literals consist of one or more consecutive
 digits (0-9) with no embedded blanks. Underscores can be used to
 separate digits in long integers if desirable.
</li>
<li><a name="p28570" class="glabel"/><b>interactive</b>
 a system where the user interacts with the computer step-by-step
</li>
<li><a name="p28640" class="glabel"/><b>interpreter</b>
  the subsysystem of Axiom responsible for handling user input during
 an interactive session. The following somewhat simplified description
 of the typical action of the interpreter. The interpreter parsers the
 user<div class="gspad">'s</div> input expression to create an
 expression tree then does a bottom-up traversal of the tree. Each
 subtree encountered which is not a value consists of a root node
 denoting an operation name and one or more leaf nodes denoting
 <a href="#p35946">operands</a>. The interpreter resolves type mismatches
 and uses type-inferencing and a library database to determine
 appropriate types of the operands and the result, and an operation to
 be performed. The interpreter then builds a domain to perform the
 indicated operation, then invokes a function from the domain to
 compute a value. The subtree is then replaced by that value and the
 process continues. Once the entire tree has been processed, the value
 replacing the top node of the tree is displayed back to the user as
 the value of the expression.
</li>
<li><a name="p29675" class="glabel"/><b>invocation</b>
  (of a function) the run-time process involved in
 <a href="#p19167">evaluating</a> a <a href="#p22911">function</a>
 <a href="#p2473">application</a>. This process has two steps. First, a
 local <a href="#p19131">environment</a> is created where
 <a href="#p21594">formal arguments</a> are locally
 <a href="#p4735">bound</a> by <a href="#p3322">assignment</a> to their
 respective actual <a href="#p2885">argument</a>. Second, the
 <a href="#p23911">function body</a> is evaluated in that local
 environment. The evaluation of a function is terminated either by
```

```
 completely evaluating the function body or by the evaluation of a
 <div class="gfunction">return</div> expression.
</li>
<li><a name="p30286" class="glabel"/><b>iteration</b>
 repeated evaluation of an expression or a sequence of
 expressions. Iterations use the reserved words
 <div class="gfunction">for</div>, <div class="gfunction">while</div>, and
 <div class="gfunction">repeat</div>.
</li>
<li><a name="p30459" class="glabel"/><b>Join</b>
  a primitive Axiom function taking two or more categories as arguments
 and producing a category containing all of the operations and
 attributes from the respective categories.
</li>
<li><a name="p30645" class="glabel"/><b>KCL</b>
  Kyoto Common LISP, a version of <a href="#p10064">Common LISP</a>
 which features compilation of the compilation of LISP into the
 <div class="gspad">C</div> Programming Language
</li>
<li><a name="p30801" class="glabel"/><b>library</b>
  In Axiom , a coolection of compiled modules respresenting the a
 <a href="#p6628">category</a> or <a href="#p17507">domain</a>
 constructor.
</li>
<li><a name="p30933" class="glabel"/><b>lineage</b>
  the sequence of <a href="#p13571">default packages</a> for a given
 domain to be searched during
 <a href="#p17853">dynamic lookup</a>.
 This sequence is computed first by ordering the category
 <a href="#p2335">ancestors</a> of the domain according to their <div
 class="gsyntax">level number</div>, an integer equal to to the
 minimum distance of the domain from the category. Parents have level
 1, parents of parents have level 2, and so on. Among categories with
 equal level numbers, ones which appear in the left-most branches of
 <div class="gsyntax">Join</div><div class="gspad">s</div> in the
 source code come first. See also <a href="#p17853">dynamic lookup</a>.
</li>
<li><a name="p31518" class="glabel"/><b>LISP</b>
  acronymn for List Processing Language, a language designed for the
 manipulation of nonnumerical data. The Axiom library is translated
 into LISP then compiled into machine code by an underlying LISP.
</li>
<li><a name="p31730" class="glabel"/><b>list</b>
 an object of a <div class="gtype">List</div> domain.
</li>
<li><a name="p31774" class="glabel"/><b>literal</b>
  an object with a special syntax in the language. In Axiom , there are
 five types of literals: <a href="#p5256">booleans</a>,
 <a href="#p28282">integers</a>, <a href="#p21186">floats</a>,
 <a href="#p48077">strings</a>, and <a href="#p49347">symbols</a>.
</li>
<li><a name="p31998" class="glabel"/><b>local</b>
  <div class="gsyntax">(syntax)</div> A keyword used in user-defined
 functions to declare that a variable is a
```

```
 <a href="#p32278">local variable</a> of that function.
 Because of default assumptions on
 variables, such a declaration is not necessary but is available to the
 user for clarity when appropriate.
</li>
<li><a name="p32278" class="glabel"/><b>local variable</b>
  (of a function) a variable <a href="#p4735">bound</a> by that
 function and such that its binding is invisible to any function that
 function calls. Also called a <div class="gsyntax">lexical</div>
 variable. By default in the interpreter:
 <ol>
  <li>
   any variable <div class="gspad">x</div> which appears on the left hand
   side of an assignment is regarded a local variable of that
   function. If the intention of an assignment is to change the value of
   a <a href="#p24833">global variable</a> <div class="gspad">x</div>,
   the body of the function must then contain the statement
   <div class="gspad">free x</div>.
  </li>
  <li>
   any other variable is regarded as a <a href="#p22739">free variable</a>.
  </li>
  <li>
    An optional declaration <div class="gspad">local x</div> is available
   to explicitly declare a variable to be a local variable. All
   <a href="#p21594">formal parameters</a> to the function can be regarded
   as local variables to the function.
  </li>
 </ol>
</li>
<li><a name="p33121" class="glabel"/><b>loop</b>
 <ol>
  <li> an expression containing a <div class="gfunction">repeat</div>
  </li>
  <li>
   a collection expression having a <div class="gfunction">for</div> or a
   <div class="gfunction">while</div>, e.g.
   <div class="gspad">[f(i) for i in S]</div>.
 </li>
 </ol>
</li>
<li><a name="p33300" class="glabel"/><b>loop body</b>
  the part of a loop following the <div class="gfunction">repeat</div>
 that tells what to do each iteration. For example, the body of the
 loop <div class="gspad">for x in S repeat B</div> is
 <div class="gspad">B</div>. For a collection expression, the body of the
 loop precedes the initial <div class="gfunction">for</div> or
 <div class="gfunction">while</div>.
</li>
<li><a name="p33585" class="glabel"/><b>macro</b>
 <ol>
  <li>
   <div class="gsyntax">(syntax)</div> An expression of the form
   <div class="gspad">macro a == b</div> where <div class="gspad">a</div> is a
```

```
   <a href="#p49347">symbol</a> causes <div class="gspad">a</div> to be
   textually replaced by the expression <div class="gspad">b</div> at
   <a href="#p38242">parse</a> time.
  </li>
  <li>
   An expression of the form <div class="gspad">macro f(a) == b</div>
   defines a parameterized macro expansion for a parameterized form
   <div class="gspad">f</div> This macro causes a form
   <div class="gspad">f</div>(<div class="gspad">x</div>) to be textually
   replaced by the expression <div class="gspad">c</div> at parse time,
   where <div class="gspad">c</div> is the expression obtained by
   replacing <div class="gspad">a</div> by <div class="gspad">x</div>
   everywhere in <div class="gspad">b</div>. See also
   <a href="#p13754">definition</a> where a similar substitution is done
   during <a href="#p19167">evaluation</a>.
  </li>
 </ol>
</li>
<li><a name="p34233" class="glabel"/><b>mode</b>
  a type expression containing a question-mark
 (<div class="gsyntax">?</div>). For example, the mode
 <div class="gsyntax">P ?</div> designates <div class="gsyntax">the class
 of all polynomials over an arbitrary ring</div>.
</li>
<li><a name="p34266" class="glabel"/><b>monoid</b>
 is a set with a single, associative operation and an identity element
</li>
<li><a name="p34398" class="glabel"/><b>mutable</b>
  objects which contain <a href="#p39600">pointers</a> to other objects
 and which have operations defined on them which alter these
 pointers. Contrast <a href="#p26892">immutable</a>. Axiom uses
 <a href="#p39949">pointer semantics</a> as does
 <a href="#p31518">LISP</a> in contrast with many other languages such as
 Pascal which use <a href="#p12604">copying semantics</a>. See
 <a href="#p39949">pointer semantics</a> for details.
</li>
<li><a name="p34778" class="glabel"/><b>name</b>
 <ol>
  <li>
   a <a href="#p49347">symbol</a> denoting a <a href="#p52894">variable</a>,
   i.e. the variable <div class="gspad">x</div>.
  </li>
  <li>
   a <a href="#p49347">symbol</a> denoting an
   <a href="#p36041">operation</a>,  i.e. the operation
   <div class="gspad">divide: (Integer, Integer) -> Integer</div>.
  </li>
 </ol>
</li>
<li><a name="p35023" class="glabel"/><b>nullary</b>
 a function with no arguments,
 e.g. <div class="gfunction">characteristic</div>.
</li>
<li><a name="p35104" class="glabel"/><b>nullary</b>
```

```
 operation or function with <a href="#p3173">arity</a> 0
</li>
<li><a name="p35156" class="glabel"/><b>Object</b>
  a category with no operations or attributes,  from which most categories
  in Axiom  are <a href="#p8634">category extensions</a>.
</li>
<li><a name="p35301" class="glabel"/><b>object</b>
  a data entity created or manipulated by programs. Elements of
 domains, functions, and domains themselves are objects. Whereas
 categories are created by functions, they cannot be dynamically
 manipulated in the current system and are thus not considered as
 objects. The most basic objects are <a href="#p31774">literals</a>;
 all other objects must be created
 <a href="#p22911">functions</a>. Objects can refer to other objects using
 <a href="#p39600">pointers</a>. Axiom language uses
 <a href="#p39949">pointer semantics</a> when dealing with
 <a href="#p34398">mutable</a> objects.
</li>
<li><a name="p35854" class="glabel"/><b>object code</b>
  code which can be directly executed by hardware; also known as
 <div class="gsyntax">machine language</div>.
</li>
<li><a name="p35946" class="glabel"/><b>operand</b>
  an argument of an <a href="#p36278">operator</a> (regarding an
 operator as a <a href="#p22911">function</a>).
</li>
<li><a name="p36041" class="glabel"/><b>operation</b>
  an abstraction of a <a href="#p22911">function</a>, described by a
 <a href="#p46813">signature</a>. For example,
 <div align="center" class="gspad">
  fact: NonNegativeInteger -> NonNegativeInteger
 </div>
 describes an operation for "the factorial of a (non-negative) integer".
</li>
<li><a name="p36278" class="glabel"/><b>operator</b>
  special reserved words in the language such as
 <div class="gfunction">+</div> and <div class="gfunction">*</div>;
 operators can be either <a href="#p42559">prefix</a> or
 <a href="#p27746">infix</a> and have a relative
 <a href="#p42098">precedence</a>.
</li>
<li><a name="p36465" glabel="class"/><b>overloading</b>
  the use of the same name to denote distinct functions; a function is
 identified by a <a href="#p46813">signature</a> identifying its name,
 the number and types of its arguments, and its return types. If two
 functions can have identical signatures, a
 <a href="#p37520">package call</a> must be made to distinquish the two.
</li>
<li><a name="p36778" class="glabel"/><b>package</b>
  a domain whose exported operations depend solely on the parameters
 and other explicit domains, e.g. a package for solving systems of
 equations of polynomials over any field, e.g. floats, rational
 numbers, complex rational functions, or power series. Facilities for
 integration, differential equations, solution of linear or polynomial
```

```
 equations, and group theory are provided by "packages". Technically, a
 package is a domain which has no <a href="#p46813">signature</a>
 containing the symbol $. While domains intuitively provide
 computational objects you can compute with, packages intuitively
 provide functions (<a href="#p41450">polymorphic</a> functions) which
 will work over a variety of datatypes.
</li>
<li><a name="p37520" class="glabel"/><b>package call</b>
  <div class="gsyntax">(syntax)</div> an expression of the form
 <div class="gspad">e $ D</div> where <div class="gspad">e</div> is an
 <a href="#p2473">application</a> and <div class="gspad">D</div> denotes
 some <a href="#p36778">package</a> (or <a href="#p17507">domain</a>).
</li>
<li><a name="p37696" class="glabel"/><b>package call</b>
  <div class="gsyntax">(syntax)</div> an expression of the form
 <div class="gspad">f(x, y)$D</div> used to identify that the function
 <div class="gspad">f</div> is to be one from <div class="gspad">D</div>.
</li>
<li><a name="p37833" class="glabel"/><b>package constructor</b>
 same as <a href="#p16173">domain constructor</a>.
</li>
<li><a name="p37878" class="glabel"/><b>parameter</b>
 see <a href="#p2885">argument</a>
</li>
<li><a name="p37908" class="glabel"/><b>parameterized datatype</b>
  a domain that is built on another, for example, polynomials with
 integer coefficients.
</li>
<li><a name="p38004" class="glabel"/><b>parameterized form</b>
  a expression of the form <div class="gspad">f(x, y)</div>, an
 <a href="#p2473">application</a> of a function.
</li>
<li><a name="p38095" class="glabel"/><b>parent</b>
  (of a domain) a category which is explicitly declared in the source
 code definition for the domain to be an <a href="#p20171">export</a>
 of the domain.
</li>
<li><a name="p38242" class="glabel"/><b>parse</b>
 <ol>
  <li>
    (verb) to produce an internal representation of a user input string;
   the resultant internal representation is then "interpreted" by Axiom
   to perform some indicated action.
  </li>
  <li>
   the transformation of a user input string representing a valid Axiom
   expression into an internal representation as a tree-structure.
  </li>
 </ol>
</li>
<li><a name="p38572" class="glabel"/><b>partially ordered set</b>
  a set with a reflexive, transitive and antisymetric
 <a href="#p4684">binary</a> operation.
</li>
```

```
<li><a name="p38600" class="glabel"/><b>pattern</b>
 The left hand side of a rewrite rule is called a pattern. Rewrite rules
 can be used to perform pattern matching, usually for simplification.
 The right hand side of a rule is called the
 <a href="p49000">substitution</a>.
</li>
<li><a name="p38661" class="glabel"/><b>pattern match</b>
 <ol>
  <li>
    (on expressions) Given a expression called a "subject"
   <div class="gspad">u</div>, the attempt to rewrite
   <div class="gspad">u</div> using a set of "rewrite rules". Each rule has
   the form <div class="gspad">A == B</div> where
   <div class="gspad">A</div> indicates a expression called a "pattern" and
   <div class="gspad">B</div> denotes a "replacement". The meaning of
   this rule is "replace <div class="gspad">A</div> by
   <div class="gspad">B"</div>. If a given pattern <div class="gspad">A</div>
   matches a subexpression of <div class="gspad">u</div>, that
   subexpression is replaced by <div class="gspad">B</div>. Once
   rewritten, pattern matching continues until no further changes occur.
  </li>
  <li>
    (on strings) the attempt to match a string indicating a "pattern" to
   another string called a "subject", for example, for the purpose of
   identifying a list of names. In a browser, users may enter
   <a href="#p46294">search strings</a> for the purpose of identifying
   constructors, operations, and attributes.
  </li>
 </ol>
</li>
<li><a name="p39400" class="glabel"/><b>pattern variable</b>
 In a rule a symbol which is not a recognized function acts as a
 pattern variable and is free to match any subexpression.
</li>
<li><a name="p39494" class="glabel"/><b>pile</b>
  alternate syntax for a block, using indentation and column alignment
 (see also <a href="#p5086">block</a>).
</li>
<li><a name="p39600" class="glabel"/><b>pointer</b>
  a reference implemented by a link directed from one object to another
 in the computer memory. An object is said to
 <div class="gsyntax">refer</div> to another if it has a pointer to that
 other object. Objects can also refer to themselves (cyclic references
 are legal). Also more than one object can refer to the same
 object. See also <a href="#p39949">pointer semantics</a>.
</li>
<li><a name="p39949" class="glabel"/><b>pointer semantics</b>
  the programming language semantics used in languages such as LISP
 which allow objects to be <a href="#p34398">mutable</a>. Consider the
 following sequence of Axiom statements:
 <ol>
  <li> <div class="gspad">x : Vector Integer := [1, 4, 7]</div>
  </li>
  <li> <div class="gspad">y := x</div>
```

```
  </li>
  <li> <div class="gspad">swap!(x, 2, 3)</div>
  </li>
 </ol>
 The function <div class="gfunction">swap!</div> from
 <div class="gtype">Vector</div> is used to interchange the 2nd and 3rd
 value in the list <div class="gspad">x</div> producing the value
 <div class="gspad">[1, 7, 4]</div>. What value does
 <div class="gspad">y</div> have after evaluation of the third statement?
 The answer is different in Axiom than it is in a language with
 <a href="#p12604">copying semantics</a>. In Axiom , first the vector
 [1, 2, 3] is created and the variable <div class="gspad">x</div> set to
 <a href="#p39600">point</a> to this object. Let
 <div class="gspad">'s</div> call this object
 <div class="gspad">V</div>. Now <div class="gspad">V</div> refers to its
 <a href="#p26892">immutable</a> components 1, 2, and 3. Next, the
 variable <div class="gspad">y</div> is made to point to
 <div class="gspad">V</div> just as <div class="gspad">x</div> does. Now the
 third statement interchanges the last 2 elements of
 <div class="gspad">V</div> (the <div class="gsyntax">!</div> at the end of
 the name <div class="gfunction">swap!</div> from
 <div class="gtype">Vector</div> tells you that this operation is
 destructive, that is, it changes the elements <div class="gsyntax">in
 place</div>). Both <div class="gspad">x</div> and
 <div class="gspad">y</div> perceive this change to
 <div class="gspad">V</div>. Thus both <div class="gspad">x</div> and
 <div class="gspad">y</div> then have the value
 <div class="gspad">[1, 7, 4]</div>.
 In Pascal, the second statement causes a copy of
 <div class="gspad">V</div> to be stored under
 <div class="gspad">y</div>. Thus the change to <div class="gspad">V</div>
 made by the third statement does not affect
 <div class="gspad">y</div>.
</li>
<li><a name="p41450" class="glabel"/><b>polymorphic</b>
  a <a href="#p22911">function</a> parameterized by one or more
 <a href="#p17507">domains</a>; a <a href="#p2267">algorithm</a> defined
 <a href="#p6628">categorically</a>. Every function defined in a domain
 or package constructor with a domain-valued parameter is
 polymorphic. For example, the same matrix
 <div class="gfunction">*</div> function is used to multiply "matrices over
 integers" as "matrices over matrices over integers"
</li>
<li><a name="p41972" class="glabel"/><b>postfix</b>
  an <a href="#p36278">operator</a> that follows its single
 <a href="#p35946">operand</a>. Postfix operators are not available in
 Axiom.
</li>
<li><a name="p42098" class="glabel"/><b>precedence</b>
  <div class="gsyntax">(syntax)</div> refers to the so-called
 <div class="gsyntax">binding power</div> of an operator. For example,
 <div class="gspad">*</div> has higher binding power than
 <div class="gspad">+</div> so that the expression
 <div class="gspad">a + b * c</div> is equivalent to
```

```
 <div class="gspad">a + (b * c)</div>.
</li>
<li><a name="p42318" class="glabel"/><b>precision</b>
  the number of digits in the specification of a number, e.g. as set by
 <div class="gfunction">precision</div> from <div class="gtype">Float</div>.
</li>
<li><a name="p42440" class="glabel"/><b>predicate</b>
 <ol>
  <li> a Boolean valued function,  e.g.
       <div class="gspad">odd: Integer -> Boolean</div>.
  </li>
  <li> an Boolean valued expression
  </li>
 </ol>
</li>
<li><a name="p42559" class="glabel"/><b>prefix</b>
  <div class="gsyntax">(syntax)</div> an
 <a href="#p36278">operator</a> such as <div class="gspad">-</div> and
 <div class="gspad">not</div> that is written
 <div class="gsyntax">before</div> its single
 <a href="#p35946">operand</a>. Every function of one argument can be used
 as a prefix operator. For example, all of the following have
 equivalent meaning in Axiom : <div class="gspad">f(x)</div>,
 <div class="gspad">f x</div>, and <div class="gspad">f.x</div>. See also
 <a href="#p17269">dot notation</a>.
</li>
<li><a name="p42917" class="glabel"/><b>quote</b>
  the prefix <a href="#p36278">operator</a>
 <div class="gfunction">'</div> meaning <div class="gsyntax">do not
 evaluate</div>.
</li>
<li><a name="p43000" class="glabel"/><b>Record</b>
  (basic domain constructor) a domain constructor used to create a
 inhomogeneous aggregate composed of pairs of "selectors" and
 <a href="#p52710">values</a>. A Record domain is written in the form
 <div class="gspad">Record(a1:D1, ..., an:Dn)</div>
 (<div class="gspad">n</div> > 0) where <div class="gspad">a1</div>, ...,
 <div class="gspad">an</div> are identifiers called the
 <div class="gsyntax">selectors</div> of the record, and
 <div class="gspad">D1</div>, ..., <div class="gspad">Dn</div> are domains
 indicating the type of the component stored under selector
 <div class="gspad">an</div>.
</li>
<li><a name="p43448" class="glabel"/><b>recurrence relation</b>
  A relation which can be expressed as a function
 <div class="gspad">f</div> with some argument <div class="gspad">n</div>
 which depends on the value of <div class="gspad">f</div> at
 <div class="gspad">k</div> previous values. In many cases, Axiom will
 rewrite a recurrence relation on compilation so as to
 <a href="#p5499">cache</a> its previous <div class="gspad">k</div> values
 and therefore make the computation significantly more efficient.
</li>
<li><a name="p43806" class="glabel"/><b>recursion</b>
  use of a self-reference within the body of a function. Indirect
```

```
 recursion is when a function uses a function below it in the call
 chain.
</li>
<li><a name="p43948" class="glabel"/><b>recursive</b>
 <ol>
  <li> A function that calls itself,  either directly or indirectly through
       another function.
  </li>
  <li> self-referential. See also <a href="#p43948">recursive</a>.
  </li>
 </ol>
</li>
<li><a name="p44097" class="glabel"/><b>reference</b>
 see <a href="#p39600">pointer</a>
</li>
<li><a name="p44126" class="glabel"/><b>Rep</b>
  a special identifier used as <a href="#p32278">local variable</a> of
 a domain constructor body to denote the representation domain for
 objects of a domain.
</li>
<li><a name="p44277" class="glabel"/><b>representation</b>
  a <a href="#p17507">domain</a> providing a data structure for
 elements of a domain; generally denoted by the special identifier
 <a href="#p44126">Rep</a> in the Axiom programming language. As domains
 are <a href="#p725">abstract datatypes</a>, this representation is not
 available to users of the domain, only to functions defined in the
 <a href="#p23911">function body</a> for a domain constructor. Any domain
 can be used as a representation.
</li>
<li><a name="p44698" class="glabel"/><b>reserved word</b>
  a special sequence of non-blank characters with special meaning in
 the Axiom language. Examples of reserved words are names such as
 <div class="gfunction">for</div>, <div class="gfunction">if</div>, and
 <div class="gfunction">free</div>, operator names such as
 <div class="gfunction">+</div> and <div class="gspad">mod</div>, special
 character strings such as <div class="gspad">==</div> and
 <div class="gspad">:=</div>.
</li>
<li><a name="p45044" class="glabel"/><b>retraction</b>
  to move an object in a parameterized domain back to the underlying
 domain, for example to move the object <div class="gspad">7</div> from
 a "fraction of integers"
 (domain <div class="gtype">Fraction Integer</div>) to
 "the integers" (domain <div class="gtype">Integer</div>).
</li>
<li><a name="p45280" class="glabel"/><b>return</b>
  when leaving a function, the value of the expression following
 <div class="gfunction">return</div> becomes the value of the function.
</li>
<li><a name="p45405" class="glabel"/><b>ring</b>
  a set with a commutative addition, associative multiplication, a unit
 element, and multiplication distributes over addition and subtraction.
</li>
<li><a name="p45557" class="glabel"/><b>rule</b>
```

```
  <div class="gsyntax">(syntax)</div> 1. An expression of the form
 <div class="gspad">rule A == B</div> indicating a "rewrite
 rule". 2. An expression of the form
 <div class="gspad">rule(R1;...;Rn)</div>
 indicating a set of "rewrite rules"
 <div class="gspad">R1</div>, ..., <div class="gspad">Rn</div>. See
 <a href="#p38661">pattern matching</a> for details.
</li>
<li><a name="p45818" class="glabel"/><b>run-time</b>
  the time of doing a computation. Contrast
 <a href="#p10167">compile-time</a>. rather than prior to it;
 <a href="#p17507">dynamic</a> as opposed to
 <a href="#p47594">static</a>. For example, the decision of the intepreter
 to build a structure such as "matrices with power series entries" in
 response to user input is made at run-time.
</li>
<li><a name="p46129" class="glabel"/><b>run-time check</b>
  an error-checking which can be done only when the program receives
 user input; for example, confirming that a value is in the proper
 range for a computation.
</li>
<li><a name="p46200" class="glabel"/><b>search order</b>
  the sequence of <a href="#p13571">default packages</a> for a given
 domain to be searched during <a href="#p17853">dynamic
 lookup</a>. This sequence is computed first by ordering the category
 <a href="#p2335">ancestors</a> of the domain according to their
 <div class="gsyntax">level number</div>, an integer equal to to the
 minimum distance of the domain from the category. Parents have level
 1, parents of parents have level 2, and so on. Among categories with
 equal level numbers, ones which appear in the left-most branches of
 <div class="gsyntax">Join</div><div class="gspad">s</div> in the
 source code come first. See also <a href="#p17853">dynamic lookup</a>.
</li>
<li><a name="p46294" class="glabel"/><b>search string</b>
 a string entered into an <a href="#p28103">input area</a> on a screen
</li>
<li><a name="p46372" class="glabel"/><b>selector</b>
 an identifier used to address a component value of a
 <a href="p43000">Record</a> datatype.
</li>
<li><a name="p46454" class="glabel"/><b>semantics</b>
  the relationships between symbols and their meanings. The rules for
 obtaining the <div class="gsyntax">meaning</div> of any syntactically
 valid expression.
</li>
<li><a name="p46594" class="glabel"/><b>semigroup</b>
  <div class="gsyntax">(algebra)</div> a <a href="#p34266">monoid</a>
 which need not have an identity; it is closed and associative.
</li>
<li><a name="p46699" class="glabel"/><b>side effect</b>
  action which changes a component or structure of a value. See
 <a href="#p14365">destructive operation</a> for details.
</li>
<li><a name="p46813" class="glabel"/><b>signature</b>
```

```
  <div class="gsyntax">(syntax)</div> an expression describing an
 <a href="#p36041">operation</a>. A signature has the form as
 <div class="gspad">name : source -> target</div>, where
 <div class="gspad">source</div> gives the type of the arguments of the
 operation, and <div class="gspad">target</div> gives the type of the
 result.
</li>
<li><a name="p47066" class="glabel"/><b>small float</b>
 the domain for hardware floating point arithmetic as provided by the
 computer hardware.
</li>
<li><a name="p47159" class="glabel"/><b>small integer</b>
 the domain for hardware integer arithmetic. as provided by the computer
 hardware.
</li>
<li><a name="p47246" class="glabel"/><b>source</b>
  the <a href="#p50664">type</a> of the argument of a
 <a href="#p22911">function</a>; the type expression before the
 <div class="gspad">-></div> in a <a href="#p46813">signature</a>. For
 example, the source of
 <div class="gspad">f : (Integer, Integer) -> Integer</div>
 is <div class="gspad">(Integer, Integer)</div>.
</li>
<li><a name="p47486" class="glabel"/><b>sparse</b>
  data structure whose elements are mostly identical (a sparse matrix
 is one filled with mostly zeroes).
</li>
<li><a name="p47594" class="glabel"/><b>static</b>
 that computation done before run-time, such as compilation. Contrast
 <a href="#p17507">dynamic</a>.
</li>
<li><a name="p47691" class="glabel"/><b>step number</b>
  the number which precedes user input lines in an interactive session;
 the output of user results is also labeled by this number.
</li>
<li><a name="p47825" class="glabel"/><b>stream</b>
  an object of <div class="gtype">Stream(R)</div>, a generalization of
 a <a href="#p31730">list</a> to allow an infinite number of
 elements. Elements of a stream are computed "on demand". Strings are
 used to implement various forms of power series.
</li>
<li><a name="p48077" class="glabel"/><b>string</b>
  an object of domain <div class="gtype">String</div>. Strings are
 <a href="#p31774">literals</a> consisting of an arbitrary sequence of
 <a href="#p9278">characters</a> surrounded by double-quotes
 (<div class="gfunction">"</div>), e.g.
 <div class="gspad">"Look here!"</div>.
</li>
<li><a name="p48303" class="glabel"/><b>subdomain</b>
  <div class="gsyntax">(basic concept)</div> a
 <a href="#p17507">domain</a> together with a
 <a href="#p42440">predicate</a> characterizing which members of the
 domain belong to the subdomain. The exports of a subdomain are usually
 distinct from the domain itself. A fundamental assumption however is
```

```
 that values in the subdomain are automatically
 <a href="#p9572">coerceable</a> to values in the domain. For example, if
 <div class="gspad">n</div> and <div class="gspad">m</div> are declared
 to be members of a subdomain of the integers, then
 <div class="gsyntax">any</div> <a href="#p4684">binary</a> operation from
 <div class="gtype">Integer</div> is available on
 <div class="gspad">n</div> and <div class="gspad">m</div>. On the other
 hand, if the result of that operation is to be assigned to, say,
 <div class="gspad">k</div>, also declared to be of that subdomain, a
 <a href="#p45818">run-time</a> check is generally necessary to ensure
 that the result belongs to the subdomain.
</li>
<li><a name="p49000" class="glabel"/><b>substitution</b>
 The right hand side of a rule is called the substitution.
 The left hand side of a rewrite rule is called a
 <a href="p38600">pattern</a>. Rewrite rules
 can be used to perform pattern matching, usually for simplification.
</li>
<li><a name="p49128" class="glabel"/><b>such that clause</b>
  the use of <div class="gfunction">|</div> followed by an expression
 to filter an iteration.
</li>
<li><a name="p49209" class="glabel"/><b>suffix</b>
  <div class="gsyntax">(syntax)</div> an
 <a href="#p36278">operator</a> which placed after its operand. Suffix
 operators are not allowed in the Axiom language.
</li>
<li><a name="p49347" class="glabel"/><b>symbol</b>
  objects denoted by <a href="#p26553">identifier</a>
 <a href="#p31774">literals</a>; an element of domain
 <div class="gtype">Symbol</div>. The interpreter defaultly converts a
 symbol <div class="gspad">x</div> into
 <div class="gtype">Variable(x)</div>.
</li>
<li><a name="p49538" class="glabel"/><b>syntax</b>
 rules of grammar,  punctuation etc. for forming correct expressions.
</li>
<li><a name="p49613" class="glabel"/><b>system commands</b>
  top-level Axiom statements that begin with
 <div class="gsyntax">)</div>. System commands allow users to query the
 database, read files, trace functions, and so on.
</li>
<li><a name="p49773" class="glabel"/><b>tag</b>
 an identifier used to discriminate a branch of a
 <a href="#p51780">Union</a> type.
</li>
<li><a name="p49851" class="glabel"/><b>target</b>
  the <a href="#p50664">type</a> of the result of a
 <a href="#p22911">function</a>; the type expression following the
 <div class="gspad">-></div> in a <a href="#p46813">signature</a>.
</li>
<li><a name="p49990" class="glabel"/><b>top-level</b>
 refers to direct user interactions with the Axiom  interpreter.
</li>
```

```
<li><a name="p50064" class="glabel"/><b>totally ordered set</b>
  <div class="gsyntax">(algebra)</div> a partially ordered set where
 any two elements are comparable.
</li>
<li><a name="p50148" class="glabel"/><b>trace</b>
  use of system function <div class="gcmd">)trace</div> to track the
 arguments passed to a function and the values returned.
</li>
<li><a name="p50262" class="glabel"/><b>tuple</b>
  an expression of two or more other expressions separated by commas,
 e.g. <div class="gspad">4, 7, 11</div>. Tuples are also used for
 multiple arguments both for <a href="#p2473">applications</a>
 (e.g. <div class="gspad">f(x, y)</div>) and in
 <a href="#p46813">signatures</a> (e.g.
 <div class="gspad">(Integer, Integer) -> Integer</div>).
 A tuple is not a data structure, rather a
 syntax mechanism for grouping expressions.
</li>
<li><a name="p50664" class="glabel"/><b>type</b>
  The type of any <a href="#p48303">subdomain</a> is the unique symbol
 <div class="gsyntax">Category</div>. The type of a
 <a href="#p17507">domain</a> is any <a href="#p6628">category</a> that
 domain belongs to. The type of any other object is either the (unique)
 domain that object belongs to or any <a href="#p48303">subdomain</a>
 of that domain. The type of objects is in general not unique.
</li>
<li><a name="p51002" class="glabel"/><b>type checking</b>
 a system function which determines whether the datatype of an object is
 appropriate for a given operation.
</li>
<li><a name="p51114" class="glabel"/><b>type constructor</b>
 a <a href="#p16173">domain constructor</a> or
 <a href="#p8355">category constructor</a>.
</li>
<li><a name="p51189" class="glabel"/><b>type inference</b>
  when the interpreter chooses the type for an object based on
 context. For example, if the user interactively issues the definition
 <div align="center" class="gspad">f(x) == (x + %i)**2</div> then
 issues <div class="gspad">f(2)</div>, the interpreter will infer the
 type of <div class="gspad">f</div> to be
 <div class="gspad">Integer -> Complex Integer</div>.
</li>
<li><a name="p51480" class="glabel"/><b>unary</b>
 operation or function with <a href="#p3173">arity</a> 1
</li>
<li><a name="p51532" class="glabel"/><b>underlying domain</b>
  for a <a href="#p17507">domain</a> that has a single domain-valued
 parameter, the <div class="gsyntax">underlying domain</div> refers to
 that parameter. For example, the domain "matrices of integers"
 (<div class="gtype">Matrix Integer</div>) has underlying domain
 <div class="gtype">Integer</div>.
</li>
<li><a name="p51780" class="glabel"/><b>Union</b>
  <div class="gsyntax">(basic domain constructor)</div> a domain
```

```
 constructor used to combine any set of domains into a single domain. A
 Union domain is written in the form
 <div class="gspad">Union(a1:D1,..., an:Dn)</div>
 (<div class="gspad">n</div> > 0) where
 <div class="gspad">a1</div>, ..., <div class="gspad">an</div> are
 identifiers called the <div class="gsyntax">tags</div> of the union,
 and <div class="gspad">D1</div>, ..., <div class="gspad">Dn</div> are
 domains called the <div class="gsyntax">branches</div> of the
 union. The tags <div class="gspad">ai</div> are optional, but required
 when two of the <div class="gspad">Di</div> are equal, e.g.
 <div class="gspad">Union(inches:Integer, centimeters:Integer)</div>. In the
 interpreter, values of union domains are automatically coerced to
 values in the branches and vice-versa as appropriate. See also
 <a href="#p6220">case</a>.
</li>
<li><a name="p52482" class="glabel"/><b>unit</b>
 <div class="gsyntax">(algebra)</div> an invertible element.
</li>
<li><a name="p52526" class="glabel"/><b>user function</b>
 a function defined by a user during an interactive session. Contrast
 <a href="#p5399">built-in function</a>.
</li>
<li><a name="p52631" class="glabel"/><b>user variable</b>
 a variable created by the user at top-level during an interactive session
</li>
<li><a name="p52710" class="glabel"/><b>value</b>
 <ol>
  <li>
   the result of <a href="#p19167">evaluating</a> an expression.
  </li>
  <li>
   a property associated with a <a href="#p52894">variable</a> in a
   <a href="#p4735">binding</a> in an <a href="#p19131">environment</a>.
  </li>
 </ol>
</li>
<li><a name="p52894" class="glabel"/><b>variable</b>
 a means of referring to an object but itself is not an object. A
 variable has a name and an associated <a href="#p4735">binding</a>
 created by <a href="#p19167">evaluation</a> of Axiom expressions such
 as <a href="#p12903">declarations</a>,
 <a href="#p3322">assignments</a>, and
 <a href="#p13754">definitions</a>. In the top-level
 <a href="#p19131">environment</a> of the interpreter, variables are
 <a href="#p24833">global variables</a>. Such variables can be freely
 referenced in user-defined functions although a
 <a href="#p22113">free</a> declaration is needed to assign values to
 them. See <a href="#p32278">local variable</a> for details.
</li>
<li><a name="p53484" class="glabel"/><b>Void</b>
  the type given when the <a href="#p52710">value</a> and <a
 href="#p50664">type</a> of an expression are not needed. Also used
 when there is no guarantee at run-time that a value and predictable
 mode will result.
```

```
 </li>
 <li><a name="p53681" class="glabel"/><b>wild card</b>
   a symbol which matches any substring including the empty string; for
  example, the search string <div class="gsyntax">*an*</div> matches an
  word containing the consecutive letters <div class="gsyntax">a</div>
  and <div class="gsyntax">n</div>
 </li>
 <li><a name="p53866" class="glabel"/><b>workspace</b>
   an interactive record of the user input and output held in an
  interactive history file. Each user input and corresponding output
  expression in the workspace has a corresponding <a href="#p47691">step
  number</a>. The current output expression in the workspace is referred
  to as <div class="gspad">%</div>. The output expression associated
  with step number <div class="gspad">n</div> is referred to by <div
  class="gspad">%%(n)</div>. The <div class="gspad">k</div>-th previous
  output expression relative to the current step number <div
  class="gspad">n</div> is referred to by <div class="gspad">%%(-
  k)</div>. Each interactive <a href="#p21847">frame</a> has its own
  workspace.
 </li>
</ul>
\getchunk{page foot}
```

graphexamples.xhtml

— graphexamples.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Graphics Examples</div>
  <hr/>
Here are some examples of Axiom graphics.
<ul>
 <li> <a href="graphexamplesassorted.xhtml">Assorted Examples</a><br/>
   Examples of each type of Axiom Graphics
 </li>
 <li> <a href="graphexamplesthreed.xhtml">Three Dimensional Graphics</a><br/>
   Plot parametrically defined surfaces of three functions.
 </li>
 <li> <a href="graphexamplesonevariable.xhtml">
   Functions of One Variable</a><br/>
   Plot curves defined by an equation y=f(x)
 </li>
 <li> <a href="graphexamplesparametric.xhtml">Parametric Curves</a><br/>
   Plot curves defined by parametric equations x=f(t), y=f(t)
 </li>
 <li> <a href="graphexamplespolar.xhtml">Polar Coordinates</a><br/>
   Plot curves given in polar form by an equation r=f(&#x003B8;)
 </li>
```

```
 <li> <a href="graphexamplesimplicit.xhtml">Implicit Curves</a><br/>
   Plot non-singular curves defined by a polynomial equation
 </li>
 <li> <a href="graphexampleslistofpoints.xhtml">Lists of Points</a><br/>
   Plot lists of points in the (x,y)-plane
 </li>
</ul>
\getchunk{page foot}
```

graphexamplesassorted.xhtml

— graphexamplesassorted.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Assorted Graphics Examples</div>
  <hr/>
Function of two variables: z=f(x,y)
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(sin(x*y),x=-2.5..2.5,y=-2.5..2.5)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Function of one variable: y=f(x)
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="draw(sin tan x - tan sin x,x=0..6)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Plane parametric curve: x=f(t),y=g(t)
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="draw(curve(sin(t)*sin(2*t),sin(3*t)*sin(4*t)),t=0..2*%pi)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Space parametric curve: x=f(t),y=g(t),z=h(t)
```

```
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="draw(curve(sin(t)*sin(2*t),sin(3*t)*sin(4*t),sin(5*t)*sin(6*t)),t=0..2*%pi)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Polar coordinates: r=f(&#x003B8;)
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="draw(sin(17*t),t=0..2*%pi,coordinates==polar)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Implicit curves: p(x,y)=0
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="draw(y^2+y=x^3-x,x,y,range==[-2..2,-2..1])" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

graphexamplesimplicit.xhtml

— graphexamplesimplicit.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Implicit Curves</div>
  <hr/>
A Conic Section (Hyperbola)
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(x*y=1,x,y,range==[-3..3,-3..3])" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
```

```
An Elliptic Curve
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="draw(y^2+y=x^3-x,x,y,range==[-2..2,-2..1])" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Cartesian Ovals
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="p:=((x^2+y^2+1)-8*x)^2-(8*(x^2+y^2+1)-4*x-1)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value='draw(p=0,x,y,range==[-1..11,-7..7],title=="Cartesian Ovals")' />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Cassinian Ovals: two loops
<ul>
 <li>
  <input type="submit" id="p5" class="noresult"
    onclick="makeRequest('p5');"
    value="q:=(x^2+y^2+7^2)^2-(6^4+4*7^2*x^2)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p5','p6']);"
    value='draw(q=0,x,y,range==[-10..10,-4..4],title=="Cassinian Oval")' />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———-

graphexampleslistofpoints.xhtml

— graphexampleslistofpoints.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
```

```
 <body>
\getchunk{page head}
  <div align="center">Plotting Lists of Points</div>
  <hr/>
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value="p:=map(point,[[1.,1.],[0.,1.],[0.,0.],[1.,0.],[1.,.5],[.5,0.],[0.,.5],[.5,1.],[.25,.25],[.25,.75],[.7
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="noresult"
    onclick="makeRequest('p2');"
    value="llp:=[[p.1,p.2],[p.2,p.3],[p.3,p.4],[p.4,p.1],[p.5,p.6],[p.6,p.7],[p.7,p.8],[p.8,p.5],[p.9,p.10],[p.1
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="lsize:=[6,6,6,6,8,8,8,8,10,10,10,10]" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="noresult"
    onclick="makeRequest('p4');"
    value="pc1:=pastel red()" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="noresult"
    onclick="makeRequest('p5');"
    value="pc2:=dim green()" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="noresult"
    onclick="makeRequest('p6');"
    value="pc3:=pastel yellow()" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="noresult"
    onclick="makeRequest('p7');"
    value="lpc:=[pc1,pc1,pc1,pc1,pc2,pc2,pc2,pc2,pc3,pc3,pc3,pc3]" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="noresult"
    onclick="makeRequest('p8');"
    value="lc:=[pastel blue(), light yellow(), dim green(), bright red(), light green(), dim yellow(), bright bl
  <div id="ansp8"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p9" class="noresult"
    onclick="makeRequest('p9');"
    value="g:=makeGraphImage(llp,lpc,lc,lsize)$GRIMAGE" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p9','p10']);"
    value='makeViewport2D(g,[title("Lines")])$VIEW2D' />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
The <a href="dbopmakeviewport2d.xhtml">makeViewport2D</a> command takes a
list of options as a parameter in this example. The string "Lines" is
designated as the viewport's title.
\getchunk{page foot}
```

graphexamplesonevariable.xhtml

— graphexamplesonevariable.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Functions of One Variable</div>
  <hr/>
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(sin tan x - tan sin x, x=0..6)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="draw(sin x + cos x, x=0..2*%pi)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="draw(sin(1/x),x=-1..1)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="draw(x*sin(1/x),x=-1..1)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

graphexamplesparametric.xhtml

— graphexamplesparametric.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Parametric Curves</div>
  <hr/>
The Lemnicate of Bernoulli
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(curve(cos(t/(1+sin(t)^2)),sin(t)*cos(t)/(1+sin(t)^2)),t=-%pi..%pi)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Lissajous curve
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="draw(curve(9*sin(3*t/4),8*sin(t)),t=-4*%pi..4*%pi)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
A gnarly closed curve
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="draw(curve(sin(t)*sin(2*t)*sin(3*t),sin(4*t)*sin(5*t)*sin(6*t)),t=0..2*%pi)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Another closed curve
<ul>
```

```
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="draw(curve(cos(4*t)*cos(7*t),cos(4*t)*sin(7*t)),t=0..2*%pi)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

graphexamplespolar.xhtml

— graphexamplespolar.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Polar Coordinates</div>
  <hr/>
A circle
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(1,t=0..2*%pi,coordinates==polar)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
A spiral
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="draw(t,t=0..100,coordinates==polar)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
A Petal Curve
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="draw(sin(4*t),t=0..2*%pi,coordinates==polar)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
A Limacon
```

```
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="draw(2+3*sin(t),t=0..2*%pi,coordinates==polar)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

graphexamplesthreed.xhtml

— graphexamplesthreed.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Three Dimensional Graphics</div>
  <hr/>
Plots of parametric surfaces defined by functions f(u,v), g(u,v) and h(u,v).

Pear Surface
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value='draw(surface((1+exp(-100*u*u))*sin(%pi*u)*sin(%pi*v),(1+exp(-100*u*u))*sin(%pi*u)*cos(%pi*v),(1+exp(-
  <div id="ansp1"><div></div></div>
 </li>
</ul>

Trigonometric Screw
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value='draw(surface(x*cos(y),x*sin(y),y*cos(x)),x=-4..4,y=0..2*%pi,var1Steps==40,var2Steps==40,title=="Trigo
  <div id="ansp2"><div></div></div>
 </li>
</ul>

Etruscan Venus
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
```

```
    value="a:=1.3*cos(2*x)*cos(y)+sin(y)*cos(x)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="noresult"
    onclick="makeRequest('p4');"
    value="b:=1.3*sin(2*x)*cos(y)-sin(y)*sin(x)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="noresult"
    onclick="makeRequest('p5');"
    value="c:=2.5*cos(y)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p3','p4','p5','p6']);"
    value='draw(surface(a,b,c),x=0..%pi,y=-%pi..%pi,var1Steps==40,var2Steps==40,title=="Etruscan Venus")' />
  <div id="ansp6"><div></div></div>
 </li>
</ul>

Banchoff Klein Bottle
<ul>
 <li>
  <input type="submit" id="p7" class="noresult"
    onclick="makeRequest('p7');"
    value="f:=cos(x)*(cos(x/2)*(sqrt(2)+cos(y))+(sin(x/2)*sin(y)*cos(y)))" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="noresult"
    onclick="makeRequest('p8');"
    value="g:=sin(x)*(cos(x/2)*(sqrt(2)+cos(y))+(sin(x/2)*sin(y)*cos(y)))" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="noresult"
    onclick="makeRequest('p9');"
    value="h:=-sin(x/2)*(sqrt(2)+cos(y))+cos(x/2)*sin(y)*cos(y)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p7','p8','p9','p10']);"
    value='draw(surface(f,g,h),x=0..4*%pi,y=0..2*%pi,var1Steps==50,var2Steps==50,title=="Banchoff Klein Bottle")
  <div id="ansp10"><div></div></div>
 </li>
</ul>

\getchunk{page foot}
```

graphicspage.xhtml

— graphicspage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
Axiom can plot curves and surfaces of various types, as well as
lists of points in the plane.
  <table>
   <tr>
    <td>
     <a href="graphexamples.xhtml">Examples</a>
    </td>
    <td>
     See examples of Axiom graphics
    </td>
   </tr>
   <tr>
    <td>
     <a href="graph2d.xhtml">2D Graphics</a>
    </td>
    <td>
     Graphics in the real and complex plane
    </td>
   </tr>
   <tr>
    <td>
     <a href="graph3d.xhtml">3D Graphics</a>
    </td>
    <td>
     Plot surfaces, curves, or tubes around curves
    </td>
   </tr>
   <tr>
    <td>
     <a href="graphviewports.xhtml">Viewports</a>
    </td>
    <td>
     Customize graphics using Viewports
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

graphviewports.xhtml

— graphviewports.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Stand-alone Viewport</div>
  <hr/>
To get a viewport on a page, you first need to create on in Axiom and
write it out to a file that can be called up. For example, we draw a
saddle function and assign the result to the variable v.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="v:=draw(x*x-y*y,x=-1..1,y=-1..1)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Now that we've created the viewport, we want to write the data out to
a file. To do this, we use the <a href="dbopwrite.xhtml">write</a> command
which takes as arguments the viewport to write out, the title of the file
to be written to, and an optiona argument telling the write command what
type (or types) of data you want to write (in addition to the ones that
Axiom writes). The optional argument could be a string, like "pixmap", or
a list of strings, like ["postscript", "pixmap"]. We need a "bitmap" data
type to include a graph in a page so in this case, we write the viewport
and tell it to also write a "pixmap" file:
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value='write(v,"saddle","bitmap")' />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Currently supported file formats are "pixmap", "bitmap", "postscript"
and "image".

Axiom automatically adds ".view" at the end of the viewport data file
to specify the file type. The ".view" is actually a directory and
contains a bitmap file, usually called image.bm.Z, which is a compressed
bitmap. Firefox can display bitmap files, as shown here. Clicking on the
image should start a "live graphics copy" so you can manipulate the image.

<br/>
<br/>
<a href="graphics.xhtml?3dobjectgraphicspage9.view">
<img src="/research2/test/mnt/fedora5/doc/viewports/3dobjectgraphicspage9.view/image.bmp"/>
</a>
\getchunk{page foot}
```

———-

graph2d.xhtml

— graph2d.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Two Dimensional Graphics</div>
  <hr/>
<ul>
 <li> <a href="graph2donevariable.xhtml">Functions of One Variable</a><br/>
      Plot curves defined by an equation y=f(x)
 </li>
 <li> <a href="graph2dparametric.xhtml">Parametric Curves</a><br/>
      Plot curves defined by parametric equations x=f(t), y=g(t)
 </li>
 <li> <a href="graph2dpolar.xhtml">Polar Coordinates</a><br/>
      Plot curves given in polar form by an equation r=f(&#x003B8;)
 </li>
 <li> <a href="graph2dimplicit.xhtml">Implicit Curves</a><br/>
      Plot non-singulare curves defined by a polynomial equation
 </li>
 <li> <a href="graph2dlistsofpoints.xhtml">List of Points</a><br/>
      Plot lists of points in the (x,y)-plane
 </li>
</ul>
\getchunk{page foot}
```

———-

graph2dimplicit.xhtml

— graph2dimplicit.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Implicit Curves</div>
  <hr/>
Axiom has facilities for graphing a non-singular algebraic curve in a
```

```
rectangular region of the plane. An algebraic curve is a curve defined by
a polynomial equation p(x,y)=0. Non-singular means that the curve is
"smooth" in that it does not cross itself or come to a point (cusp).
Algebraically, this means that for any point (a,b) on the curve (i.e.
a point such that p(a,b)=0), the partial derivatives dp/dx(a,b) and
dp/dy(a,b) are not both zero. We require that the polynomial have rational
or integral coefficients. Here is a Cartesian ovals algebraic curve
example:
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value="p:=((x^2+y^2+1)-8*x)^2-(8*(x^2+y^2+1)-4*x-1)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value='draw(p=0,x,y,range==[-1..11,-7..7],title=="Cartesian Ovals")' />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
A range must be declared for each variable specified in the algebraic
curve equation.
\getchunk{page foot}
```

graph2dlistsofpoints.xhtml

— graph2dlistsofpoints.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">List of Points</div>
  <hr/>
Axiom has the ability to create lists of points in a two dimensional
graphics viewport. This is done by utilizing the
<a href="db.xhtml?GraphImage">GraphImage</a> and
<a href="db.xhtml?TwoDimensionalViewport">TwoDimensionalViewport</a>
domain facilities.

In this example, the <a href="dbopmakegraphimage.xhtml">makeGraphImage</a>
takes a list of lists of points parameter, a list of colors for each point
in the graph, a list of colors for each line in the graph, and a list of
numbers which indicate the size of each point in the graph. The following
lines create list of lists of points which can be read be made into two
```

```
dimensional graph images.

First we make a list of points
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value="p:=map(point,[[1.,1.],[0.,1.],[0.,0.],[1.,0.],[1.,.5],[.5,0.],[0.,.5],[.5,1.],[.25,.25],[.25,.75],[.7
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Then we select pairs of these points which represent the endpoints of lines.
<ul>
 <li>
  <input type="submit" id="p2" class="noresult"
    onclick="makeRequest('p2');"
    value="llp:=[[p.1,p.2],[p.2,p.3],[p.3,p.4],[p.4,p.1],[p.5,p.6],[p.6,p.7],[p.7,p.8],[p.8,p.5],[p.9,p.10],[p.1
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Next we set the point color and size, and the line color for all components
of the graph.
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="lsize:=[6,6,6,6,8,8,8,8,10,10,10,10]" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="noresult"
    onclick="makeRequest('p4');"
    value="pc1:=pastel red()" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="noresult"
    onclick="makeRequest('p5');"
    value="pc2:=dim green()" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="noresult"
    onclick="makeRequest('p6');"
    value="pc3:=pastel yellow()" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="noresult"
    onclick="makeRequest('p7');"
    value="lpc:=[pc1,pc1,pc1,pc1,pc2,pc2,pc2,pc2,pc3,pc3,pc3,pc3]" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p8" class="noresult"
    onclick="makeRequest('p8');"
    value="lc:=[pastel blue(), light yellow(), dim green(), bright red(), light green(), dim yellow(), bright bl
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Now the graph image is created and named according to the component
specifications indicated above. The
<a href="dbopmakeviewport2d.xhtml">makeViewport2D</a> command then
creates a two dimensional viewport for this graph according to the list
of options specified within the brackets.
<ul>
 <li>
  <input type="submit" id="p9" class="noresult"
    onclick="makeRequest('p9');"
    value="g:=makeGraphImage(llp,lpc,lc,lsize)$GRIMAGE" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
The <a href="dbopmakeviewport2d.xhtml">makeViewport2D</a> command takes a
list of options as a parameter. In this example the string "Lines" is
designated as the viewport's title.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p9','p10']);"
    value='makeViewport2D(g,[title("Lines")])$VIEW2D' />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

graph2donevariable.xhtml

— graph2donevariable.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Functions of One Variable</div>
  <hr/>
Here we wish to plot a function y=f(x) on the interval [a,b]. As an
example, let's take the function y=sin(tan(x))-tan(sin(x)) on the interval
[0,6]. Here is the simplest command that will do this:
<ul>
 <li>
```

```
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(sin(tan(x))-tan(sin(x)),x=0..6)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Notice that Axiom compiled a function before the graph was put on the
screen. The expression sin(tan(x))-tan(sin(x)) was converted to a compiled
function so that its value for various values of x could be computed
quickly and efficiently. Let's graph the same function on a different
interval and this time we'll give the graph a title. The title is a string,
which is an optional argument of the command 'draw'.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
 value='draw(sin(tan(x))-tan(sin(x)),x=10..16,title=="y=sin tan x-tan sin x")'
  />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Once again the expression sin(tan(x))-tan(sin(x)) was converted to a
compiled function before any points were computed. If you want to graph the
same function on a number of intervals, it's a good idea to write down a
function definition so that the function only has to be compiled once.
Here's an example:
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="f(x)==(x-1)*(x-2)*(x-3)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value='draw(f,0..2,title=="y=f(x) on [0,2]")' />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p5']);"
    value='draw(f,0..4,title=="y=f(x) on [0,4]")' />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Notice that our titles can be whatever we want, as long as they are
encloded by double quotes. However, a title which is too long to fit
within the viewport title window will be clipped.
\getchunk{page foot}
```

graph2dparametric.xhtml

— graph2dparametric.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Parametric Curves</div>
  <hr/>
One way of producing interesting curves is by using parametric equations.
Let x=f(t) and y=g(t) for two functions f and g as the parameter t ranges
over an interval [a,b]. Here's an example:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(curve(sin(t)*sin(2*t)*sin(3*t),sin(4*t)*sin(5*t)*sin(6*t)),t=0..2*%pi)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Here 0..2*%pi represents the interval over which the variable t ranges.
In the case of parametric curves, Axiom will compile two functions, one for
each of the functions f and g. You may also put a title on a graph. The
title may be an arbitrary string and is an optional argument to the
command 'draw'. For example:
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value='draw(curve(cos(t),sin(t)),t=0..2*%pi,title=="The Unit Circle")' />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
If you plan on plotting x=f(t), y=g(t) as t ranges over several intervals,
you may want to define functions f and g, so that they need not be
recompiled every time you create a new graph. Here's an example:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="f(t:SF):SF == sin(3*t/4)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="g(t:SF):SF == sin(t)" />
  <div id="ansp4"><div></div></div>
```

```
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p4','p5']);"
    value="draw(curve(f,g),0..%pi)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p3','p4','p6']);"
    value="draw(curve(f,g),-%pi..2*%pi)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p3','p4','p7']);"
    value="draw(curve(f,g),-4*%pi..4*%pi)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
These examples show how the curve changes as the range of the parameter
t varies.
\getchunk{page foot}
```

———-

graph2dpolar.xhtml

— graph2dpolar.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Polar Coordinates</div>
  <hr/>
Graphs in polar coordinates are given by an equation r=f(&#x003B8;) as
theta ranges over an interval. This is equivalent to the parametric curve
x=f(&#x003B8;)*cos(&#x003B8;),y=f(&#x003B8;)*sin(&#x003B8;) as &#x003B8;
ranges over the same interval. You may create such curves using the
command 'draw', with the optional argument 'coordinates==polar'.
Here are some examples:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value='draw(1,t=0..2*%pi,coordinates==polar,title=="The Unit Circle")' />
  <div id="ansp1"><div></div></div>
 </li>
```

```
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
  value='draw(sin(17*t),t=0..2*%pi,coordinates==polar,title=="A Petal Curve")'
   />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
You may also define your own functions, when you plan on plotting the same
curve as &#x003B8; varies over several intervals.
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="f(t)==cos(4*t/7)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value="draw(f,0..2*%pi,coordinates==polar)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p5']);"
    value="draw(f,0..14*%pi,coordinates==polar)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
For information on plotting graphs in other coordinate systems see the
pages for the <a href="db.xhtml?CoordinateSystems">CoordinateSystems</a>
domain.
\getchunk{page foot}
```

graph3d.xhtml

— graph3d.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Three Dimensional Graphing</div>
  <hr/>
<ul>
 <li> <a href="graph3dtwovariables.xhtml">Functions of Two Variables</a><br/>
      Plot surfaces defined by an equation z=f(x,y)
 </li>
 <li> <a href="graph3dparametric.xhtml">Parametric Curves</a><br/>
      Plot curves defined by equations x=f(t), y=g(t), z=h(t)
```

```
 </li>
 <li> <a href="graph3dtubeplots.xhtml">Parametric Tube Plots</a><br/>
      Plot a tube around a parametric space curve
 </li>
 <li> <a href="graph3dsurfaces.xhtml">Parametric Surfaces</a><br/>
      Plot surfaces defined by x=f(u,v), y=g(u,v), z=h(u,v)
 </li>
 <li> <a href="graph3dobjects.xhtml">Building Objects</a><br/>
      Create objects constructed from geometric primitives
 </li>
</ul>
\getchunk{page foot}
```

graph3dobjects.xhtml

— graph3dobjects.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Building Three Dimensional Objects from Primatives</div>
  <hr/>
Rather than using the <a href="dbopdraw.xhtml">draw</a> and
<a href="dbopmakeobject.xhtml">makeObject</a> commands, you can create
three-dimensional graphs from primitives. Operation
<a href="dbopcreate3space.xhtml">create3Space</a> creates a three-space
object to which points, curves, and polygons can be added using the
operations from the <a href="db.xhtml?ThreeSpace">ThreeSpace</a> domain.
The resulting object can then be displayed in a viewport using
<a href="dbopmakeviewport3d.xhtml">makeViewport3D</a>.

Create the empty three-space object space.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="space:=create3Space()$(ThreeSpace DFLOAT)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Objects can be sent to this space using the operations exported by the
<a href="db.xhtml?ThreeSpace">ThreeSpace</a> domain. The following examples
place curves into space.

Add these three curves to the space.
<ul>
```

```
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="closedCurve(space,[ [0,30,20],[0,30,30],[0,40,30],[0,40,100],[0,30,100],[0,30,110],[0,60,110],[0,60,1
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="closedCurve(space,[ [80,0,30],[80,0,100],[70,0,110],[40,0,110],[30,0,100],[30,0,90],[40,0,90],[40,0,9
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="closedCurve(space,[ [70,0,35],[65,0,30],[45,0,30],[40,0,35],[40,0,60],[50,0,60],[50,0,70],[30,0,70],[
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Create and display the viewport using
<a href="dbopmakeviewport3d.xhtml">makeViewport3D</a>. Options may also be
given but here are displayed as a list with values enclosed in parentheses.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5']);"
    value='makeViewport3D(space,title=="Letters")' />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
<br/>
<br/>
Cube Example
<br/>
As a second example of the use of primitives, we generate a cube using
a polygon mesh. It is important to use a consistent orientation of the
polygons for correct generation of three-dimensional objects.

Again start with an empty three-space object.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="spaceC:=create3Space()$(ThreeSpace DFLOAT)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
For convenience, give the <a href="db.xhtml?DoubleFloat">DoubleFloat</a>
values +1 and -1 names.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="x:DFLOAT:=1" />
```

```
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="y:DFLOAT:=-1" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Define the vertices of the cube.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p7','p8','p9']);"
    value="a:=point [x,x,y,1::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p7','p8','p10']);"
    value="b:=point [y,x,y,4::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p7','p8','p11']);"
    value="c:=point [y,x,x,8::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p7','p8','p12']);"
    value="d:=point [x,x,x,12::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p7','p8','p13']);"
    value="e:=point [x,y,y,16::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p7','p8','p14']);"
    value="f:=point [y,y,y,20::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p7','p8','p15']);"
    value="g:=point [y,y,x,24::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p7','p8','p16']);"
    value="h:=point [x,y,x,27::DFLOAT]$(Point DFLOAT)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Add the faces of the cube as polygons to the space using a consistent
orientation.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p6','p7','p8','p9','p10','p11','p12','p13','p14','p15','p16','p17']);"
    value="polygon(spaceC,[d,c,g,h]); polygon(spaceC,[d,h,e,a]); polygon(spaceC,[c,d,a,b]); polygon(spaceC,[g,c,
  <div id="ansp17"><div></div></div>
 </li>
</ul>
Create and display the viewport.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="handleFree(['p6','p7','p8','p9','p10','p11','p12','p13','p14','p15','p16','p17','p18']);"
    value='makeViewport3D(spaceC,title=="Cube")' />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

graph3dparametric.xhtml

— graph3dparametric.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Parametric Space Curves</div>
  <hr/>
This page describes the plotting in three dimensional space of a curve
defined by the parametric equations x=f(t), y=g(t), z=h(t), where f,g, and
h are functions of the parameter t which ranges over a specified interval.
The basic draw command for this function utilizes either the uncompiled
functions or compiled functions format and uses the
<a href="dbopcurve.xhtml">curve</a> command to specify the three
functions for the x, y, and z components of the curve. The general format
for uncompiled functions is:
<pre>
   draw(curve(f(t),g(t),h(t)), t=a..b)
```

```
</pre>
where a..b is the segment defining the interval [a,b] over which the
parameter t ranges. In this case the functions are not compiled until
the draw command is executed. Here is an example:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(curve(cos(t),sin(t),t),t=-12..12)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
In the case of compiled functions, the functions are named and compiled
independently. This is useful if you intend to use the functions often,
or if the functions are long and complex. The following lines show
functions whose parameters are of the type SmallFloat. The functions
are compiled and stored by Axiom when entered.
<ul>
 <li>
  <input type="submit" id="p2" class="noresult"
    onclick="makeRequest('p2');"
    value="i1(t:SF):SF==sin(t)*cos(3*t/5)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="i2(t:SF):SF==cos(t)*cos(3*t/5)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="noresult"
    onclick="makeRequest('p4');"
    value="i3(t:SF):SF==cos(t)*sin(3*t/5)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Once the functions are compiled the draw command only needs the names of
the functions to execute. Here is a compiled functions example:
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p3','p4','p5']);"
    value="draw(curve(i1,i2,i3),0..15*%pi)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Note that the parameter range does not take the variable name as in the
case of uncompiled functions. It is understood that the indicated range
applies to the parameter of the functions, which in this case is t.
\getchunk{page foot}
```

graph3dsurfaces.xhtml

— graph3dsurfaces.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Parametric Surfaces</div>
  <hr/>
Graphing a surface defined by x=f(u,v), y=g(u,v), z=h(u,v). This page
describes plotting of surfaces defined by the parametric equations of two
variables, x=f(u,v), y=g(u,v), and z=h(u,v), for which the ranges of u and
v are explicitly defined. The basic draw command for this function utilizes
either the uncompiled function or compiled function format and uses the
<a href="dbopsurface.xhtml">surface</a> command to specify the three
functions for the x, y, and z components of the surface. The general
format for uncompiled functions is:
<pre>
    draw(surface(f(u,v),g(u,v),h(u,v)), u=a..b, v=c..d)
</pre>
where a..b and c..d are segments defining the intervals [a,b] and [c,d]
over which the parameters u and v span. In this case the functions are
not compiled until the draw command is executed. Here is an example of a
surface plotted using the parabolic cylindrical coordinate system option:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(surface(u*cos(v),u*sin(v),v*cos(u)),u=-4..4,v=0..2*%pi,coordinates==parabolicCylindrical)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
In the case of compiled functions, the functions are named and compiled
independentlyh. This is useful if you intend to use the functions often,
or if the functions are long and complex. The following lines show functions
whose parameters are of the type SmallFloat. The functions are compiled and
stored by Axiom when entered.
<ul>
 <li>
  <input type="submit" id="p2" class="noresult"
    onclick="makeRequest('p2');"
    value="n1(u:SF,v:SF):SF == u*cos(v)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="n2(u:SF,v:SF):SF == u*sin(v)" />
```

```
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="noresult"
    onclick="makeRequest('p4');"
    value="n3(u:SF,v:SF):SF == u" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Once the function is compiled the draw command only needs the names of
the functions to execute. Here is a compiled functions example plotted
using the toroidal coordinate system option:
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p3','p4','p5']);"
    value="draw(surface(n1,n2,n3),1.0..4.0,1.0..4*%pi,coordinates==toroidal(1$SF))" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Note that the parameter ranges do not take the variable names as in the case
of uncompiled functions. The variables are entered in the order in which
they are defined in the function specification. In this case the first
range specifies the u-variable and the second range specifies the v-variable.
\getchunk{page foot}
```

graph3dtubeplots.xhtml

— graph3dtubeplots.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Parametric Tube Plots</div>
  <hr/>
This page describes the plotting in three dimensional space of a tube
around a parametric space curve defined by the parametric equations
x=f(t), y=g(t), z=h(t), where f, g, and h are functions of the parameter t
which ranges over a specified interval. The basic draw command for this
function utilizes either the uncompiled functions or compiled functions
format and uses the <a href="dbopcurve.xhtml">curve</a> command to specify
the three functions for the x, y, and z components of the curve. This uses
the same format as that for space curves except that it requires a
specification for the radius of the tube. If the radius of the tube is 0,
then the result is the space curve itself. The general format for
uncompiled functions is:
```

```
<pre>
    draw(curve(f(t),g(t),h(t)),t=a..b,tubeRadius==r)
</pre>
where a..b is the segment defining the interval [a,b] over which the
parameter t ranges, and the tubeRadius is indicated by the variable r.
In this case the functions are not compiled until the draw command is
executed. Here is an example:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(curve(sin(t)*cos(3*t/5),cos(t)*cos(3*t/5),cos(t)*sin(3*t/5)),t=0..15*%pi,tubeRadius==.15)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
In the case of compiled functions, the functions are named and compiled
independently. This is useful if you intend to use the functions often,
or if the functions are long and complex. The following lines show
functions whose parameters are of the type SmallFloat. The functions are
compiled and stored by Axiom when entered.
<ul>
 <li>
  <input type="submit" id="p2" class="noresult"
    onclick="makeRequest('p2');"
    value="t1(t:SF):SF==4/(2-sin(3*t))*cos(2*t)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="t2(t:SF):SF==4/(2-sin(3*t))*sin(2*t)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="noresult"
    onclick="makeRequest('p4');"
    value="t3(t:SF):SF==4/(2-sin(3*t))*cos(3*t)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Once the function is compiled the draw command only needs the names of
the functions to execute. Here is a compiled functions example of a trefoil
knot:
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p3','p4','p5']);"
    value="draw(curve(t1,t2,t3),0..2*%pi,tubeRadius==.2)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Note that the parameter range does not take the variable name as in the
case of uncompiled functions. It is understood that the indicated range
applies to the parameter of the functions, which in this case is t.
```

```
Typically, the radius of the tube should be set between 0 and 1. A radius
of less than 0 results in it’s positive counterpart and a radius of greater
than one cause self-intersection.
\getchunk{page foot}
```

———-

graph3dtwovariables.xhtml

— graph3dtwovariables.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Functions of Two Variables</div>
  <hr/>
This page describes the plotting of surfaces defined by an equation of
two variables, z=f(x,y), for which the ranges of x and y are explicitly
defined. The basic draw command for this function utilizes either the
uncompiled function or compiled function format. The general format for an
uncompiled function is:
<pre>
   draw(f(x,y), x=a..b, y=c..d)
</pre>
where a..b and c..d are segments defining the intervals [a,b] and [c,d]
over which the variables x and y span. In this case, the function is not
compiled until the draw command is executed. Here is an example:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="draw(cos(x*y),x=-3..3,y=-3..3)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
In the case of a compiled function, the function is named and compiled
independently. This is useful if you intend to use a function often, or
if the function is long and complex. The following line shows a function
whose parameters are of the type SmallFloat. The function is compiled and
stored by Axiom when it is entered.
<ul>
 <li>
  <input type="submit" id="p2" class="noresult"
    onclick="makeRequest('p2');"
    value="f(x:SF,y:SF):SF==sin(x)*cos(y)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
```

```
Once the function is compiled the draw command only needs the name of the
function to execute. Here is a compiled function example:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="draw(f,-%pi..%pi,-%pi..%pi)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Note that the parameter ranges do not take the variable names as in the
case of uncompiled functions. The variables are entered in the order in
which they are defined in the function specificationl. In this case the
first range specifies the x-variable and the second range specifies the
y-variable.
\getchunk{page foot}
```

htxtoppage.xhtml

— htxtoppage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      htxtoppage not implemented
\getchunk{page foot}
```

indefiniteintegral.xhtml

— indefiniteintegral.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    var myform = document.getElementById("form2");
    return('integrate('+myform.expr.value+','+myform.vars.value+')');
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <form id="form2">
   Enter the function you want to integrate:<br/>
   <input type="text" id="expr" tabindex="10" size="50"
     value="1/(x^2+6)"/><br/>
```

```
   Enter the variable of integration:
   <input type="text" id="vars" size="5" tabindex="20" value="x"/><br/>
  </form>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

introtofloat.xhtml

— **introtofloat.xhtml** —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Introduction to Float</div>
  <hr/>
Scientific notation is supported for input and output of floating
point numbers. A floating point number is written as a string of
digits containing a decimal point optionally followed by the letter "E",
and then the exponent. We begin by doing some calculations using
arbitrary precision floats. The default precision is twenty decimal
digits.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="1.234" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
A decimal base for the exponent is assumed, so the number 1.234E2 denotes
1.234*10**2
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="1.234E2" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The normal arithmetic operations are available for floating point numbers.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="sqrt(1.2+2.3/3.4^4.5)" />
```

```
  <div id="ansp3"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

jenks.xhtml

— jenks.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
 <center>
  <a href="axbook/book-contents.xhtml">
   <img src="axbook/ps/lightbayou.png"/>
  </a>
 </center>
 <center>
  <h1>
   <a href="axbook/book-contents.xhtml">
    AXIOM -- Richard D. Jenks and Robert S. Sutor
   </a>
  </h1>
 </center>
 <center>
  <h2>
   <a href="axbook/book-contents.xhtml">
    The Scientific Computation System
   </a>
  </h2>
 </center>
 <center>
  <h2>
   <a href="axbook/book-contents.xhtml">
    Volume 0 -- The Textbook
   </a>
  </h2>
 </center>
 <a href="axbook/book-contents.xhtml#chapter0">
  Chapter 0: Introduction to Axiom
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter1">
  Chapter 1: An Overview of Axiom
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter2">
 Chapter 2: Using Types and Modes
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter3">
 Chapter 3: Using HyperDoc
 </a><br/>
```

```
 <a href="axbook/book-contents.xhtml#chapter4">
 Chapter 4: Input Files and Output Styles
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter5">
 Chapter 5: Overview of Interactive Language
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter6">
 Chapter 6: User-Defined Functions, Macros and Rules
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter7">
 Chapter 7: Graphics
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter8">
 Chapter 8: Advanced Problem Solving
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter9">
 Chapter 9: Some Examples of Domains and Packages
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter10">
 Chapter 10: Interactive Programming
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter11">
 Chapter 11: Packages
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter12">
 Chapter 12: Categories
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter13">
 Chapter 13: Domains
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter14">
 Chapter 14: Browse
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter15">
 Chapter 15: What's New in Axiom Version 2.0
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter17">
 Chapter 17: Categories
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter18">
 Chapter 18: Domains
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter19">
 Chapter 19: Packages
 </a><br/>
 <a href="axbook/book-contents.xhtml#chapter21">
 Chapter 21: Programs for AXIOM Images
 </a><br/>
\getchunk{page foot}
```

———-

laurentseries.xhtml

— laurentseries.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function').value;
    myivar = document.getElementById('ivar').value;
    mypvar = document.getElementById('pvar').value;
    myevar = document.getElementById('evar').value;
    myival = document.getElementById('ival').value;
    mysval = document.getElementById('sval').value;
    ans = 'series('+myivar+'+->'+myfunc+','+mypvar+'='+myevar+','+
         myival+'..,'+mysval+')';
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
      Enter the formula for the general coefficient of the series:
    </td>
   </tr>
   <tr>
    <td>
     <input type="text" id="function" size="80" tabindex="10"
       value="(-1)^(n-1)/(n+2)"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the index variable for your formula:
     <input type="text" id="ivar" size="10" tabindex="20" value="n"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the power series variable:
     <input type="text" id="pvar" size="10" tabindex="30" value="x"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the point about which to expand:
     <input type="text" id="evar" size="10" tabindex="40" value="0"/>
    </td>
```

```
   </tr>
  </table>
For Laurent Series, the exponent of the power series variable ranges
from an initial value, an arbitrary integer value, to plus
infinity; the step size is any positive integer.
  <table>
   <tr>
    <td>
     Enter the initial value of the index (an integer):
     <input type="text" id="ival" size="10" tabindex="50" value="-1"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the step size (a positive integer):
     <input type="text" id="sval" size="10" tabindex="60" value="1"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

linalgpage.xhtml

— **linalgpage.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Linear Algebra</div>
  <hr/>
  <table>
   <tr>
    <td>
     <a href="linintro.xhtml">Introduction</a>
    </td>
   </tr>
   <tr>
    <td>
     Create and manipulate matrices. Work with the entries of a
     matrix. Perform matrix arithmetic.
    </td>
   </tr>
   <tr>
    <td>
     <a href="lincreate.xhtml">Creating Matrices</a>
    </td>
   </tr>
```

```
   <tr>
    <td>
     Create matrices from scratch and from other matrices
    </td>
   </tr>
   <tr>
    <td>
     <a href="linoperations.xhtml">Operations on Matrices</a>
    </td>
   </tr>
   <tr>
    <td>
     Algebraic manipulations with matrices. Compute the inverse,
     determinant, and trace of a matrix. Find the rank, nullspace,
     and row echelon form of a matrix.
    </td>
   </tr>
   <tr>
    <td>
     <a href="lineigen.xhtml">Eigenvalues and Eigenvectors</a>
    </td>
   </tr>
   <tr>
    <td>
     How to compute eigenvalues and eigenvectors
    </td>
   </tr>
  </table>
<hr/>
 <ul>
  <li>
   <a href="linhilbert.xhtml">
    Example: Determinant of a Hilbert Matrix
   </a>
  </li>
  <li>
   <a href="linpermaent.xhtml">
    Computing the Permanent
   </a>
  </li>
  <li>
   <a href="linvectors.xhtml">
    Working with Vectors
   </a>
  </li>
  <li>
   <a href="linsquarematrices.xhtml">
    Working with Square Matrices
   </a>
  </li>
  <li>
   <a href="lin1darrays.xhtml">
    Working with One-Dimensional Arrays
   </a>
```

```
  </li>
  <li>
   <a href="lin2darrays.xhtml">
    Working with Two-Dimensional Arrays
   </a>
  </li>
  <li>
   <a href="linconversion.xhtml">
    Conversion (Polynomials of Matrices)
   </a>
  </li>
 </ul>
\getchunk{page foot}
```

linconversion.xhtml

— linconversion.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Conversion</div>
  <hr/>
Conversion is the process of changing an object of one type into an
object of another type. The syntax for conversion is object::newType.
<hr/>
By default, 3 has the type
<a href="db.xhtml?PositiveInteger">PositiveInteger</a>
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="3" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
We can change thisinto an object of type
<a href="db.xhtml?Fraction(Integer)">Fraction Integer</a> by using "::".
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="3::Fraction Integer" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
```

```
A coercion is a special kind of conversion that Axiom is allowed to do
automatically when you enter an expression. Coercions are usually
somewhat safer than more general conversions. The Axiom library contains
operations called
<a href="dbopcoerce.xhtml">coerce</a> and
<a href="dbopconvert.xhtml">convert</a>. Only the
<a href="dbopcoerce.xhtml">coerce</a> operations can be used by the
interpreter to change an object into an object of another type unless
you explicitly use a "::".

By now you will be quite familiar with what types and modes look like.
It is useful to think of a type or mode as a pattern for what you want
the result to be. Let's start with a square matrix of polynomials with
complex rational number coefficients.
<ul>
 <li>
  <input type="submit" id="p3" class="noresult"
    onclick="makeRequest('p3');"
    value="m:SquareMatrix(2,POLY COMPLEX FRAC INT)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value="m:=matrix [ [x-3/4*%i,z*y^2+1/2],[3/7*%i*y^4-x,12-%i*9/5]]" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
We first want to interchange the <a href="db.xthml?Complex">Complex</a>
and <a href="db.xhtml?Fraction">Fraction</a> layers. We do the conversion
by doing the interchange in the type expression.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p4','p5']);"
    value="m1:=m::SquareMatrix(2,POLY FRAC COMPLEX INT)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Interchange the <a href="db.xhtml?Polynomial">Polynomial</a> and the
<a href="db.xhtml?Fraction">Fraction</a> levels.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p3','p4','p5','p6']);"
    value="m2:=m1::SquareMatrix(2,FRAC POLY COMPLEX INT)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Interchange the <a href="db.xhtml?Polynomial">Polynomial</a> and the
<a href="db.xhtml?Complex">Complex</a> levels.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
```

```
    onclick="handleFree(['p3','p4','p5','p6','p7']);"
    value="m3:=m2::SquareMatrix(2,FRAC COMPLEX POLY INT)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
All the entries have changed types, although in comparing the last two
results only the entry in the lower left corner looks different. We did
all the intermediate steps to show you what Axiom can do.

In fact, we could have combined all these into one conversion.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p3','p4','p8']);"
    value="m::SquareMatrix(2,FRAC COMPLEX POLY INT)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>

There are times when Axiom is not able to do the conversion in one step.
You may need to break up the transformation into several conversions in
order to get an object of the desired type.

We cannot move either the <a href="db.xhtml?Fraction">Fraction</a> or
<a href="db.xhtml?Complex">Complex</a> above (or to the left of,
depending on how you look at it)
<a href="db.xhtml?SquareMatrix">SquareMatrix</a> because each of these
levels requires that its argument type have commutative multiplication,
whereas <a href="db.xhtml?SquareMatrix">SquareMatrix</a> does not.
(<a href="db.xhtml?Fraction">Fraction</a> requires that its argument
belong to the category
<a href="db.xhtml?IntegralDomain">IntegralDomain</a> and
<a href="db.xhtml?Complex">Complex</a> requires that its argument belongs to
<a href="db.xhtml?CommutativeRing">CommutativeRing</a>. See the
<a href="axbook/section-2.1.xhtml">Jenks section 2.1</a> for a brief
discussion of categories. The <a href="db.xhtml?Integer">Integer</a> level
did not move anywhere because it does not allow any arguments. We also did
not move the <a href="db.xhtml?SquareMatrix">SquareMatrix</a> part
anywhere, but we could have. Recall that m looks like this:
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p3','p4','p9']);"
    value="m" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
If we want a polynomial with matrxi coefficients rather than a matrix with
polynomial entries, we can just do the conversion.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p3','p4','p10']);"
    value="m::POLY SquareMatrix(2,COMPLEX FRAC INT)" />
```

```
  <div id="ansp10"><div></div></div>
 </li>
</ul>
We have not yet used modes for any conversions. Modes are a great
shorthand for indicating the type of the object you want. Instead of
using the long type expression in the last example we could have
simply said this:
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p3','p4','p11']);"
    value="m::POLY ?" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
We can also indicate more structure if we want the entries of the matrices
to be fractions.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p3','p4','p12']);"
    value="m::POLY SquareMatrix(2,FRAC ?)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

lincreate.xhtml

— lincreate.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Creating Matrices</div>
  <hr/>
There are many ways to create a matrix from a collection of values or
from existing matrices.

If the matrix has almost all items equal to the same value, use
<a href="dbopnew.xhtml">new</a> to create a matrix filled with that value
and then reset the entries that are different.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
```

```
    value="m:Matrix(Integer):=new(3,3,0)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
To change the entry in the second row, third column to 5, use
<a href="dbopsetelt.xhtml">setelt</a>.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="setelt(m,2,3,5)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
An alternative syntax is to use assignment.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="m(1,2):=10" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The matrix was destructively modified.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="m" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
If you already have the matrix entries as a list of lists, use
<a href="dbopmatrix.xhtml">matrix</a>.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="matrix [[1,2,3,4],[0,9,8,7]]" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
If the matrix is diagonal, use
<a href="dbopdiagonalmatrix.xhtml">diagonalMatrix</a>
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="dm:=diagonalMatrix [1,x^2,x^3,x^4,x^5]" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Use <a href="dbopsetrowbang.xhtml">setRow!</a> and
<a href="dbopsetcolumnbang.xhtml">setColumn!</a>
```

```
to change a row or column of a matrix.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value="setRow!(dm,5,vector [1,1,1,1,1])" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p6','p7','p8']);"
    value="setColumn!(dm,2,vector [y,y,y,y,y])" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Use <a href="dbopcopy.xhtml">copy</a> to make a copy of a matrix.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p6','p7','p8','p9']);"
    value="cdm:=copy(dm)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
This is useful if you intend to modify a matrix destructively but want a
copy of the original.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p6','p7','p8','p9','p10']);"
    value="setelt(dm,4,1,1-x^7)" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p6','p7','p8','p9','p10','p11']);"
    value="[dm,cdm]" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
Use <a href="dbopsubmatrix.xhtml">subMatrix</a>(dm,2,3,2,4) to extract
part of an existing matrix. The syntax is
<pre>
  subMatrix(m,firstrow,lastrow,firstcol,lastcol)
</pre>
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p6','p7','p8','p9','p10','p11','p12']);"
    value="subMatrix(dm,2,3,2,4)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
To change a submatrix, use
```

```
<a href="dbopsetsubmatrixbang.xhtml">setsubMatrix!</a>.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="d:=diagonalMatrix [1.2,-1.3,1.4,-1.5]" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
If e is too big to fit where you specify, an error message is displayed. Use
<a href="dbopsubmatrix.xhtml">subMatrix</a>.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="makeRequest('p14');"
    value="e:=matrix [[6.7,9.11],[-31.33,67.19]]" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
This changes the submatrix of d whose upper left corner is at the first row
and second column and whose size is that of e.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p13','p14','p15']);"
    value="setsubMatrix!(d,1,2,e)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p13','p14','p15','p16']);"
    value="d" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Matrices can be joined either horizontally or vertically to make new
matrices.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="makeRequest('p17');"
    value="a:=matrix [[1/2,1/3,1/4],[1/5,1/6,1/7]]" />
  <div id="ansp17"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest('p18');"
    value="b:=matrix [[3/5,3/7,3/11],[3/13,3/17,3/19]]" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
Use <a href="dbophorizconcat.xhtml">horizConcat</a> to append them side to
```

```
side. The two matrices must have the same number of rows.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p17','p18','p19']);"
    value="horizConcat(a,b)" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
Use <a href="dbopvertconcat.xhtml">vertConcat</a> to stack one upon the
other. The two matrices must have the same number of columns.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p17','p18','p20']);"
    value="vab:=vertConcat(a,b)" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
The operation <a href="dboptranspose.xhtml">transpose</a> is used to create
a new matrix by reflection across the main diagonal.
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="handleFree(['p17','p18','p20','p21']);"
    value="transpose vab" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

lineigen.xhtml

— lineigen.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Computation of Eigenvalues and Eigenvectors</div>
  <hr/>
In this section we show you some of Axiom's facilities for computing and
manipulating eigenvalues and eigenvectors, also called characteristic
values and characteristic vectors, respectively.

Let's first create a matrix with integer entries.
<ul>
```

```
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="m1:=matrix [[1,2,1],[2,1,-2],[1,-2,4]]" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
To get a list of the rational eigenvalues, use the operation
<a href="dbopeigenvalues.xhtml">eigenvalues</a>.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="leig:=eigenvalues(m1)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Given an explicit eigenvalue,
<a href="dbopeigenvector.xhtml">eigenvector</a> computes the eigenvectors
corresponding to it.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="eigenvector(first(leig),m1)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The operation <a href="dbopeigenvectors.xhtml">eigenvectors</a> returns a
list of pairs of values and vectors. When an eigenvalue is rational, Axiom
gives you the value explicitly; otherwise, its minimal polynomial is given,
(the polynomial of lowest degree with the eigenvalues as roots), together
with a parametric representation of the eigenvector using the eigenvalue.
This means that if you ask Axiom to <a href="dbopsolve.xhtml">solve</a>
the minimal polynomial, then you can substitute these roots into the
parametric form of the corresponding eigenvectors.

You must be aware that unless an exact eigenvalue has been computed,
the eigenvector may be badly in error.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="eigenvectors(m1)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Another possibility is to use the operation
<a href="dbopradicaleigenvectors.xhtml">radicalEigenvectors</a> tries to
compute explicitly the eignevectors in terms of radicals.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
```

```
    value="radicalEigenvectors(m1)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Alternatively, Axiom can compute real or complex approximations to the
eigenvectors and eigenvalues using the operations
<a href="dboprealeigenvectors.xhtml">realEigenvectors</a> or
<a href="dbopcomplexeigenvectors.xhtml">complexEigenvectors</a>. They
each take an additional argument epsilon to specify the "precision"
required. In the real case, this means that each approximation will be
within plus or minus epsilon of the actual result. In the complex case, this
means that each approximation will be within plus or minus epsilon of the
actual result in each of the real and imaginary parts.

The precision can be specified as a <a href="db.xhtml?Float">Float</a> if
the results are desired in floating-point notation, or as
<a href="dbfractioninteger.xhtml">Fraction Integer</a> if the results are
to be expressed using rational (or complex rational) numbers.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p6']);"
    value="realEigenvectors(m1,1/1000)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
If an n by n matrix has n distinct eigenvalues (and therefore n eigenvectors)
the operation <a href="dbopeigenmatrix.xhtml">eigenMatrix</a> gives you a
matrix of the eigenvectors.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p7']);"
    value="eigenMatrix(m1)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="m2:=matrix [[-5,-2],[18,7]]" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p8','p9']);"
    value="eigenMatrix(m2)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
If a symmetric matrix has a basis of orthonormal eigenvectors, then
<a href="dboporthonormalbasis.xhtml">orthonormalBasis</a> computes a list
of these vectors.
<ul>
 <li>
```

```
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="m3:=matrix [[1,2],[2,1]]" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p10','p11']);"
    value="orthonormalBasis(m3)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

linhilbert.xhtml

— linhilbert.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">An Example: Determinant of a Hilbert Matrix</div>
  <hr/>
Consider the problem of computing the determinant of a 10 by 10 Hilbert
matrix. The (i,j)-th entry of a Hilbert matrix is given by 1/(i+j+1).

First do the computation using rational numbers to obtain the exact result.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
value="a:MATRIX FRAC INT:=matrix [[1/(i+j+1) for j in 0..9] for i in 0..9]" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="d:=determinant a" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="d::Float" />
  <div id="ansp3"><div></div></div>
 </li>
```

```
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value=
  "b:Matrix DFLOAT:=matrix [[1/(i+j+1$DFLOAT) for j in 0..9] for i in 0..9]"/>
  <div id="ansp4"><div></div></div>
 </li>
</ul>
The result given by hardware floats is correct only to four significant digits
of precision. In the jargon of numerical analysis, the Hilbert matrix is said
to be "ill-conditioned".
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="determinant b" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Now repeat the computation at a higher precision using Float.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="digits 40" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value=
  "c:Matrix Float:=matrix [[1/(i+j+1$Float) for j in 0..9] for i in 0..9]" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p6','p7','p8']);"
    value="determinant c" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Reset <a href="dbopdigits.xhtml">digits</a> to its default value.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="digits 20" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

linintro.xhtml

— linintro.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Expanding to Higher Dimensions</div>
  <hr/>
To get higher dimensional aggregates, you can create one-dimensional
aggregates with elements that are themselves aggregates, for example,
lists of list, one-dimensional arrays of list of multisets, and so on. For
applications requiring two-dimensional homogeneous aggregates, you will
likely find two-dimensional arrays and matrices useful.

The entries in <a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a>
and <a href="?Matrix">Matrix</a> objects are all the same type,
except that those for <a href="db.xhtml?Matrix">Matrix</a> must belong to a
<a href="db.xhtml?Ring">Ring</a>. You create and access elements in roughly
the same way. Since matrices have an understood algebraic structure, certain
algebraic operations are available for matrices but not for arrays. Because
of this, we limit our discussion here to <a href="db.xhtml?Matrix">Matrix</a>,
that can be regarded as an extension of
<a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a>. See
<a href="pagetwodimensionalarray.xhtml">TwoDimensionalArray</a>
For more
information about Axiom's linear algebra facilities see
<a href="pagematrix.xhtml">Matrix</a>,
<a href="pagepermanent.xhtml">Permanent</a>,
<a href="pagesquarematrix.xhtml">SquareMatrix</a>,
<a href="pagevector.xhtml">Vector</a>,
<a href="axbook/section-8.4.xhtml">
Computation of Eigenvalues and Eigenvectors</a>, and
<a href="axbook/section-8.5.xhtml">
Solution of Linear and Polynomial Equations</a>.

You can create a matrix from a list of lists, where each of the inner
lists represents a row of the matrix.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="m:=matrix([[1,2],[3,4]])" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The "collections" construct (see
<a href="axbook/section-5.5.xhtml">
```

```
Creating Lists and Streams with Iterators</a>)
is useful for creating matrices whose entries are given by formulas.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="matrix([[1/(i+j-x) for i in 1..4] for j in 1..4])" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Let vm denote the three by three Vandermonde matrix.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="vm:=matrix [[1,1,1],[x,y,z],[x*x,y*y,z*z]]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Use this syntax to extract an entry in the matrix.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value="vm(3,3)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
You can also pull out a <a href="dboprow.xhtml">row</a> or a column.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p5']);"
    value="column(vm,2)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
You can do arithmetic.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p3','p6']);"
    value="vm*vm" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
You can perform operations such as
<a href="dboptranspose.xhtml">transpose</a>,
<a href="dboptrace.xhtml">trace</a>, and
<a href="dbopdeterminant.xhtml">determinant</a>
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p3','p7']);"
```

```
    value="factor determinant vm" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

linoperations.xhtml

— linoperations.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Operations on Matrices</div>
  <hr/>
Axiom provides both left and right scalar multiplication.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="m:=matrix [[1,2],[3,4]]" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="4*m*(-5)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
You can add, subtract, and multiply matrices provided, of course, that the
matrices have compatible dimensions. If not, an error message is displayed.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="n:=matrix([[1,0,-2],[-3,5,1]])" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
This following product is defined but n*m is not.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p3','p4']);"
    value="m*n" />
```

```
  <div id="ansp4"><div></div></div>
 </li>
</ul>
The operations <a href="dbopnrows.xhtml">nrows</a> and
<a href="dbopncols.xhtml">ncols</a> return the number of rows and
columns of a matrix. You can extract a row or a column of a matrix using
the operations <a href="dboprow.xhtml">row</a> and
<a href="dbopcolumn.xhtml">column</a>. The object returned ia a
<a href="db.xhtml?Vector">Vector</a>. Here is the third column of the matrix n.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p5']);"
    value="vec:=column(n,3)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
You can multiply a matrix on the left by a "row vector" and on the right by
a "column vector".
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p5','p6']);"
    value="vec*m" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The operation <a href="dbopinverse.xhtml">inverse</a> computes the inverse
of a matrix if the matrix is invertible, and returns "failed" if not. This
Hilbert matrix invertible.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="hilb:=matrix([[1/(i+j) for i in 1..3] for j in 1..3])" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p7','p8']);"
    value="inverse(hilb)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
This matrix is not invertible.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="mm:=matrix([[1,2,3,4],[5,6,7,8],[9,10,11,12],[13,14,15,16]])" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
```

```
    onclick="handleFree(['p9','p10']);"
    value="inverse(mm)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
The operation <a href="dbopdeterminant.xhtml">determinant</a> computes the
determinant of a matrix provided that the entries of the matrix belong to a
<a href="db.xhtml?CommutativeRing">CommutativeRing</a>. The above matrix mm
is not invertible and, hence, must have determinant 0.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p9','p11']);"
    value="determinant(mm)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
The operation <a href="dboptrace.xhtml">trace</a> computes the trace of a
square matrix.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p9','p12']);"
    value="trace(mm)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
The operation <a href="dboprank.xhtml">rank</a> computes the rank of a matrix:
the maximal number of linearly independent rows or columns.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p9','p13']);"
    value="rank(mm)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
The operation <a href="dbopnullity.xhtml">nullity</a> computes the nullity
of a matrix: the dimension of its null space.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p9','p14']);"
    value="nullity(mm)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
The operation <a href="dbopnullspace.xhtml">nullSpace</a> returns a list
containing a basis for the null space of a matrix. Note that the nullity is
the number of elements in a basis for the null space.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p9','p15']);"
```

```
    value="nullSpace(mm)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
The operation <a href="dboprowechelon.xhtml">rowEchelon</a> returns the row
echelon form of a matrix. It is easy to see that the rank of this matrix is
two and that its nullity is also two.
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p9','p16']);"
    value="rowEchelon(mm)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
For more information see
<a href="axbook/section-1.6.xhtml">Expanding to Higher Dimensions</a>,
<a href="axbook/section-8.4.xhtml">
Computation of Eigenvalues and Eigenvectors</a>, and
<a href="axbook/section-9.27.xhtml#subsec-9.27.4">
An Example: Determinant of a Hilbert Matrix</a>. Also see
<a href="db.xhtml?Permanent">Permanent</a>,
<a href="db.xhtml?Vector">Vector</a>,
<a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a>, and
<a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a>. Issue the
system command
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="showcall('p17');"
   value=")show Matrix"/>
  <div id="ansp17"><div></div></div>
 </li>
</ul>
to display the full ist of operations defined by
<a href="db.xhtml?Matrix">Matrix</a>.
\getchunk{page foot}
```

linpermaent.xhtml

— linpermaent.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Permanent</div>
```

```
  <hr/>
The package <a href="db.xhtml?Permanent">Permanent</a> provides the function
<a href="dboppermanent.xhtml">permanent</a> for square matrices. The
<a href="dboppermanent.xhtml">permanent</a> of a square matrix can be
computed in the same way as the determinant by expansion of minors except
that for the permanent the sign for each element is 1, rather than being 1
if the row plus column indices is positive and -1 otherwise. This function
is much more difficult to compute efficiently than the
<a href="dbopdeterminant.xhtml">determinant</a>. An example of the use of
<a href="dboppermanent.xhtml">permanent</a> is the calculation of the nth
derangement number, defined to be the number of different possibilities
for n couples to dance but never with their own spouse. Consider an n by x
matrix with entries 0 on the diagonal and 1 elsewhere. Think of the rows as
one-half of each couple (for example, the males) and the columns the other
half. The permanent of such a matrix gives the desired derangement number.
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value=
     "kn n == (r:MATRIX INT:=new(n,n,1); for i in 1..n repeat r.i.i:=0; r)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Here are some derangement numbers, which you see grow quite fast.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="permanent(kn(5)::SQMATRIX(5,INT))" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="[permanent(kn(n)::SQMATRIX(n,INT)) for n in 1..13]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

linsquarematrices.xhtml

— linsquarematrices.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
```

```
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">SquareMatrix</div>
  <hr/>
The top level matrix type in Axiom is
<a href="db.xhtml?Matrix">Matrix</a>, see
(<a href="pagematrix.xhtml">Matrix</a>), which provides basic arithmetic
and linear algebra functions. However, since the matrices can be of any
size it is not true that any pair can be added or multiplied. Thus
<a href="db.xhtml?Matrix">Matrix</a> has little algebraic structure.

Sometimes you want to use matrices as coefficients for polynomials or in
other algebraic contexts. In this case,
<a href="db.xhtml?SquareMatrix">SquareMatrix</a> should be used. The
domain <a href="db.xhtml?SquareMatrix">SquareMatrix(n,R)</a> gives the
ring of n by n square matrices over R.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="m:=squareMatrix [[1,-%i],[%i,4]]" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The usual arithmetic operations are available.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="m*m-m" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Square matrices can be used where ring elements are required. For example,
here is a matrix with matrix entries.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="mm:=squareMatrix [[m,1],[1-m,m^2]]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Or you can construct a polynomial with square matrix coefficients.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="p:=(x+m)^2" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
This value can be converted to a square matrix with polynomial coefficients.
<ul>
```

```
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p4','p5']);"
    value="p::SquareMatrix(2,?)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
For more information on related topics see
<a href="axbook/section-2.2.xhtml#subsec-2.2.4">Modes</a> and
<a href="pagematrix.xhtml">Matrix</a>. Issue the system command
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="showcall('p6');"
   value=")show SquareMatrix"/>
  <div id="ansp6"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?SquareMatrix">SquareMatrix</a>.
\getchunk{page foot}
```

linvectors.xhtml

— linvectors.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Vector</div>
  <hr/>
The <a href="db.xhtml?Vector">Vector</a> domain is used for storing
data in a one-dimensonal indexed data structure. A vector is a
homogeneous data structure in that all the components of the vector
must belong to the same Axiom domain. Each vector has a fixed length
specified by the user; vectors are not extensible. This domain is
similar to the
<a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a> domain,
except that when the components of a
<a href="db.xhtml?Vector">Vector</a> belong to a
<a href="db.xhtml?Ring">Ring</a>, arithmetic operations are provided.
For more examples of operations that are defined for both
<a href="db.xhtml?Vector">Vector</a> and
<a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a>, see
<a href="pageonedimensionalarray.xhtml">OneDimensionalArray</a>.
```

```
As with the <a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a>
domain, a
<a href="db.xhtml?Vector">Vector</a> can be created by calling the operation
<a href="dbopnew.xhtml">new</a>, its components can be accessed by calling
the operations <a href="dbopelt.xhtml">elt</a> and
<a href="dbopqelt.xhtml">qelt</a>, and its components can be reset by
calling the operations
<a href="dbopsetelt.xhtml">setelt</a> and
<a href="dbopseteltbang.xhtml">setelt!</a>. This creates a vector of
integers of length 5 all of whose components are 12.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="u:VECTOR INT:=new(5,12)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
This is how you create a vector from a list of its components.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="v:VECTOR INT:=vector([1,2,3,4,5])" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Indexing for vectors begins at 1. The last element has index equal to
the length of the vector, which is computed by
<a href="dboplength.xhtml">#</a>.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="#(v)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
This is the standard way to use <a href="dbopelt.xhtml">elt</a> to extract
an element.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p2','p4']);"
    value="v.2" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
This is the standard way to use setelt to change an element. It is the
same as if you had typed setelt(v,3,99).
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p5']);"
```

```
    value="v.3:=99" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Now look at v to see the change. You can use
<a href="dbopqelt.xhtml">qelt</a> and
<a href="dbopqseteltbang.xhtml">qsetelt!</a> (instead of
<a href="dbopelt.xhtml">elt</a> and
<a href="dbopsetelt.xhtml">setelt</a>, respectively) but only when you
know that the indexis within the valid range.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p2','p6']);"
    value="v" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
When the components belong to a
<a href="db.xhtml?Ring">Ring</a>,
Axiom provides arithmetic operations for
<a href="db.xhtml?Vector">Vector</a>. These include left and right
scalar multiplication.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p2','p7']);"
    value="5*v" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p2','p8']);"
    value="v*7" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="w:VECTOR INT:=vector([2,3,4,5,6])" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Addition and subtraction are also available
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p2','p9','p10']);"
    value="v+w" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
Of course, when adding or subtracting, the two vectors must have the
same length or an error message is displayed.
```

```
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p9','p11']);"
    value="v-w" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
For more information about other aggregate domains, see
<a href="pagelist.xhtml">List</a>,
<a href="pagematrix.xhtml">Matrix</a>,
<a href="pageonedimensionalarray.xhtml">OneDimensionalArray</a>.
<a href="pageset.xhtml">Set</a>,
<a href="pagetable.xhtml">Table</a>, and
<a href="pagetwodimensionalarray.xhtml">TwoDimensionalArray</a>.
Issue the system command
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="showcall('p12');"
   value=")show Vector"/>
  <div id="ansp12"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?Vector">Vector</a>.
\getchunk{page foot}
```

lin1darrays.xhtml

— lin1darrays.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">One Dimensional Array</div>
  <hr/>
The <a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a> is used
for storing data in a one-dimensional indexed data structure. Such an array
is a homogeneous data structure in that all the entries of the array must
belong to the same Axiom domain. Each array has a fixed length specified
by the user and arrays are not extensible. The indexing of
one-dimensional arrays is one-based. This means that the "first" element of
an array is given the index 1. See also
<a href="db.xhtml?Vector">Vector</a> and
<a href="db.xhtml?FlexibleArray">FlexibleArray</a>. To create a
```

```
one-dimensional array, apply the operation
<a href="dboponedimensionalarray.xhtml">oneDimensionalArray</a> to a list.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest(’p1’);"
    value="oneDimensionalArray [i^2 for i in 1..10]" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Another approach is to first create a, a one-dimensional array of 10 0’s.
<a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a> has a
convenient abbreviation
<a href="db.xhtml?OneDimensionalArray">ARRAY1</a>.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest(’p2’);"
    value="a:ARRAY1 INT := new(10,0)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Set each ith element to i, then display the result.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree([’p2’,’p3’]);"
    value="for i in 1..10 repeat a.i:=i ; a" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Square each element by mapping the function i+->i**2 onto each element.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’]);"
    value="map!(i+->i^2,a); a" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Reverse the elements in place.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’]);"
    value="reverse! a" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Swap the 4th and 5th element.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’,’p4’,’p5’,’p6’]);"
```

```
    value="swap!(a,4,5); a" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Sort the elements in place.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7']);"
    value="sort! a" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Create a new one-dimensional array b containing the last 5 elements of a.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8']);"
    value="b:=a(6..10)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Replace the first 5 elements of a with those of b.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5','p6','p7','p8','p9']);"
    value="copyInto!(a,b,1)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———-

lin2darrays.xhtml

— lin2darrays.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Two Dimensional Array</div>
  <hr/>
The <a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a> is used
for storing data in a two-dimensional data structure indexed by row and by
column. Such an array is a homogeneous data structure in that all the
entries of the array must belog to the same Axiom domain (although see
```

```
the <a href="axbook/section-2.6.xhtml">Any</a> domain). Each array has a
fixed number of rows and columns specified by the user and arrays are not
extensible. In Axiom, the indexing of two-dimensional arrays is one-based.
This means that both the "first" row of an array and the "first" column of
an array are given the index 1. Thus, the entry in the upper left corner
of an array is in position (1,1).

The operation <a href="dbopnew.xhtml">new</a> creates an array with a
specified number of rows and columns and fills the components of that
array with a specified entry. The arguments of this operation specify the
number of rows, the number of columns, and the entry. This creates a
five-by-four array of integers, all of which are zero.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="arr:ARRAY2 INT:=new(5,4,0)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The entries of this array can be set to other integers using the
operation <a href="dbopsetelt.xhtml">setelt</a>.

Issue this to set the element in the upper left corner of this array to 17.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="setelt(arr,1,1,17)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Now the first element of the array is 17.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="arr" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Likewise, elements of an array are extracted using the operation
<a href="dbopelt.xhtml">elt</a>.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p4']);"
    value="elt(arr,1,1)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Another way to use these two operations is as follows. This sets the
element in position (3,2) of the array to 15.
<ul>
```

```
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p5']);"
    value="arr(3,2):=15" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
This extracts the element in position (,32) of the array.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p5','p6']);"
    value="arr(3,2)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The operations <a href="dbopelt.xhtml">elt</a> and
<a href="dbopsetelt.xhtml">setelt</a> come equipped with an error check which
verifies that the indices are in the proper ranges. For example, the
above array has five rows and four columns, so if you ask for the entry
in position (6,2) with arr(6,2) Axiom displays an error message. If there
is no need for an error check, you can call the operations
<a href="dbopqelt.xhtml">qelt</a> and
<a href="dbopqseteltbang.xhtml">qsetelt!</a> which provide the same
functionality but without the error check. Typically, these operations
are called in well-tested programs.

The operations <a href="dboprow.xhtml">row</a> and
<a href="dbopcolumn.xhtml">column</a> extract rows and columns,
respectively, and return objects of
<a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a> with the
same underlying element type.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p5','p7']);"
    value="row(arr,1)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p5','p8']);"
    value="column(arr,1)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>

You can determine the dimensions of an array by calling the operations
<a href="dbopnrows.xhtml">nrows</a> and <a href="dbopncols.xhtml">ncols</a>,
which return the number of rows and columns, respectively.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p9']);"
```

```
    value="nrows(arr)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p10']);"
    value="ncols(arr)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
To apply an operation to every element of an array, use
<a href="dbopmap.xhtml">map</a>. This creates a new array. This
expression negates every element.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11']);"
    value="map(-,arr)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
This creates an array where all the elements are doubled.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p2','p5','p12']);"
    value="map((x+->x+x),arr)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
To change the array destructively, use
<a href="dbopmapbang.xhtml">map!</a> instead of
<a href="dbopmap.xhtml">map</a>. If you need to make a copy of any array,
use <a href="dbopcopy.xhtml">copy</a>.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p1','p2','p5','p13']);"
    value="arrc:=copy(arr)" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p1','p2','p5','p13','p14']);"
    value="map!(-,arrc)" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p1','p2','p5','p13','p14','p15']);"
    value="arrc" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p1','p2','p5','p16']);"
    value="arrc:=copy(arr)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Use <a href="dbopmemberq.xhtml">member?</a> to see if a given element
is in an array.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p1','p2','p5','p17']);"
    value="member?(17,arr)" />
  <div id="ansp17"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="handleFree(['p1','p2','p5','p18']);"
    value="member?(10317,arr)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
To see how many times an element appears in an array, use
<a href="dbopcount.xhtml">count</a>.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p1','p2','p5','p19']);"
    value="count(17,arr)" />
  <div id="ansp19"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p1','p2','p5','p20']);"
    value="count(0,arr)" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>

For more information about the operations available for
<a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a>, issue
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="makeRequest('p21');"
    value=")show TwoDimensionalArray" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>
For more information on related topics, see
<a href="pagematrix.xhtml">Matrix</a> and
<a href="lin1darrays.xhtml">OneDimensionalArray</a>
\getchunk{page foot}
```

———-

man0page.xhtml

— man0page.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
Enter search string (use <b>*</b> for wild card unless counter-indicated):
  <form>
   <input type="text" name="searchbox" size="50"/>
  </form>

  <table>
   <tr>
    <td>
     <a href="(|kSearch| '|\\stringvalue{pattern}|).xhtml">
      <b>Constructors</b>
     </a>
    </td>
    <td>
     Search for
     <a href="(|cSearch| '|\\stringvalue{pattern}|)">
      <b>categories</b>
     </a>,
     <a href="(|dSearch| '|\\stringvalue{pattern}|)">
      <b>domains</b>
     </a>,
     or
     <a href="(|pSearch| '|\\stringvalue{pattern}|)">
      <b>packages</b>
     </a>
    </td>
   </tr>
   <tr>
    <td>
     <a href="(|oSearch| '|\\stringvalue{pattern}|).xhtml">
      <b>Operations</b>
     </a>
    </td>
    <td>Search for operations.</td>
   </tr>
   <tr>
    <td>
     <a href="(|aSearch| '|\\stringvalue{pattern}|).xhtml">
      <b>Attributes</b>
     </a>
    </td>
    <td>Search for attributes.</td>
   </tr>
   <tr>
```

```
  <td>
   <a href="(|aokSearch| '|\\stringvalue{pattern}|).xhtml">
    <b>General</b>
   </a>
  </td>
  <td>Search for all three of the above.</td>
 </tr>
 <tr>
  <td>
   <a href="(|docSearch| '|\\stringvalue{pattern}|).xhtml">
    <b>Documentation</b>
   </a>
  </td>
  <td>Search library documentation.
  </td>
 </tr>
 <tr>
  <td>
   <a href="(|genSearch| '|\\stringvalue{pattern}|).xhtml">
    <b>Complete</b>
   </a>
  </td>
  <td>All of the above.
  </td>
 </tr>
 <tr>
  <td>
   <a href="(|detailedSearch| '|\\stringvalue{pattern}|).xhtml">
    <b>Selectable</b>
   </a>
  </td>
  <td>Detailed search with selectable options.
  </td>
 </tr>
 <hr/>
 <tr>
  <td>
   <a href="htsearch \\stringvalue{pattern}.xhtml">
    <b>Reference</b>
   </a>
  </td>
  <td>Search Reference documentation (<b>*</b> wild card is not accepted).
  </td>
 </tr>
 <tr>
  <td>
   <a href="ugSysCmdPage.xhtml">
    <b>Commands</b>
   </a>
  </td>
  <td>View system command documentation.
  </td>
 </tr>
</table>
```

```
\getchunk{page foot}
```

———-

menualgebraadjointmatrix.xhtml

— menualgebraadjointmatrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebraadjointmatrix not implemented
\getchunk{page foot}
```

———-

menualgebraapplytolist.xhtml

— menualgebraapplytolist.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebraapplytolist not implemented
\getchunk{page foot}
```

———-

menualgebracharacteristicpolynomial.xhtml

— menualgebracharacteristicpolynomial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebracharacteristicpolynomial not implemented
\getchunk{page foot}
```

———-

menualgebradeterminant.xhtml

— menualgebradeterminant.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
```

```
\getchunk{page head}
menualgebradeterminant not implemented
\getchunk{page foot}
```

——

menualgebraeigenvalues.xhtml

— menualgebraeigenvalues.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebraeigenvalues not implemented
\getchunk{page foot}
```

——

menualgebraeigenvectors.xhtml

— menualgebraeigenvectors.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebraeigenvectors not implemented
\getchunk{page foot}
```

——

menualgebraentermatrix.xhtml

— menualgebraentermatrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebraentermatrix not implemented
\getchunk{page foot}
```

——

menualgebrainvertmatrix.xhtml

— menualgebrainvertmatrix.xhtml —

```
\getchunk{standard head}
```

```
 </head>
 <body>
\getchunk{page head}
menualgebrainvertmatrix not implemented
\getchunk{page foot}
```

———-

menualgebrageneratematrix.xhtml

— **menualgebrageneratematrix.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebrageneratematrix not implemented
\getchunk{page foot}
```

———-

menualgebramakelist.xhtml

— **menualgebramakelist.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebramakelist not implemented
\getchunk{page foot}
```

———-

menualgebramaptolist.xhtml

— **menualgebramaptolist.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebramaptolist not implemented
\getchunk{page foot}
```

———-

menualgebramaptomatrix.xhtml

— **menualgebramaptomatrix.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebramaptomatrix not implemented
\getchunk{page foot}
```

———-

menualgebrareducelist.xhtml

— **menualgebrareducelist.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebrareducelist not implemented
\getchunk{page foot}
```

———-

menualgebratransposematrix.xhtml

— **menualgebratransposematrix.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menualgebratransposematrix not implemented
\getchunk{page foot}
```

———-

menuaxiomaddtopath.xhtml

— **menuaxiomaddtopath.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomaddtopath not implemented
\getchunk{page foot}
```

———-

menuaxiomclearmemory.xhtml

— menuaxiomclearmemory.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomclearmemory not implemented
\getchunk{page foot}
```

menuaxiomdeletefunction.xhtml

— menuaxiomdeletefunction.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomdeletefunction not implemented
\getchunk{page foot}
```

menuaxiomdeletevariable.xhtml

— menuaxiomdeletevariable.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomdeletevariable not implemented
\getchunk{page foot}
```

menuaxiominterrupt.xhtml

— menuaxiominterrupt.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiominterrupt not implemented
\getchunk{page foot}
```

menuaxiomrestart.xhtml

— menuaxiomrestart.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomrestart not implemented
\getchunk{page foot}
```

menuaxiomshowdefinition.xhtml

— menuaxiomshowdefinition.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomshowdefinition not implemented
\getchunk{page foot}
```

menuaxiomdisplay.xhtml

— menuaxiomdisplay.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomdisplay not implemented
\getchunk{page foot}
```

menuaxiomset.xhtml

— menuaxiomset.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomset not implemented
\getchunk{page foot}
```

menuaxiomshowfunctions.xhtml

— menuaxiomshowfunctions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomshowfunctions not implemented
\getchunk{page foot}
```

menuaxiomshowvariables.xhtml

— menuaxiomshowvariables.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomshowvariables not implemented
\getchunk{page foot}
```

menuaxiomtoggletimedisplay.xhtml

— menuaxiomtoggletimedisplay.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuaxiomtoggletimedisplay not implemented
\getchunk{page foot}
```

menucalculuscalculussum.xhtml

— menucalculuscalculussum.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuscalculussum not implemented
\getchunk{page foot}
```

menucalculuscalculusproduct.xhtml

— menucalculuscalculusproduct.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuscalculusproduct not implemented
\getchunk{page foot}
```

menucalculuschangevariable.xhtml

— menucalculuschangevariable.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuschangevariable not implemented
\getchunk{page foot}
```

menucalculuscontinuedfractions.xhtml

— menucalculuscontinuedfractions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuscontinuedfractions not implemented
\getchunk{page foot}
```

menucalculusdifferentiate.xhtml

— menucalculusdifferentiate.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusdifferentiate not implemented
\getchunk{page foot}
```

menucalculusdividepolynomials.xhtml

— menucalculusdividepolynomials.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusdividepolynomials not implemented
\getchunk{page foot}
```

menucalculusfindlimit.xhtml

— menucalculusfindlimit.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusfindlimit not implemented
\getchunk{page foot}
```

menucalculusgetseries.xhtml

— menucalculusgetseries.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusgetseries not implemented
\getchunk{page foot}
```

menucalculusgreatestcommondivisor.xhtml

— menucalculusgreatestcommondivisor.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusgreatestcommondivisor not implemented
\getchunk{page foot}
```

menucalculusleastcommonmultiple.xhtml

— menucalculusleastcommonmultiple.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusleastcommonmultiple not implemented
\getchunk{page foot}
```

menucalculusintegrate.xhtml

— menucalculusintegrate.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusintegrate not implemented
\getchunk{page foot}
```

menucalculusinverselaplacetransform.xhtml

— menucalculusinverselaplacetransform.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusinverselaplacetransform not implemented
\getchunk{page foot}
```

menucalculuslaplacetransform.xhtml

— menucalculuslaplacetransform.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuslaplacetransform not implemented
\getchunk{page foot}
```

menucalculuslevel3.xhtml

— **menucalculuslevel3.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuslevel3 not implemented
\getchunk{page foot}
```

menucalculuslevel3a.xhtml

— **menucalculuslevel3a.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuslevel3a not implemented
\getchunk{page foot}
```

menucalculuslevel3b.xhtml

— **menucalculuslevel3b.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuslevel3b not implemented
\getchunk{page foot}
```

menucalculuslevel3c.xhtml

— **menucalculuslevel3c.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuslevel3c not implemented
\getchunk{page foot}
```

menucalculuspadeapproximation.xhtml

— menucalculuspadeapproximation.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuspadeapproximation not implemented
\getchunk{page foot}
```

menucalculuspartialfractions.xhtml

— menucalculuspartialfractions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculuspartialfractions not implemented
\getchunk{page foot}
```

menucalculusrischintegrate.xhtml

— menucalculusrischintegrate.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menucalculusrischintegrate not implemented
\getchunk{page foot}
```

menueditcopy.xhtml

— menueditcopy.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditcopy not implemented
\getchunk{page foot}
```

menueditcopyasimage.xhtml

— menueditcopyasimage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditcopyasimage not implemented
\getchunk{page foot}
```

———-

menueditcopytex.xhtml

— menueditcopytex.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditcopytex not implemented
\getchunk{page foot}
```

———-

menueditcopytext.xhtml

— menueditcopytext.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditcopytext not implemented
\getchunk{page foot}
```

———-

menueditcut.xhtml

— menueditcut.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditcut not implemented
\getchunk{page foot}
```

———-

menueditpaste.xhtml

— menueditpaste.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditpaste not implemented
\getchunk{page foot}
```

menueditdeleteselection.xhtml

— menueditdeleteselection.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditdeleteselection not implemented
\getchunk{page foot}
```

menueditselectiontoimage.xhtml

— menueditselectiontoimage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditselectiontoimage not implemented
\getchunk{page foot}
```

menueditselectiontoinput.xhtml

— menueditselectiontoinput.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menueditselectiontoinput not implemented
\getchunk{page foot}
```

menuequationsrealrootsofpolynmial.xhtml

— menuequationsrealrootsofpolynmial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationsrealrootsofpolynmial not implemented
\getchunk{page foot}
```

———-

menuequationsatvalue.xhtml

— menuequationsatvalue.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationsatvalue not implemented
\getchunk{page foot}
```

———-

menuequationsboundaryvalueproblem.xhtml

— menuequationsboundaryvalueproblem.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationsboundaryvalueproblem not implemented
\getchunk{page foot}
```

———-

menuequationsinitialvalueproblem1.xhtml

— menuequationsinitialvalueproblem1.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationsinitialvalueproblem1 not implemented
\getchunk{page foot}
```

———-

menuequationsinitialvalueproblem2.xhtml

— menuequationsinitialvalueproblem2.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationsinitialvalueproblem2 not implemented
\getchunk{page foot}
```

———–

menuequationssolvealgebraicsystem.xhtml

— menuequationssolvealgebraicsystem.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationssolvealgebraicsystem not implemented
\getchunk{page foot}
```

———–

menuequationseliminatevariable.xhtml

— menuequationseliminatevariable.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationseliminatevariable not implemented
\getchunk{page foot}
```

———–

menuequationssolvelinearsystem.xhtml

— menuequationssolvelinearsystem.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationssolvelinearsystem not implemented
\getchunk{page foot}
```

———–

menuequationssolveode.xhtml

— menuequationssolveode.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationssolveode not implemented
\getchunk{page foot}
```

menuequationssolveodewithlaplace.xhtml

— menuequationssolveodewithlaplace.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationssolveodewithlaplace not implemented
\getchunk{page foot}
```

menuequationsrootsofpolynomial.xhtml

— menuequationsrootsofpolynomial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationsrootsofpolynomial not implemented
\getchunk{page foot}
```

menuequationssolve.xhtml

— menuequationssolve.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationssolve not implemented
\getchunk{page foot}
```

menuequationssolvenumerically.xhtml

— menuequationssolvenumerically.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menuequationssolvenumerically not implemented
\getchunk{page foot}
```

———-

menufileexit.xhtml

— menufileexit.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufileexit not implemented
\getchunk{page foot}
```

———-

menufileinputfile.xhtml

— menufileinputfile.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufileinputfile not implemented
\getchunk{page foot}
```

———-

menufileloadlibrary.xhtml

— menufileloadlibrary.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufileloadlibrary not implemented
\getchunk{page foot}
```

———-

menufileopen.xhtml

— **menufileopen.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufileopen not implemented
\getchunk{page foot}
```

menufileprint.xhtml

— **menufileprint.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufileprint not implemented
\getchunk{page foot}
```

menufileread.xhtml

— **menufileread.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufileread not implemented
\getchunk{page foot}
```

menufilesave.xhtml

— **menufilesave.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufilesave not implemented
\getchunk{page foot}
```

menufilesaveas.xhtml

— menufilesaveas.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufilesaveas not implemented
\getchunk{page foot}
```

menufiletogglespool.xhtml

— menufiletogglespool.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menufiletogglespool not implemented
\getchunk{page foot}
```

menunumericsetprecision.xhtml

— menunumericsetprecision.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menunumericsetprecision not implemented
\getchunk{page foot}
```

menunumerictobigfloat.xhtml

— menunumerictobigfloat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menunumerictobigfloat not implemented
\getchunk{page foot}
```

menunumerictofloat.xhtml

— menunumerictofloat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menunumerictofloat not implemented
\getchunk{page foot}
```

———-

menunumerictogglenumericoutput.xhtml

— menunumerictogglenumericoutput.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menunumerictogglenumericoutput not implemented
\getchunk{page foot}
```

———-

menusimplifyaddalgebraicequality.xhtml

— menusimplifyaddalgebraicequality.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyaddalgebraicequality not implemented
\getchunk{page foot}
```

———-

menusimplifycomplexsimplification.xhtml

— menusimplifycomplexsimplification.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifycomplexsimplification not implemented
\getchunk{page foot}
```

———-

menusimplifycontractlogarithms.xhtml

— menusimplifycontractlogarithms.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifycontractlogarithms not implemented
\getchunk{page foot}
```

menusimplifyevaluatenounform.xhtml

— menusimplifyevaluatenounform.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyevaluatenounform not implemented
\getchunk{page foot}
```

menusimplifyexpandexpression.xhtml

— menusimplifyexpandexpression.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyexpandexpression not implemented
\getchunk{page foot}
```

menusimplifyexpandlogarithms.xhtml

— menusimplifyexpandlogarithms.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyexpandlogarithms not implemented
\getchunk{page foot}
```

menusimplifyfactorialsandgamma.xhtml

— **menusimplifyfactorialsandgamma.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyfactorialsandgamma not implemented
\getchunk{page foot}
```

menusimplifyfactorcomplex.xhtml

— **menusimplifyfactorcomplex.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyfactorcomplex not implemented
\getchunk{page foot}
```

menusimplifyfactorexpression.xhtml

— **menusimplifyfactorexpression.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifyfactorexpression not implemented
\getchunk{page foot}
```

menusimplifymoduluscomputation.xhtml

— **menusimplifymoduluscomputation.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifymoduluscomputation not implemented
\getchunk{page foot}
```

menusimplifysimplifyexpression.xhtml

— **menusimplifysimplifyexpression.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifysimplifyexpression not implemented
\getchunk{page foot}
```

menusimplifysubtitute.xhtml

— **menusimplifysubtitute.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifysubtitute not implemented
\getchunk{page foot}
```

menusimplifysimplifyradicals.xhtml

— **menusimplifysimplifyradicals.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifysimplifyradicals not implemented
\getchunk{page foot}
```

menusimplifytogglealgebraicflag.xhtml

— **menusimplifytogglealgebraicflag.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifytogglealgebraicflag not implemented
\getchunk{page foot}
```

menusimplifytrigsimplification.xhtml

— menusimplifytrigsimplification.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
menusimplifytrigsimplification not implemented
\getchunk{page foot}
```

———-

numbasicfunctions.xhtml

— numbasicfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Basic Functions</div>
  <hr/>
The size of an integer in Axiom is only limited by the amount of computer
storage you have available. The usual arithmetic operations are available.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="2^(5678-4856+2*17)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
There are a number of ways of working with the sign of an integer. Let's
use the x as an example.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="x:=-101" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
First of all, there is the absolute value function.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
```

```
    value="abs(x)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The <a href="dbopsign.xhtml">sign</a> operation returns -1 if its argument
is negative, 0 if zero and 1 if positive.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p2','p4']);"
    value="sign(x)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
You can determine if an integer is negative in several other ways.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p5']);"
    value="x &lt; 0" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p2','p6']);"
    value="x &lt;= -1" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p2','p7']);"
    value="negative?(x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Similarly, you can find out if it is positive.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p2','p8']);"
    value="x > 0" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p2','p9']);"
    value="x >= 1" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p2','p10']);"
    value="positive?(x)" />
  <div id="ansp10"><div></div></div>
```

```
 </li>
</ul>
This is the recommended way of determining whether an integer is zero.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p11']);"
    value="zero?(x)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
<hr/>
Use the <a href="dbopzeroq.xhtml">zero?</a> whenever you are testing any
mathematical object for equality with zero. This is usually more efficient
than using <a href="dbopequal.xhtml">=</a> (think of matrices: it is easier
to tell if a matrix is zero by just checking term by term than constructing
another "zero" amtrix and comparing the two matrices term by term) and also
avoids the problem that <a href="dbopequal.xhtml">=</a> is usually used
for creating equations.
<hr/>
This is the recommended way of determining whether an integer is equal to one.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p2','p12']);"
    value="one?(x)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
This syntax is used to test equality using <a href="dbopequal.xhtml">=</a>.
It says that you want a <a href="db.xhtml?Boolean">Boolean</a> (true or false)
answer rather than an equation.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p2','p13']);"
    value="(x=-101)@Boolean" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
The operations <a href="dbopoddq.xhtml">odd?</a> and
<a href="dbopevenq.xhtml">even?</a> determine whether an integer is odd
or even, respectively. They each return a
<a href="db.xhtml?Boolean">Boolean</a>
object.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p2','p14']);"
    value="odd?(x)" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
```

```
    onclick="handleFree(['p2','p15']);"
    value="even?(x)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
The operation <a href="dbopgcd.xhtml">gcd</a> computes the greatest common
divisor of two integers.
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="makeRequest('p16');"
    value="gcd(56788,43688)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
The operation <a href="dboplcm.xhtml">lcm</a> computes their least common
multiple.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="makeRequest('p17');"
    value="lcm(56788,43688)" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
To determine the maximum of two integers, use <a href="dbopmax.xhtml">max</a>.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest('p18');"
    value="max(678,567)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
To determine the minimum, use <a href="dbopmin.xhtml">min</a>.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="makeRequest('p20');"
    value="min(678,567)" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
The <a href="dbopreduce.xhtml">reduce</a> operation is used to extend
binary operations to more than two arguments. For example, you can use
<a href="dbopreduce.xhtml">reduce</a> to find the maximum integer in a
list or compute the least common multiple of all integers in a list.
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="makeRequest('p21');"
    value="reduce(max,[2,45,-89,78,100,-45])" />
  <div id="ansp21"><div></div></div>
 </li>
```

```
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="makeRequest('p22');"
    value="reduce(min,[2,45,-89,78,100,-45])" />
  <div id="ansp22"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p23" class="subbut"
    onclick="makeRequest('p23');"
    value="reduce(gcd,[2,45,-89,78,100,-45])" />
  <div id="ansp23"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p24" class="subbut"
    onclick="makeRequest('p24');"
    value="reduce(lcm,[2,45,-89,78,100,-45])" />
  <div id="ansp24"><div></div></div>
 </li>
</ul>
The infix operator "/" is not used to compute the quotient of integers.
Rather , it is used to create rational numbers as described in
<a href="numintegerfractions.xhtml">Fractions</a>.
<ul>
 <li>
  <input type="submit" id="p25" class="subbut"
    onclick="makeRequest('p25');"
    value="13/4" />
  <div id="ansp25"><div></div></div>
 </li>
</ul>
The infix operator <a href="dbopquo.xhtml">quo</a> computes the integer
quotient.
<ul>
 <li>
  <input type="submit" id="p26" class="subbut"
    onclick="makeRequest('p26');"
    value="13 quo 4" />
  <div id="ansp26"><div></div></div>
 </li>
</ul>
The infix operation <a href="dboprem.xhtml">rem</a> computes the integer
remainder.
<ul>
 <li>
  <input type="submit" id="p27" class="subbut"
    onclick="makeRequest('p27');"
    value="13 rem 4" />
  <div id="ansp27"><div></div></div>
 </li>
</ul>
One integer is evenly divisible by another if the remainder is zero.
The operation <a href="dbopexquo.xhtml">exquo</a> can also be used. See
<a href="axbook/section-2.5.xhtml">Unions</a> for an example.
<ul>
```

```
 <li>
  <input type="submit" id="p28" class="subbut"
    onclick="makeRequest('p28');"
    value="zero?(167604736446952 rem 2003644)" />
  <div id="ansp28"><div></div></div>
 </li>
</ul>
The operation <a href="dbopdivide.xhtml">divide</a> returns a record of
the quotient and remainder and thus is more efficient when both are needed.
<ul>
 <li>
  <input type="submit" id="p29" class="subbut"
    onclick="makeRequest('p29');"
    value="d:=divide(13,4)" />
  <div id="ansp29"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p30" class="subbut"
    onclick="handleFree(['p29','p30']);"
    value="d.quotient" />
  <div id="ansp30"><div></div></div>
 </li>
</ul>
Records are discussed in detail in
<a href="axbook/section-2.4.xhtml">Records</a>.
<ul>
 <li>
  <input type="submit" id="p31" class="subbut"
    onclick="handleFree(['p29','p31']);"
    value="d.remainder" />
  <div id="ansp31"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———

numberspage.xhtml

— numberspage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
The following types of numbers are among those available in Axiom
  <table>
   <tr>
    <td>
     <a href="numintegers.xhtml">Integers</a>
    </td>
    <td>
     Arithmetic with arbitrarily large integers
```

```
    </td>
   </tr>
   <tr>
    <td>
     <a href="numfractions.xhtml">Fractions</a>
    </td>
    <td>
     Rational numbers and general fractions
    </td>
   </tr>
   <tr>
    <td>
     <a href="nummachinefloats.xhtml">Machine Floats</a>
    </td>
    <td>
     Fixed precision machine floating point
    </td>
   </tr>
   <tr>
    <td>
     <a href="numfloat.xhtml">Real Numbers</a>
    </td>
    <td>
     Arbitrary precision decimal arithmetic
    </td>
   </tr>
   <tr>
    <td>
     <a href="numcomplexnumbers.xhtml">Complex Numbers</a>
    </td>
    <td>
     Complex numbers in general
    </td>
   </tr>
   <tr>
    <td>
     <a href="numfinitefields.xhtml">Finite Fields</a>
    </td>
    <td>
     Arithmetic in characteristic p
    </td>
   </tr>
  </table>
  <hr/>
Addtional topics
 <ul>
  <li> <a href="numnumericfunctions.xhtml">Numeric Functions</a></li>
  <li> <a href="numcardinalnumbers.xhtml">Cardinal Numbers</a></li>
  <li> <a href="nummachinesizedintegers.xhtml">Machine-sized Integers</a></li>
  <li> <a href="numromannumerals.xhtml">Roman Numerals</a></li>
  <li> <a href="numcontinuedfractions.xhtml">Continued Fractions</a></li>
  <li> <a href="numpartialfractions.xhtml">Partial Fractions</a></li>
  <li> <a href="numquaternions.xhtml">Quaternions</a></li>
  <li> <a href="numoctonions.xhtml">Octonions</a></li>
```

```
  <li> <a href="numrepeatingdecimals.xhtml">Repeating Decimals</a></li>
  <li> <a href="numrepeatingbinaryexpansions.xhtml">
        Repeating Binary Expansions
       </a>
  </li>
  <li> <a href="numrepeatinghexexpansions.xhtml">
        Repeating Hexadecimal Expansions
       </a>
  </li>
  <li> <a href="numotherbases.xhtml">Expansions in other Bases</a></li>
 </ul>
\getchunk{page foot}
```

numcardinalnumbers.xhtml

— numcardinalnumbers.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Cardinal Numbers</div>
  <hr/>
The <a href="dbopcardinalnumber.xhtml">CardinalNumber</a> can be used for
values indicating the cardinality of sets, both finite and infinite. For
example, the <a href="dbopdimension.xhtml">dimension</a> operation in the
category <a href="dbopvectorspace.xhtml">VectorSpace</a> returns a cardinal
number.

The non-negative integers have a natural construction as cardinals
<pre>
0=#{ }, 1={0}, 2={0,1}, ..., n={i | 0 &#60;= i &#60; n}
</pre>
The fact that 0 acts as a zero for the multiplication of cardinals is
equivalent to the axiom of choice.

Cardinal numbers can be created by conversion from non-negative integers.
 <ul>
  <li>
   <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
     value="c0:=0::CardinalNumber" />
   <div id="ansp1"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
     value="c1:=1::CardinalNumber" />
   <div id="ansp2"><div></div></div>
```

```
  </li>
  <li>
   <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
     value="c2:=2::CardinalNumber" />
   <div id="ansp3"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
     value="c3:=3::CardinalNumber" />
   <div id="ansp4"><div></div></div>
  </li>
 </ul>
The can also be obtained as the named cardinal Aleph(n)
 <ul>
  <li>
   <input type="submit" id="p5" class="subbut" onclick="makeRequest('p5');"
     value="A0:=Aleph 0" />
   <div id="ansp5"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p6" class="subbut" onclick="makeRequest('p6');"
     value="A1:=Aleph 1" />
   <div id="ansp6"><div></div></div>
  </li>
 </ul>
The <a href="dbopfiniteq.xhtml">finite?</a> operation tests whether a value
is a finite cardinal, that is, a non-negative integer.
 <ul>
  <li>
   <input type="submit" id="p7" class="subbut"
     onclick="handleFree(['p3','p7']);"
     value="finite? c2" />
   <div id="ansp7"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p8" class="subbut"
     onclick="handleFree(['p5','p8']);"
     value="finite? A0" />
   <div id="ansp8"><div></div></div>
  </li>
 </ul>
Similarly, the <a href="dbopcountableq.xhtml">countable?</a> operation
determines whether a value is a countable cardinal, that is, finite or
Aleph(0).
 <ul>
  <li>
   <input type="submit" id="p9" class="subbut"
     onclick="handleFree(['p3','p9']);"
     value="countable? c2" />
   <div id="ansp9"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p10" class="subbut"
     onclick="handleFree(['p5','p10']);"
```

```
     value="countable? A0" />
   <div id="ansp10"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p11" class="subbut"
     onclick="handleFree([’p6’,’p11’]);"
     value="countable? A1" />
   <div id="ansp11"><div></div></div>
  </li>
 </ul>
Arithmetic operations are defined on cardinal numbers as follows:
<table>
 <tr>
  <td>
   x+y = #(X+Y)
  </td>
  <td>
   cardinality of the disjoint union
  </td>
 </tr>
 <tr>
  <td>
   x-y = #(X-Y)
  </td>
  <td>
   cardinality of the relative complement
  </td>
 </tr>
 <tr>
  <td>
   x*y = #(X*Y)
  </td>
  <td>
   cardinality of the Cartesian product
  </td>
 </tr>
 <tr>
  <td>
   x+*y = #(X**Y)
  </td>
  <td>
   cardinality of the set of maps from Y to X
  </td>
 </tr>
</table>
Here are some arithmetic examples:
 <ul>
  <li>
   <input type="submit" id="p12" class="subbut"
     onclick="handleFree([’p3’,’p6’,’p12’]);"
     value="[c2+c2,c1+A1]" />
   <div id="ansp12"><div></div></div>
  </li>
  <li>
```

```
   <input type="submit" id="p13" class="subbut"
     onclick="handleFree(['p1','p2','p3','p5','p6','p13']);"
     value="[c0*c2,c1*c2,c2*c2,c0*A1,c1*A1,c2*A1,A0*A1]" />
   <div id="ansp13"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p14" class="subbut"
     onclick="handleFree(['p1','p2','p3','p6','p14']);"
     value="[c2**c0,c2**c1,c2**c2,A1**c0,A1**c1,A1**c2]" />
   <div id="ansp14"><div></div></div>
  </li>
 </ul>
Subtraction is a partial operation; it is not defined when subtracting
a larger cardinal from a smaller one, nor when subtracting two equal
infinite cardinals.
 <ul>
  <li>
   <input type="submit" id="p15" class="subbut"
     onclick="handleFree(['p2','p3','p4','p5','p6','p15']);"
     value="[c2-c1,c2-c2,c2-c3,A1-c2,A1-A0,A1-A1]" />
   <div id="ansp15"><div></div></div>
  </li>
 </ul>
The generalized continuum hypothesis asserts that
<pre>
 2**Aleph i = Aleph(i+1)
</pre>
and is independent of the axioms of set theory. (Goedel, The consistency
of the continuum hypothesis, Ann. Math. Studies, Princeton Univ. Press,
1940) The <a href="dbopcardinalnumber.xhtml">CardinalNumber</a> domain
provides an operation to assert whether the hypothesis is to be assumed.
 <ul>
  <li>
   <input type="submit" id="p16" class="subbut"
     onclick="makeRequest('p16');"
     value="generalizedContinuumHypothesisAssumed true" />
   <div id="ansp16"><div></div></div>
  </li>
 </ul>
When the generalized continuum hypothesis is assumed, exponentiation to
a transfinite power is allowed.
 <ul>
  <li>
   <input type="submit" id="p17" class="subbut"
     onclick="handleFree(['p1','p2','p3','p5','p6','p17']);"
     value="[c0**A0,c1**A0,c2**A0,A0**A0,A0**A1,A1**A0,A1**A1]" />
   <div id="ansp17"><div></div></div>
  </li>
 </ul>
Three commonly encountered cardinal numbers are
<pre>
  a = #Z                 countable infinity
  c = #R                 the continuum
  f = #{g|g: [0,1]->R}
```

```
</pre>
In this domain, these values are obtained under the generalized continuum
hypothesis in this way:
 <ul>
  <li>
   <input type="submit" id="p18" class="subbut"
     onclick="makeRequest('p18');"
     value="a:=Aleph 0" />
   <div id="ansp18"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p19" class="subbut"
     onclick="handleFree(['p18','p19']);"
     value="c:=2**a" />
   <div id="ansp19"><div></div></div>
  </li>
  <li>
   <input type="submit" id="p20" class="subbut"
     onclick="handleFree(['p18','p19','p20']);"
     value="f:=2**c" />
   <div id="ansp20"><div></div></div>
  </li>
 </ul>
\getchunk{page foot}
```

numcomplexnumbers.xhtml

— numcomplexnumbers.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
<div align="center">Complex Numbers</div>
<hr/>
The <a href="db.xhtml?Complex">Complex</a> constructor implements
complex objects over a commutative ring R. Typically, the ring R is
<a href="db.xhtml?Integer">Integer</a>,
<a href="dbfractioninteger.xhtml">Fraction Integer</a>,
<a href="db.xhtml?Float">Float</a>,
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>,
R can also be a symbolic type, like
<a href="dbpolynomialinteger.xhtml">Polynomial Integer</a>.
For more information about the numerical and graphical aspects of
complex numbers, see
<a href="axbook/book-contents.xhtml#chapter8">Numeric Functions</a>
in section 8.1.
```

```
Complex objects are created by the
<a href="dbcomplexcomplex.xhtml">complex</a> operation
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="a:=complex(4/3,5/2)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="b:=complex(4/3,-5/2)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The standard arithmetic operations are available.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="a+b" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p4']);"
    value="a-b" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p5']);"
    value="a*b" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
If R is a field, you can also divide the complex objects.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p6']);"
    value="a/b" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Use a conversion
(see <a href="axbook/section-2.7.xhtml">Conversion</a> in
section 2.7) to view the last object as a fraction of complex
integers.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p6','p7']);"
    value="%::Fraction Complex Integer" />
```

```
  <div id="ansp7"><div></div></div>
 </li>
</ul>
The predefined macro <tt>%i</tt> is defined to be complex(0,1).
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest(’p8’);"
    value="3.4+6.7*%i" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
You can also compute the
<a href="dbcomplexconjugate.xhtml">conjugate</a> and
<a href="dbcomplexnorm.xhtml">norm</a> of a complex number.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree([’p1’,’p9’]);"
    value="conjugate a" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree([’p1’,’p10’]);"
    value="norm a" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
The <a href="dbcomplexreal.xhtml">real</a> and
<a href="dbcompleximag.xhtml">imag</a> operations are provided to
extract the real and imaginary parts, respectively.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree([’p1’,’p11’]);"
    value="real a" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree([’p1’,’p12’]);"
    value="imag a" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
The domain
<a href="dbcomplexinteger.xhtml">Complex Integer</a>
is also called the Gaussian integers. If R is the integers (or, more
generally, a
<a href="db.xhtml?EuclideanDomain">Euclidean Domain</a>),
you can compute greatest common divisors.
<ul>
 <li>
```

```
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="gcd(12-12*%i,31+27*%i)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
You can also compute least common multiples
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="makeRequest('p14');"
    value="lcm(13-13*%i,31+27*%i)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
You can <a href="dbcomplexfactor.xhtml">factor</a> Gaussian integers.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="makeRequest('p15');"
    value="factor(13-13*%i)" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="makeRequest('p16');"
    value="factor complex(2,0)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

———-

numcontinuedfractions.xhtml

— numcontinuedfractions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Continued Fractions</div>
  <hr/>
Continued fractions have been a fascinating and useful tool in mathematics
for well over three hundred years. Axiom implements continued fractions
for fractions of any Euclidean domain. In practice, this usually means
rational numbers. In this section we demonstrate some of the operations
available for manipulating both finite and infinite continued fractions.
```

```
It may be helpful if you review
<a href="db.xhtml?Stream">Stream</a> to remind yourself of some of the
operations with streams.

The <a href="db.xhtml?ContinuedFraction">ContinuedFraction</a> domain is a
field and therefore you can add, subtract, multiply, and divide the
fractions. The
<a href="dbopcontinuedfraction.xhtml">continuedFraction</a> operation
converts its fractional argument to a continued fraction.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="c:=continuedFraction(314159/100000)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
This display is the compact form of the bulkier
<pre>
  3 +             1
     ---------------------------
     7 +            1
         -----------------------
         15 +         1
              ------------------
              1 +        1
                  --------------
                  25 +     1
                       ---------
                       1 +   1
                           -----
                           7 + 1
                               -
                               4
</pre>
You can write any rational number in a similar form. The fraction will
be finite and you can always take the "numerators" to be 1. That is, any
rational number can be written as a simple, finite continued fraction of
the form
<pre>
a(1) +            1
     ---------------------------
  a(2) +            1
         -----------------------
       a(3) +         1
                        .
                         .
                          .
                           1

              -----------------
              a(n-1) +     1
                       ---------
                          a(n)
</pre>
```

```
The a(i) are called partial quotients and the operation
<a href="dboppartialquotients.xhtml">partialQuotients</a> creates a
stream of them.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="partialQuotients c" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
By considering more and more of the fraction, you get the
<a href="dbopconvergents.xhtml">convergents</a>. For example, the
first convergent is a(1), the second is a(1)+1/a(2) and so on.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="convergents c" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Since this ia a finite continued fraction, the last convergent is the
original rational number, in reduced form. The result of
<a href="dbopapproximants.xhtml">approximants</a> is always an infinite
stream, though it may just repeat the "last" value.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="approximants c" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Inverting c only changes the partial quotients of its fraction by
inserting a 0 at the beginning of the list.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="pq:=partialQuotients(1/c)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Do this to recover the original continued fraction from this list of
partial quotients. The three argument form of the
<a href="dbopcontinuedfraction.xhtml">continuedFraction</a> operation takes
an element which is the whole part of the fraction, a stream of elements
which are the denominators of the fraction.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p5','p6']);"
    value="continuedFraction(first pq,repeating [1],rest pq)" />
```

```
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The streams need not be finite for
<a href="dbopcontinuedfraction.xhtml">continuedFraction</a>. Can you guess
which irrational number has the following continued fraction? See the end
of this section for the answer.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut" onclick="makeRequest('p7');"
    value="z:=continuedFraction(3,repeating [1],repeating [3,6])" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
In 1737 Euler discovered the infinite continued fraction expansion
<pre>
 e - 1             1
 ----- =  ---------------------------
p         2 +          1
             -----------------------
             6  +        1
                  ------------------
                 10 +       1
                      --------------
                      14 +  ...
</pre>
We use this expansion to compute rational and floating point
approximations of e. (For this and other interesting expansions,
see C. D. Olds, Continued Fractions, New Mathematical Library,
Random House, New York, 1963 pp.134-139).

By looking at the above expansion, we see that the whole part is 0
and the numerators are all equal to 1. This constructs the stream of
denominators.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut" onclick="makeRequest('p8');"
    value="dens:Stream Integer:=cons(1,generate((x+->x+4),6))" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Therefore this is the continued fraction expansion for (e-1)/2.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p8','p9']);"
    value="cf:=continuedFraction(0,repeating [1],dens)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
These are the rational number convergents.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
```

```
    onclick="handleFree(['p8','p9','p10']);"
    value="ccf:=convergents cf" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
You can get rational convergents for e by multiplying by 2 and adding 1.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p8','p9','p10','p11']);"
    value="eConvergents:=[2*e+1 for e in ccf]" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
You can also compute the floating point approximations to these convergents.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p8','p9','p10','p11','p12']);"
    value="eConvergents::Stream Float" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
Compare this to the value of e computed by the
<a href="dbopexp.xhtml">exp</a> operation in
<a href="db.xhtml?Float">Float</a>.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut" onclick="makeRequest('p13');"
    value="exp 1.0" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
In about 1658, Lord Brouncker established the following expansion for 4/pi.
<pre>
  1 +            1
     ---------------------------
     2 +           9
         -----------------------
         2  +        25
              ------------------
              2 +       49
                  --------------
                  2  +     81
                       ---------
                       2 +   ...
</pre>
Let's use this expansion to compute rational and floating point
approximations for pi.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut" onclick="makeRequest('p14');"
    value="cf:=continuedFraction(1,[(2*i+1)^2 for i in 0..],repeating [2])" />
  <div id="ansp14"><div></div></div>
```

```
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p14','p15']);"
    value="ccf:=convergents cf" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p14','p15','p16']);"
    value="piConvergents:=[4/p for p in ccf]" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
As you can see, the values are converging to
<pre>
  pi = 3.14159265358979323846..., but not very quickly.
</pre>
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p14','p15','p16','p17']);"
    value="piConvergents::Stream Float" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
You need not restrict yourself to continued fractions of integers. Here is
an expansion for a quotient of Gaussian integers.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut" onclick="makeRequest('p18');"
    value="continuedFraction((-122+597*%i)/(4-4*%i))" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
This is an expansion for a quotient of polynomials in one variable with
rational number coefficients.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut" onclick="makeRequest('p19');"
    value="r:Fraction UnivariatePolynomial(x,Fraction Integer)" />
  <div id="ansp19"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p19','p20']);"
    value="r:=((x-1)*(x-2))/((x-3)*(x-4))" />
  <div id="ansp20"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="handleFree(['p19','p20','p21']);"
    value="continuedFraction r" />
  <div id="ansp21"><div></div></div>
```

```
 </li>
</ul>
To conclude this section, we give you evidence that
<pre>
  z =  3 +            1
          ---------------------------
          3 +           1
              -----------------------
              6 +         1
                  -------------------
                   3 +       1
                        --------------
                        6  +    1
                             ---------
                             3 + ...
</pre>
is the expansion of the square root of 11.
<ul>
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="handleFree(['p7','p22']);"
    value="[i*i for i in convergents(z)]::Stream Float" />
  <div id="ansp22"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numexamples.xhtml

— numexamples.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Examples</div>
  <hr/>
One can show that if an integer of the form 2**k+1 is prime, then k
must be a power of two.

Pierre Fermat conjectured that every integer of the form 2**(2**n)+1
is prime. Let's look for a counterexample. First define a function:
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value="f(n:NNI):INT == 2^(2^n)+1" />
```

```
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Now try commands like:
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="factor f(1)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="factor f(2)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
until you find an integer of this form which is composite. You can also
try the following command:
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="[factor f(n) for n in 1..6]" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Obviously, Fermat didn't have access to Axiom.
\getchunk{page foot}
```

numfactorization.xhtml

— numfactorization.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Primes and Factorization</div>
  <hr/>
Use the operation <a href="dbopfactor.xhtml">factor</a> to factor integers.
It returns an object of type <a href="db.xhtml?Factored(Integer)">
Factored Integer</a>. See <a href="factored.xhtml">Factored</a> for a
discussion of the manipulation of factored objects.
<ul>
 <li>
```

```
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="factor 102400" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The operation <a href="dbopprimeq.xhtml">prime?</a> returns true or false
depending on whether its argument is a prime.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="prime? 7" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="prime? 8" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The operation <a href="dbopnextprime.xhtml">nextPrime</a> returns the
least prime number greater than its argument.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="nextPrime 100" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
The operation <a href="dbopprevprime.xhtml">prevPrime</a> returns the
greatest prime number less than its argument.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="prevPrime 100" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
To compute all primes between two integers (inclusively), use the
operation <a href="dbopprimes.xhtml">primes</a>.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="primes(100,175)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
You might sometimes want to see the factorization of an integer
when it is considered a Gaussian (that is, complex) integer. See
```

```
<a href="dbcomplexcomplex.xhtml">Complex</a> for more details.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="factor(2::Complex Integer)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numfinitefields.xhtml

— numfinitefields.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
<div align="center">Finite Fields</div>
<hr/>
A <sl>finite field</sl> (also called a <sl>Galois field</sl>) is a finite
algebraic structure where on can add, multiply, and divide under the same
laws (for example, commutativity, associativity, or distributivity) as
apply to the rational, real, or complex numbers. Unlike those three fields,
for any finite field there exists a positive prime integer p, called the
<a href="dbcharacteristic.xhtml">characteristic</a>, such that p*x=0 for
any element x in the finite field. In fact, the number of elements in a
finite filed is a power of the characteristic and for each prime p and
positive integer n there exists exactly one finite field with p**n elements,
up to an isomorphism. (For more information about the algebraic structure and
properties of finite fields, see for example, S. Lang <sl>Algebr</sl>,
Second Edition, New York, Addison-Wesley Publishing Company, Inc. 1984,
ISBN 0 201 05476 6; or R. Lidl, H. Niederreiter, <sl>Finite Fields</sl>,
Encyclopedia of Mathematics and Its Applications, Vol. 20, Cambridge.
Cambridge Univ. Press, 1983, ISBN 0 521 30240 4)

When n=1, the field has p elements and is called a <sl>prime field</sl>,
discussed in
<a href="axbook/section-8.11.xhtml#subsec-8.11.1">
Modular Arithmetic and Prime Fields</a>
in section 8.11.1. There are several ways of implementing extensions of
finite fields, and Axiom provides quite a bit of freedom to allow you to
choose the one that is best for your application. Moreover, we provide
operations for converting among the different representations of extensions
and different extensions of a single field. Finally, note that you usually
need to package call operations from finite fields if the operations do not
take as an argument an object of the field. See
<a href="">Package Calling and Target Types</a>
in section 2.9 for more information on package calling.
<ul>
```

```
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.1">
   Modular Arithmetic and Prime Fields
  </a>
 </li>
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.2">
   Extensions of Finite Fields
  </a>
 </li>
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.3">
   Irreducible Modulus Polynomial Representations
  </a>
 </li>
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.4">
   Cyclic Group Representations
  </a>
 </li>
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.5">
   Normal Basis Representations
  </a>
 </li>
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.6">
   Conversion Operations for Finite Fields
  </a>
 </li>
 <li>
  <a href="axbook/section-8.11.xhtml#subsec-8.11.7">
   Utility Operations for Finite Fields
  </a>
 </li>
</ul>
\getchunk{page foot}
```

numfloat.xhtml

— numfloat.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Real Numbers</div>
  <hr/>
Axiom provides two kinds of floating point numbers. The domain
<a href="db.xhtml?Float">Float</a>
(abbreviation <a href="db.xhtml?Float">FLOAT</a>)
```

```
implements a model of arbitrary precisions floating point numbers. The
domain
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>
(abbreviation <a href="db.xhtml?DoubleFloat">DFLOAT</a>)
is intended to make available hardware floating point arithmetic in Axiom.
The actual model of floating point
<a href="db.xhtml?DoubleFloat">DoubleFloat</a> that Axiom
provides is system dependent. For example, on the IBM System 370, Axiom
uses IBM double precision which has fourteen hexadecimal digits of
precision or roughly sixteen decimal digits. Arbitrary precision floats
allow the user to specify the precision at which arithmetic operations
are computed. Although this is an attractive facility, it comes at a cost.
Arbitrary precision floating point arithmetic typically takes twenty to
two hundred times more time than hardware floating point.

For more information about Axiom's numeric and graphic facilities
see <a href="axbook/book-contents.xhtml#chapter7">Graphics</a> in section 7,
<a href="axbook/book-contents.xhtml#chapter8">Numeric Functions</a>
in section 8.1, and <a href="nummachinefloats.xhtml">DoubleFloat</a>
<ul>
 <li>
  <a href="introtofloat.xhtml">Introduction to Float</a> (see
  <a href="axbook/section-9.27.xhtml#subsec-9.27.1">
   Jenks section 9.27.1
  </a>)
 </li>
 <li>
  <a href="conversionfunctions.xhtml">Conversion Functions</a> (see
  <a href="axbook/section-9.27.xhtml#subsec-9.27.2">
   Jenks section 9.27.2
  </a>)
 </li>
 <li>
  <a href="outputfunctions.xhtml">Output Functions</a> (see
  <a href="axbook/section-9.27.xhtml#subsec-9.27.3">
   Jenks, section 9.27.3
  </a>)
 </li>
 <li>
  <a href="determinantofhilbert.xhtml">
   An Example: Determinant of a Hilbert Matrix</a> (see
  <a href="axbook/section-9.27.xhtml#subsec-9.27.4">
   Jenks, section 9.27.4
  </a>)
 </li>
</ul>
\getchunk{page foot}
```

numfractions.xhtml

— numfractions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Fractions</div>
  <hr/>
Axiom handles fractions in many different contexts and will
automatically simplify fractions whenever possible. Here are
some examples:
 <ul>
  <li> <input type="submit" id="p1" value="1/4-1/5" class="subbut"
         onclick="makeRequest('p1');"/>
       <div id="ansp1"><div></div></div>
  </li>
  <li> <input type="submit" id="p2" value="f:=(x^2+1)/(x-1)" class="subbut"
         onclick="makeRequest('p2');"/>
       <div id="ansp2"><div></div></div>
  </li>
  <li> <input type="submit" id="p3" value="g:=(x^2-3*x+2)/(x+2)" class="subbut"
         onclick="makeRequest('p3');"/>
       <div id="ansp3"><div></div></div>
  </li>
  <li> <input type="submit" id="p4" value="f*g" class="subbut"
         onclick="handleFree(['p2','p3','p4']);"/>
       <div id="ansp4"><div></div></div>
  </li>
 </ul>
 <hr/>
Additional Topics:
 <table>
  <tr>
   <td>
    <a href="numrationalnumbers.xhtml">Rational Numbers</a>
   </td>
   <td>
    Quotients of integers
   </td>
  </tr>
  <tr>
   <td>
    <a href="numquotientfields.xhtml">Quotient Fields</a>
   </td>
   <td>
    Quotients over an arbitrary integral domain
   </td>
```

```
  </tr>
 </table>
\getchunk{page foot}
```

numfunctions.xhtml

— numfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Integer Number Theory Functions</div>
  <hr/>
The <a href="db.xhtml?IntegerNumberTheoryFunctions">
IntegerNumberTheoryFunctions</a> package contains a variety of
operations of interest to number theorists. Many of these operations
deal with divisibility properties of integers (Recall that an integer
a divides an integer b if there is an integer c such that b=a*c.)

The operation <a href="dbopdivisors.xhtml">divisors</a> returns a list
of the divisors of an integer
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="div144:=divisors(144)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can now compute the number of divisors of 144 and the sum of the
divisors of 144 by counting and summing the elements of the list we
just created.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="#(div144)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="reduce(+,div144)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
```

```
Of course, you can compute the number of divisors of an integer n,
usually denoted d(n), and the sum of the divisors of an integer n,
usually denoted &#x003C2;(n), without ever listing the divisors of n.

In Axiom, you can simply call the operations
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="numberOfDivisors(144)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="sumOfDivisors(144)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
The key is that d(n) and &#x003C2;(n) are "multiplicative functions".
This means that when n and m are relatively prime, that is, when n and
m have no factors in common, then d(nm)=d(n)d(m) and &#x003C2;(nm)=
&#x003C2;(n)&#x003C2;(m). Note that these functions are trivial to
compute when n is a prime power and are computed for general n from
the prime factorization of n. Other examples of multiplicative functions
are &#x003C2;_k(n), the sum of the k-th powers of the divisors of n and
&#x003C6;(n), the number of integers between 1 and n which are prime to n.
The corresponding Axiom operations are called
<a href="dbopsumofkthpowerdivisors.xhtml">sumOfKthPowerDivisors</a> and
<a href="dbopeulerphi.xhtml">eulerPhi</a>.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="sumOfKthPowerDivisors(144,2)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="eulerPhi(144)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>

An interesting function is called &#x003BC;(n), the Moebius mu function,
defined as
<pre>
        0 if n has a repeated prime factor
          (i.e. is divisible by a square)
  &#x003BC;(n)= 1 if n is 1
       (-1)^k if n is the product of k distinct primes
</pre>
The corresponding Axiom operation is
```

```
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="moebiusMu(2*2*2)" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="moebiusMu(1)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="moebiusMu(5*7*13)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
This function occurs in the following theorem:
<br/>
<b>Theorem</b>(Moebius Inversion Formula):<br/>
Let f(n) be a function on the positive integers and let F(n) be defined
by F(n)=sum of f(n) over d | n where the sum is taken over the positive
divisors of n. Then the values of f(n) can be recovered from the values
of F(n):f(n) = sum of &#x003BC;F(n/d) over d|n, where the sum is taken
over the positive divisors of n.

When f(n)=1, the F(n)=d(n). Thus, if you sum &#x003BC;(d)*d(n/d) over
the positive divisors of d of n, you should always get 1.
<ul>
 <li>
  <input type="submit" id="p11" class="noresult"
    onclick="makeRequest('p11');"
    value="f1(n)==reduce(+,[moebiusMu(d)*numberOfDivisors(quo(n,d)) for d in divisors(n)])" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p11','p12']);"
    value="f1(200)" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p11','p13']);"
    value="f1(846)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
Similarly, when f(n)=n, then F(n)=&#x003C2;(n). Thus, if you sum
&#x003BC;(d)*&#x003C2;(n/d) over the positive divisors d of n, you
should always get n.
```

```
<ul>
 <li>
  <input type="submit" id="p14" class="noresult"
    onclick="makeRequest('p14');"
    value="f2(n)==reduce(+,[moebiusMu(d)*sumOfDivisors(quo(n,d)) for d in divisors(n)])" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p14','p15']);"
    value="f2(200)" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p14','p16']);"
    value="f2(846)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
The Fibonacci numbers are defined by
<pre>
  F(1)=1
  F(2)=1
  F(n)=F(n-1)+F(n-2) for n=3,4,...
</pre>
The operation <a href="dbopfibonacci.xhtml">fibonacci</a> computes the
nth Fibonacci number.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="makeRequest('p17');"
    value="fibonacci(25)" />
  <div id="ansp17"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest('p18');"
    value="[fibonacci(n) for n in 1..15]" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
Fibonacci numbers can also be expressed as sums of binomial
coefficients.
<ul>
 <li>
  <input type="submit" id="p19" class="noresult"
    onclick="makeRequest('p19');"
    value="fib(n)==reduce(+,[binomial(n-1-k,k) for k in 0..quo(n-1,2)])" />
  <div id="ansp19"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p19','p20']);"
```

```
    value="fib(25)" />
  <div id="ansp20"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="handleFree(['p19','p21']);"
    value="[fib(n) for n in 1..15]" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>

Quadratic symbols can be computed with the operations
<a href="dboplegendre.xhtml">legendre</a> and
<a href="dbopjacobi.xhtml">jacobi</a>. The Legendre symbol (a/p) is
defined for integers a and p with p an odd prime number. By definition,
<pre>
        = -1 when a is not a square (mod p)
  (a/p) =  0 when a is divisible by p
        = +1 when a is a square (mod p)
</pre>
You compute (a/p) via the command legendre(a,p)
<ul>
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="makeRequest('p22');"
    value="legendre(3,5)" />
  <div id="ansp22"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p23" class="subbut"
    onclick="makeRequest('p23');"
    value="legendre(23,691)" />
  <div id="ansp23"><div></div></div>
 </li>
</ul>

The Jacobi symbol (a/n) is the usual extension of the Legendre symbol,
where n is an arbitrary integer. The most important property of the
Jacobi symbol is the following: if K is a quadratic field with
discriminant d and quadratic character &#x003C7;, the &#x003C7;(n)=(d/n).
Thus, you can use the Jacobi symbol to compute, say, the class numbers
of imaginary quadratic fields from a standard class number formula. This
function computes the class number of the imaginary quadratic field with
discriminant d.
<ul>
 <li>
  <input type="submit" id="p24" class="noresult"
    onclick="makeRequest('p24');"
    value="h(d)==quo(reduce(+,[jacobi(d,k) for k in 1..quo(-d,2)]),2-jacobi(d,2))" />
  <div id="ansp24"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p25" class="subbut"
    onclick="handleFree(['p24','p25']);"
```

```
    value="h(-163)" />
  <div id="ansp25"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p26" class="subbut"
    onclick="handleFree(['p24','p26']);"
    value="h(-499)" />
  <div id="ansp26"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p27" class="subbut"
    onclick="handleFree(['p24','p27']);"
    value="h(-1832)" />
  <div id="ansp27"><div></div></div>
 </li>
</ul>

\getchunk{page foot}
```

numgeneralinfo.xhtml

— numgeneralinfo.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">General Integer Information</div>
  <hr/>
Axiom provides many operations for manipulating arbitrary precision integers.
In this section we will show some of those that come from
<a href="db.xhtml?Integer">Integer</a> itself plus some that are implemented
in other packages. More examples of integers are in the following sections:
<a href="axbook/section-1.4.xhtml">Numbers</a>.
<a href="numfunctions.xhtml">IntegerNumberTheoryFunctions</a>,
<a href="numrepeatingdecimals.xhtml">DecimalExpansion</a>,
<a href="numrepeatingbinaryexpansions.xhtml">BinaryExpansion</a>,
<a href="numrepeatinghexexpansions.xhtml">HexadecimalExpansion</a>, and
<a href="numotherbases.xhtml">RadixExpansion</a>
<ul>
 <li> <a href="numbasicfunctions.xhtml">Basic Functions</a></li>
 <li> <a href="numbasicfunctions.xhtml">Primes and Factorization</a></li>
 <li> <a href="numnumbertheoreticfunctions.xhtml">
        Some Number Theoretic Functions</a></li>
</ul>
\getchunk{page foot}
```

numintegerfractions.xhtml

— numintegerfractions.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      numintegerfractions not implemented
\getchunk{page foot}
```

———-

numintegers.xhtml

— numintegers.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Integers</div>
  <hr/>
In Axiom, integers can be as large as you like. Try the following
examples.
 <ul>
  <li> <input type="submit" id="p1" value="x:=factorial(200)" class="subbut"
         onclick="makeRequest('p1');"/>
       <div id="ansp1"><div></div></div>
  </li>
  <li> <input type="submit" id="p2" value="y:=2^90-1" class="subbut"
         onclick="makeRequest('p2');"/>
       <div id="ansp2"><div></div></div>
  </li>
 </ul>
Of course, you can now do arithmetic as usual on these (very) large
integers:
 <ul>
  <li> <input type="submit" id="p3" value="x+y" class="subbut"
         onclick="handleFree(['p1','p2','p3']);"/>
       <div id="ansp3"><div></div></div>
  </li>
  <li> <input type="submit" id="p4" value="x-y" class="subbut"
         onclick="handleFree(['p1','p2','p4']);"/>
       <div id="ansp4"><div></div></div>
  </li>
  <li> <input type="submit" id="p5" value="x*y" class="subbut"
         onclick="handleFree(['p1','p2','p5']);"/>
```

```
       <div id="ansp5"><div></div></div>
  </li>
 </ul>
Axiom can factor integers, but numbers with small prime factors
 <ul>
  <li> <input type="submit" id="p6" value="factor(x)" class="subbut"
         onclick="handleFree(['p1','p6']);"/>
       <div id="ansp6"><div></div></div>
  </li>
 </ul>
will factor more rapidly than numbers with large prime factors.
 <ul>
  <li> <input type="submit" id="p7" value="factor(y)" class="subbut"
         onclick="handleFree(['p2','p7']);"/>
       <div id="ansp7"><div></div></div>
  </li>
 </ul>
 <hr/>
Additional topics
 <table>
  <tr>
   <td>
    <a href="numgeneralinfo.xhtml">General Info</a>
   </td>
   <td>
    General information and examples of integers
   </td>
  </tr>
  <tr>
   <td>
    <a href="numfactorization.xhtml">Factorization</a>
   </td>
   <td>
    Primes and factorization
   </td>
  </tr>
  <tr>
   <td>
    <a href="numfunctions.xhtml">Functions</a>
   </td>
   <td>
    Number theoretic functions
   </td>
  </tr>
  <tr>
   <td>
    <a href="numexamples.xhtml">Examples</a>
   </td>
   <td>
    Examples from number theory
   </td>
  </tr>
  <tr>
   <td>
```

```
    <a href="numproblems.xhtml">Problems</a>
   </td>
   <td>
    Problems from number theory
   </td>
  </tr>
 </table>
\getchunk{page foot}
```

nummachinefloats.xhtml

— **nummachinefloats.xhtml** —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Machine Floats</div>
  <hr/>
Axiom provides two kinds of floating point numbers. The domain
<a href="db.xhtml?Float">Float</a>
(abbreviation <a href="db.xhtml?Float">FLOAT</a>)
implements a model of arbitrary precisions floating point numbers. The
domain
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>
(abbreviation <a href="db.xhtml?DoubleFloat">DFLOAT</a>)
is intended to make available hardware floating point arithmetic in Axiom.
The actual model of floating point
<a href="db.xhtml?DoubleFloat">DoubleFloat</a> that Axiom
provides is system dependent. For example, on the IBM System 370, Axiom
uses IBM double precision which has fourteen hexadecimal digits of
precision or roughly sixteen decimal digits. Arbitrary precision floats
allow the user to specify the precision at which arithmetic operations
are computed. Although this is an attractive facility, it comes at a cost.
Arbitrary precision floating point arithmetic typically takes twenty to
two hundred times more time than hardware floating point.

By default, floating point numbers that you enter into Axiom are of type
<a href="db.xhtml?Float">Float</a>.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="2.71828" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
```

```
You must therefore tell Axiom that you want to use
<a href="db.xhtml?DoubleFloat">DoubleFloat</a> values and operations. The
following are some conservative guidelines for getting Axiom to use
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>.

To get a value of type <a href="db.xhtml?DoubleFloat">DoubleFloat</a>.,
use a target with
    "@", ...
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="2.71828@DoubleFloat" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
a conversion,...
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="2.71828::DoubleFloat" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
or an assignment to a declared variable. It is more efficient if you
use a target rather than an explicit or implicit conversion.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="eApprox:DoubleFloat:=2.71828" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
You also need to declare functions that work with
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>.
<ul>
 <li>
  <input type="submit" id="p5" class="noresult"
    onclick="makeRequest('p5');"
    value="avg:List DoubleFloat -> DoubleFloat" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="noresult"
    onclick="makeRequest('p6');"
    value="avg l==(empty? l => 0::DoubleFloat; reduce(_+,l)/#l)"/>
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p5','p6','p7']);"
    value="avg []" />
```

```
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p5','p6','p8']);"
    value="avg [3.4,9.7,-6.8]" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Use package calling for operations from
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>
unless the arguments themselves are already of type
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="cos(3.1415926)$DoubleFloat" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="cos(3.1415926)::DoubleFloat" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
By far, the most common usage of
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>
is for functions to be graphied. For more information about Axiom's
numerical and graphical facilities, see
<a href="axbook/book-contents.xhtml#chapter7">Graphics</a>
in section 7,
<a href="axbook/book-contents.xhtml#chapter8">Numeric Functions</a>
in section 8.1, and
<a href="numfloat.xhtml">Float</a>

The usual arithmetic and elementary functions are available for
<a href="db.xhtml?DoubleFloat">DoubleFloat</a>. Use
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="showcall('p11');"
   value=")show DoubleFloat"/>
  <div id="ansp11"><div></div></div>
 </li>
</ul>
to get a list of operations.
\getchunk{page foot}
```

nummachinesizedintegers.xhtml

— nummachinesizedintegers.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
<div align="center">Machine-sized Integers</div>
<hr/>
The <a href="db.xhtml?SingleInteger">SingleInteger</a> is intended to
provide support in Axiom for machine integer arithmetic. It is generally
much faster than (bignum) <a href="db.xhtml?Integer">Integer</a> arithmetic
but suffers from a limited range of values. Since Axiom can be implemented
on top of various dialects of Lisp, the actual representation of small
integers may not correspond exactly to the host machines integer
representation.

You can discover the minimum and maximum values in your implementation by
using <a href="dbopmin.xhtml">min</a> and <a href="dbopmax.xhtml">max</a>
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="min()$SingleInteger" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="max()$SingleInteger" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
To avoid confusion with <a href="db.xhtml?Integer">Integer</a>, which is
the default type for integers, you usually need to work with declared
variables (see <a href="axbook/section-2.3.xhtml">Declarations</a>).
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="a:=1234::SingleInteger" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
or use package calling (see
<a href="axbook/section-2.9.xhtml">Package Calling and Target Types</a>).
<ul>
 <li>
  <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
    value="b:=1234$SingleInteger" />
  <div id="ansp4"><div></div></div>
```

```
 </li>
</ul>
You can add, multiply, and subtract
<a href="db.xhtml?SingleInteger">SingleInteger</a> objects, and ask for the
greatest common divisor
(<a href="dbopgcd.xhtml">gcd</a>).
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p4','p5']);"
    value="gcd(a,b)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
The least common multiple
(<a href="dboplcm.xhtml">lcm</a>) is also available.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p3','p4','p6']);"
    value="lcm(a,b)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Operations
<a href="dbopmulmod.xhtml">mulmod</a>,
<a href="dbopaddmod.xhtml">addmod</a>,
<a href="dbopsubmod.xhtml">submod</a>, and
<a href="dbopinvmod.xhtml">invmod</a>
are similar -- they provide arithmetic modulo a given small integer.
Here is 5*6 mod 13.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut" onclick="makeRequest('p7');"
    value="mulmod(5,6,13)$SingleInteger" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
To reduce a small integer modulo a prime, use
<a href="dboppositiveremainder.xhtml">positiveRemainder</a>
<ul>
 <li>
  <input type="submit" id="p8" class="subbut" onclick="makeRequest('p8');"
    value="positiveRemainder(37,13)$SingleInteger" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Operations <a href="dbopsingleintegerand.xhtml">And</a>,
<a href="dbopsingleintegeror.xhtml">Or</a>,
<a href="dbopsingleintegerxor.xhtml">xor</a>,
and <a href="dbopsingleintegernot.xhtml">Not</a>
provide bit level operations on small integers.
<ul>
 <li>
```

```
  <input type="submit" id="p9" class="subbut" onclick="makeRequest('p9');"
    value="And(3,4)$SingleInteger" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Use shift(int,numToShift) to shift bits, where int is shifted left if
numToShift is positive, right if negative.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut" onclick="makeRequest('p10');"
    value="shift(1,4)$SingleInteger" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut" onclick="makeRequest('p11');"
    value="shift(31,-1)$SingleInteger" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
Many other operations are available for small integers, including many of
those provided for <a href="db.xhtml?Integer">Integer</a>.
To see other operations use the system command
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="showcall('p12');"
   value=")show SingleInteger"/>
  <div id="ansp12"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numnumbertheoreticfunctions.xhtml

— numnumbertheoreticfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Some Number Theoretic Functions</div>
  <hr/>
Axiom provides several number theoretic operations for integers.
More examples are in
<a href="numfunctions.xhtml">IntegerNumberTheoryFunctions</a>,

The operation <a href="dbopfibonacci.xhtml">fibonacci</a> computes the
```

```
Fibonacci numbers. The algorithm has a running time O(log(n)**3) for
argument n.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="[fibonacci(k) for k in 0..]" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The operation <a href="dboplegendre.xhtml">legendre</a> computes the
Legendre symbol for its two integer arguments where the second one is
prime. If you know the second argument to be prime, use
<a href="dbopjacobi.xhtml">jacobi</a> instead where no check is made.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="[legendre(i,11) for i in 0..10]" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The operation <a href="dbopjacobi.xhtml">jacobi</a> computes the Jacobi symbol
for its two integer arguments. By convention, 0 is returned if the greatest
common divisor of the numerator and denominator is not 1.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="[jacobi(i,15) for i in 0..9]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The operation <a href="dbopeulerphi.xhtml">eulerPhi</a> compute the values
of Euler's &#x003C6;-function where &#x003C6;(n) equals the number of
positive integers less than or equal to n that are relatively prime to the
positive integer n.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="[eulerPhi i for i in 1..]" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
The operation <a href="dbopmoebiusmu.xhtml">moebiusMu</a> computes the
Moebius &#x003BC; function.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="[moebiusMu i for i in 1..]" />
  <div id="ansp5"><div></div></div>
 </li>
```

```
</ul>
Although they have somewhat limited utility, Axiom provides Roman numerals.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="a:=roman(78)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="b:=roman(87)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p6','p7','p8']);"
    value="a+b" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p6','p7','p9']);"
    value="a*b" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p6','p7','p10']);"
    value="b rem a" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numnumericfunctions.xhtml

— numnumericfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Numeric Functions</div>
  <hr/>
Axiom provides two basic floating point types:
```

```
<a href="numfloat.xhtml">Float</a> and
<a href="nummachinefloats.xhtml">DoubleFloat</a>. This section
describes how to use numerical operations defined on these types and
the related complex types. As we mentioned in
<a href="axbook/book-contents.xhtml#chapter1">An Overview of Axiom</a>
in chapter 1., the
<a href="numfloat.xhtml">Float</a> type is a software implementation of
floating point numbers in which the exponent and the significand may have
any number of digits. See
<a href="numfloat.xhtml">Float</a> for detailed information about this
domain. The
<a href="nummachinefloats.xhtml">DoubleFloat</a> is usually a hardware
implementation of floating point numbers, corresponding to machine double
precision. The types
<a href="dbcomplexfloat.xhtml">Complex Float</a> and
<a href="dbcomplexdoublefloat.xhtml">Complex DoubleFloat</a> are the
corresponding software implementations of complex floating point numbers.
In this section the term floating point type means any of these four
types. The floating point types immplement the basic elementary functions.
These include (where $ means
<a href="nummachinefloats.xhtml">DoubleFloat</a>,
<a href="numfloat.xhtml">Float</a>,
<a href="dbcomplexfloat.xhtml">Complex Float</a>,
<a href="dbcomplexdoublefloat.xhtml">Complex DoubleFloat</a>):<br/>
<a href="dbopexp.xhtml">exp</a>,
<a href="dboplog.xhtml">log</a>: $ -> $<br/>
<a href="dbopsin.xhtml">sin</a>,
<a href="dbopcos.xhtml">cos</a>,
<a href="dboptan.xhtml">tan</a>,
<a href="dbopcot.xhtml">cot</a>,
<a href="dbopsec.xhtml">sec</a>,
<a href="dbopcsc.xhtml">csc</a>: $ -> $<br/>
<a href="dbopasin.xhtml">asin</a>,
<a href="dbopacos.xhtml">acos</a>,
<a href="dbopatan.xhtml">atan</a>,
<a href="dbopacot.xhtml">acot</a>,
<a href="dbopasec.xhtml">asec</a>,
<a href="dbopacsc.xhtml">acsc</a>: $ -> $<br/>
<a href="dbopsinh.xhtml">sinh</a>,
<a href="dbopcosh.xhtml">cosh</a>,
<a href="dboptanh.xhtml">tanh</a>,
<a href="dbopcoth.xhtml">coth</a>,
<a href="dbopsech.xhtml">sech</a>,
<a href="dbopcsch.xhtml">csch</a>: $ -> $<br/>
<a href="dbopasinh.xhtml">asinh</a>,
<a href="dbopacosh.xhtml">acosh</a>,
<a href="dbopatanh.xhtml">atanh</a>,
<a href="dbopacoth.xhtml">acoth</a>,
<a href="dbopasech.xhtml">asech</a>,
<a href="dbopacsch.xhtml">acsch</a>: $ -> $<br/>
<a href="dboppi.xhtml">pi</a>: () -> $<br/>
<a href="dbopsqrt.xhtml">sqrt</a>: $ -> $<br/>
<a href="dbopnthroot.xhtml">nthRoot</a>: ($,Integer) -> $<br/>
<a href="dbopstarstar.xhtml">**</a>: ($,Fraction Integer) -> $<br/>
```

```
<a href="dbopstarstar.xhtml">**</a>: ($,$) -> $<br/>
The handling of roots depends on whether the floating point type is
real or complex: for the real floating point types,
<a href="nummachinefloats.xhtml">DoubleFloat</a> and
<a href="numfloat.xhtml">Float</a>, if a real root exists the one with
the same sign as the radicand is returned; for the complex floating
point types, the principal value is returned. Also, for real floating
point types the inverse functions produce errors if the results are not
real. This includes cases such as asin(1.2), log(-3.2), sqrt(-1,1).
The default floating point type is <a href="numfloat.xhtml">Float</a>
or <a href="dbcomplexfloat.xhtml">Complex Float</a>, just use normal
decimal notation.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="exp(3.1)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="exp(3.1+4.5*%i)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
To evaluate functions using
<a href="nummachinefloats.xhtml">DoubleFloat</a> or
<a href="dbcomplexdoublefloat.xhtml">Complex DoubleFloat</a>, a
declaration or conversion is required.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="(r:DFLOAT:=3.1; t:DFLOAT:=4.5; exp(r+t*%i))" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
    value="exp(3.1::DFLOAT+4.5::DFLOAT*%i)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
A number of special functions are provided by the package
<a href="db.xhtml?DoubleFloatSpecialFunctions">DoubleFloatSpecialFunctions</a>
for the machine precision floating point types. The special functions
provided are listed below, where F stands for the types
<a href="numfloat.xhtml">Float</a>
or <a href="dbcomplexfloat.xhtml">Complex Float</a>. The real versions
of the functions yield an error if the result is not real.
<ul>
 <li>
  <a href="dbopgamma.xhtml">Gamma</a>: F -> F<br/>
  Gamma(z) is the Euler gamma
  function, Gamma(Z), defined by<br/>
  Gamma(z) = integrate(t^(z-1)*exp(-t),t=0..%infinity)
 </li>
```

```
<li>
 <a href="dbopbeta.xhtml">Beta</a>: F -> F<br/>
 Beta(u,v) is the Euler Beta
 function B(u,v), defined by <br/>
 Beta(u,v)=integrate(t^(u-1)*(1-t)^(b-1),t=0..1)<br/>
 This is related to Gamma(z) by<br/>
 Beta(u,v)=Gamma(u)*Gamma(v)/Gamma(u+v)
</li>
<li>
 <a href="dboploggamma.xhtml">logGamma</a>: F -> F<br/>
 logGamma(z) is the natural logarithm of Gamma(z). This can often be
 computed even if Gamma(z) cannot.
</li>
<li>
 <a href="dbopdigamma.xhtml">digamma</a>: F -> F<br/>
 digamma(z), also called psi(z), is the function psi(z), defined by<br/>
 psi(z)=Gamma'(z)/Gamma(z)
</li>
<li>
<a href="dboppolygamma.xhtml">polygamma</a>: (NonNegativeInteger, F) -> F<br/>
 polygamma(n,z) is the n-th derivative of digamma(z)
</li>
<li>
 <a href="dbopbesselj.xhtml">besselJ</a>: (F, F) -> F<br/>
 besselJ(v,z) is the Bessel function of the first kind, J(v,z). This
 function satisfies the differential equation<br/>
 z^(2w)''(z)+zw'(z)+(z^2-v^2)w(z)=0
</li>
<li>
 <a href="dbopbessely.xhtml">besselY</a>: (F, F) -> F<br/>
 besselY(v,z) is the Bessel function of the second kind, Y(v,z). This
 function satisfies the same differential equation as
 <a href="dbopbesselj.xhtml">besselJ</a>. The implementation simply
 uses the relation<br/>
 Y(v,z)=(J(v,z)cos(v*%pi)-J(-v,z))/sin(v*%pi)
</li>
<li>
 <a href="dbopbesseli.xhtml">besselI</a>: (F, F) -> F<br/>
 besselI(v,z) if the modifed Bessel function of the first kind, I(v,z).
 This function satisfies the differential equation<br/>
 z^2w''(z)+zw'(z)-(z^2+v^2)w(z)=0
</li>
<li>
 <a href="dbopbesselk.xhtml">besselK</a>: (F, F) -> F<br/>
 besselK(v,z) is the modifed Bessel function of the second kind, K(v,z).
 This function satisfies the same differential equation as
 <a href="dbopbesseli.xhtml">besselI</a>. The implementation simply uses
 the relation<br/>
 K(v,z)=%pi*(I(v,z)-I(-v,z))/(2sin(v*%pi))
</li>
<li>
 <a href="dbopairyai.xhtml">airyAi</a>: F -> F<br/>
 airyAi(z) is the Airy function Ai(z). This function satisfies the
 differential equation<br/>
```

```
  w''(z)-zw(z)=0<br/>
  The implementation simply uses the relation<br/>
  Ai(-z)=1/3*sqrt(z)*(J(-1/3,2/3*z^(3/2))+J(1/3,2/3*z^(3/2)))
 </li>
 <li>
  <a href="dbopairybi.xhtml">airyBi</a>: F -> F<br/>
  airyBi(z) is the Airy function Bi(z). This function satisfies the
  same differential equation as airyAi.
  The implementation simply uses the relation<br/>
  Bi(-z)=1/3*sqrt(3*z)*(J(-1/3,2/3*z^(3/2))-J(1/3,2/3*z^(3/2)))
 </li>
 <li>
  <a href="dbophypergeometric0f1.xhtml">hypergeometric0F1</a>: (F, F) -> F<br/>
  hypergeometric0F1(c,z) is the hypergeometric function 0F1(;c;z). The above
  special functions are defined only for small floating point types. If you
  give <a href="numfloat.xhtml">Float</a> arguments, they are converted to
  <a href="nummachinefloats.xhtml">DoubleFloat</a> by Axiom.
 </li>
 <li>
  <input type="submit" id="p5" class="subbut" onclick="makeRequest('p5');"
    value="Gamma(0.5)^2" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut" onclick="makeRequest('p6');"
    value="(a:=2.1; b:=1.1; besselI(a+%i*b,b*a+1))" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
A number of additional operations may be used to compute numerical
values. These are special polynomial functions that can be evaluated
for values in any commutative ring R, and in particular for values in
any floating-point type. The following operations are provided by the
package <a href="db.xhtml?OrthogonalPolynomialFunctions">
OrthogonalPolynomialFunctions</a>:
<ul>
 <li> <a href="dbopchebyshevt.xhtml">chebyshevT</a>:
      (nonNegativeInteger,R) -> R
   <br/>
      chebyshevT(n,z) is the nth Chebyshev polynomial of the first kind,
      T[n](z). These are defined by
   <br/>
      (1-t*z)/(1-2*t*z*t**2)=sum(T[n](z)*t**n,n=0..)
 </li>
 <li> <a href="dbopchebyshevu.xhtml">chebyshevU</a>:
      (nonNegativeInteger,R) -> R
   <br/>
      chebyshevU(n,z) is the nth Chebyshev polynomial of the second kind,
      U[n](z). These are defined by
   <br/>
     1/(1-2*t*z+t**2)=sum(U[n](z)*t**n,n=0..)
 </li>
 <li> <a href="dbophermiteh.xhtml">hermiteH</a>:
      (NonNegativeInteger,R) -> R
```

```
   <br/>
      hermiteH(n,z) is the nth Hermite polynomial, H[n](z). These are
      defined by
   <br/>
      exp(2*t*z-t**2)=sum(H[n](z)*t**n/n!,n=0..)
 </li>
 <li> <a href="dboplaguerrel.xhtml">laguerreL</a>:
      (NonNegativeInteger,R) -> R
   <br/>
       laguerreL(n,z) is the nth Laguerre polynomial, L[n](z). These are
       defined by
   <br/>
       (exp(-t*z/(1-t))/(1-t)=sum(L[n](z)*t**n/n!,n=0..)
 </li>
 <li> <a href="dboplaguerrel.xhtml">laguerreL</a>:
      (NonNegativeInteger,NonNegativeInteger,R) -> R
   <br/>
      labuerreL(m,n,2) is the associated Laguerre polynomial, L&lt;m>[n](z).
      This is the nth derivative of L[n](z).
 </li>
 <li> <a href="dboplegendrep.xhtml">legendreP</a>:
      (NonNegativeInteger,R) -> R
   <br/>
      legendreP(n,z) is the nth Legendre polynomial, P[n](z). These are
      defined by
   <br/>
    1/sqrt(1-2*z*t+t**2)=sum(P[n](z)*t**n,n=0..)
 </li>
</ul>
<br/>
<br/>
These operations require non-negative integers for the indices,
but otherwise the argument can be given as desired.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="[chebyshevT(i,z) for i in 0..5]" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
The expression chebyshevT(n,z) evaluates to the nth Chebyshev polynomial
of the first kind.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="chebyshevT(3,5.0+6.0*%i)" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="chebyshevT(3,5.0::DoubleFloat)" />
```

```
  <div id="ansp9"><div></div></div>
 </li>
</ul>
The expression chebyshevU(n,z) evaluates to the nth Chebyshev polynomial
of the second kind.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="[chebyshevU(i,z) for i in 0..5]" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="makeRequest('p11');"
    value="chebyshevU(3,0.2)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
The expression hermiteH(n,z) evaluates to the nth Hermite polynomial.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="makeRequest('p12');"
    value="[hermiteH(i,z) for i in 0..5]" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="makeRequest('p13');"
    value="hermiteH(100,1.0)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
The expression laguerreL(n,z) evaluates to the nth Laguerre polynomial.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="makeRequest('p14');"
    value="[laguerreL(i,z) for i in 0..4]" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="makeRequest('p15');"
    value="laguerreL(4,1.2)" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="makeRequest('p16');"
    value="[laguerreL(j,3,z) for j in 0..4]" />
  <div id="ansp16"><div></div></div>
 </li>
```

```
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="makeRequest('p17');"
    value="laguerreL(1,3,2.1)" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
The expression legendreP(n,z) evaluates to the nth Legendre polynomial.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest('p18');"
    value="[legendreP(i,z) for i in 0..5]" />
  <div id="ansp18"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="makeRequest('p19');"
    value="legendreP(3,3.0*%i)" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
<br/>
<br/>
Finally, three number-theoretic polynomial operations may be evaluated.
The following operations are provided by the package
<a href="db.xhtml?NumberTheoreticPolynomialFunctions">
NumberTheoreticPolynomialFunctions</a>.
<ul>
 <li> <a href="dbopbernoullib.xhtml">bernoulliB</a>:
      (NonNegativeInteger,R) -> R
   <br/>
      bernoulliB(n,z) is the nth Bernoulli polynomial, B[n](z). These are
      defined by
   <br/>
      t*exp(z*t)/(exp t - 1)=sum(B[n](z)*t**n/n! for n=0..)
 </li>
 <li> <a href="dbopeulere.xhtml">eulerE</a>:
      (NonNegativeInteger,R) -> R
   <br/>
      eulerE(n,z) is the nth Euler polynomial, E[n](z). These are defined by
   <br/>
      2*exp(z*t)/(exp t + 1)=sum(E[n](z)*t**n/n! for n=0..)
 </li>
 <li> <a href="dbopcyclotomic.xhtml">cyclotomic</a>:
      (NonNegativeInteger,R) -> R
   <br/>
      cyclotomic(n,z) is the nth cyclotomic polynomial &#x003C6;(n,z).
      This is the polynomial whose roots are precisely the primitive nth
      roots of unity. This polynomial has degree given by the Euler
      totient function &#x003C6;(n).
 </li>
</ul>
```

```
The expression bernoulliB(n,z) evaluates to the nth Bernoulli polynomial.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="makeRequest('p20');"
    value="bernoulliB(3,z)" />
  <div id="ansp20"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="makeRequest('p21');"
    value="bernoulliB(3,0.7+0.4*%i)" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>
The expression eulerE(n,z) evaluates to the nth Euler polynomial.
<ul>
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="makeRequest('p22');"
    value="eulerE(3,z)" />
  <div id="ansp22"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p23" class="subbut"
    onclick="makeRequest('p23');"
    value="eulerE(3,0.7+0.4*%i)" />
  <div id="ansp23"><div></div></div>
 </li>
</ul>
The expression cyclotomic(n,z) evaluates to the nth cyclotomic polynomial.
<ul>
 <li>
  <input type="submit" id="p24" class="subbut"
    onclick="makeRequest('p24');"
    value="cyclotomic(3,z)" />
  <div id="ansp24"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p25" class="subbut"
    onclick="makeRequest('p25');"
    value="cyclotomic(3,(-1.0+0.0*%i)**(2/3))" />
  <div id="ansp25"><div></div></div>
 </li>
</ul>
<br/>
<br/>
Drawing complex functions in Axiom is presently somewhat awkward compared
to drawing real functions. It is necessary to use the
<a href="dbopdraw.xhtml">draw</a> operations that operate on functions
rather than expressions.

This is the complex exponential function. When this is displayed in color,
the height is the value of the real part of the function and the color is
```

the imaginary part. Red indicates large negative imaginary values, green indicates imaginary values near zero and blue/violet indicates large positive imaginary values.

```
<ul>
 <li>
  <input type="submit" id="p26" class="subbut"
    onclick="makeRequest('p26');"
    value='draw((x,y)+->real exp complex(x,y),-2..2,-2*%pi..2*%pi,colorFunction==(x,y)+->imag exp complex(x,y),t
  <div id="ansp26"><div></div></div>
 </li>
</ul>
```

This is the complex arctangent function. Again, the height is the real part of the function value but here the color indicates the function value's phase. The position of the branch cuts are clearly visible and one can see that the function is real only for a real argument.

```
<ul>
 <li>
  <input type="submit" id="p27" class="subbut"
    onclick="makeRequest('p27');"
    value='vp:=draw((x,y)+->real atan complex(x,y),-%pi..%pi,-%pi..%pi,colorFunction==(x,y)+->argument atan comp
  <div id="ansp27"><div></div></div>
 </li>
</ul>
```

This is the complex Gamma function.

```
<ul>
 <li>
  <input type="submit" id="p28" class="subbut"
    onclick="makeRequest('p28');"
    value='draw((x,y)+->max(min(real Gamma complex(x,y),4),-4),-%pi..%pi,-%pi..%pi,style=="shade",colorFunction=
  <div id="ansp28"><div></div></div>
 </li>
</ul>
```

This shows the real Beta function near the origin.

```
<ul>
 <li>
  <input type="submit" id="p29" class="subbut"
    onclick="makeRequest('p29');"
    value='draw(Beta(x,y)/100,x=-1.6..1.7,y=-1.6..1.7,style=="shade",title=="Beta(x,y)",var1Steps==40,var2Steps=
  <div id="ansp29"><div></div></div>
 </li>
</ul>
```

This is the Bessel function J(alpha,x) for index alpha in the range -6..4 and argument x in the range 2..14.

```
<ul>
 <li>
  <input type="submit" id="p30" class="subbut"
    onclick="makeRequest('p30');"
    value='draw((alpha,x)+->min(max(besselJ(alpha,x+8),-6), 6),-6..4,-6..6,title=="besselJ(alpha,x)",style=="sha
  <div id="ansp30"><div></div></div>
 </li>
</ul>
```

This is the modified Bessel function I(alpha,x) evaluated for various real values of the index alpha and fixed argument x=5.

```
<ul>
```

```
 <li>
  <input type="submit" id="p31" class="subbut"
    onclick="makeRequest('p31');"
    value="draw(besselI(alpha,5),alpha=-12..12,unit==[5,20])" />
  <div id="ansp31"><div></div></div>
 </li>
</ul>
This is similar to the last example except the index alpha takes on complex
values in a 6x6 rectangle centered on the origin.
<ul>
 <li>
  <input type="submit" id="p32" class="subbut"
    onclick="makeRequest('p32');"
    value='draw((x,y)+->real besselI(complex(x/20,y/20),5),-60..60,-60..60,colorFunction==(x,y)+->argument besse
  <div id="ansp32"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numoctonions.xhtml

— numoctonions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Octonions</div>
  <hr/>
The Octonions, also called the Cayley-Dixon algebra, defined over a
commutative ring are an eight-dimensional non-associative algebra. Their
construction from quaternions is similar to the construction of quaternions
from complex numbers (see <a href="numquaternions.xhtml">Quaternion</a>).
As <a href="db.xhtml?Octonion">Octonion</a> creates an eight-dimensional
algebra, you have to give eight components to construct an octonion.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="oci1:=octon(1,2,3,4,5,6,7,8)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="oci2:=octon(7,2,3,-4,5,6,-7,0)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
```

```
Or you can use two quaternions to create an octonion.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="oci3:=octon(quatern(-7,-12,3,-10),quatern(5,6,9,0))" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
You can easily demonstrate the non-associativity of multiplication.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="(oci1*oci2)*oci3-oci1*(oci2*oci3)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
As with the quaternions, we have a real part, the imaginary parts i, j,
k, and four additional imaginary parts E, I, J, and K. These parts
correspond to the canonical basis (1,i,j,k,E,I,J,K). For each basis
element there is a component operation to extract the coefficient of
the basis element for a given octonion.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
value="[real oci1, imagi oci1, imagj oci1, imagk oci1,
imagE oci1, imagI oci1, imagJ oci1, imagK oci1]"/>
  <div id="ansp5"><div></div></div>
 </li>
</ul>
A basis with respect to the quaternions is given by (1,E). However, you
might ask, what then are the commuting rules? To answer this, we create
some generic elements. We do this in Axim by simply changing the ground
ring from
<a href="db.xhtml?Integer">Integer</a> to
<a href="dbpolynomialinteger.xhtml">Polynomial Integer</a>.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut" onclick="makeRequest('p6');"
    value="q:Quaternion Polynomial Integer:=quatern(q1,qi,qj,qk)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut" onclick="makeRequest('p7');"
    value="E:Octonion Polynomial Integer:=octon(0,0,0,0,1,0,0,0)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Note that quaternions are automatically converted to octonions in the
obvious way.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
```

```
    onclick="handleFree(['p6','p7','p8']);"
    value="q*E" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p6','p7','p9']);"
    value="E*q" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p6','p10']);"
    value="q*1$(Octonion Polynomial Integer)" />
  <div id="ansp10"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p6','p11']);"
    value="1$(Octonion Polynomial Integer)*q" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
Finally, we check that the <a href="dbopnorm.xhtml">norm</a>, defined as
the sum of the squares of the coefficients, is a multiplicative map.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut" onclick="makeRequest('p12');"
    value="o:Octonion Polynomial Integer:=octon(o1,oi,oj,ok,oE,oI,oJ,oK)" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p12','p13']);"
    value="norm o" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut" onclick="makeRequest('p14');"
    value="p:Octonion Polynomial Integer:=octon(p1,pi,pj,pk,pE,pI,pJ,pK)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
Since the result is 0, the norm is multiplicative
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p12','p14','p15']);"
    value="norm(o*p)-norm(o)*norm(p)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
Issue the system command
<ul>
```

```
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="showcall('p16');"
   value=")show Octonion"/>
  <div id="ansp16"><div></div></div>
 </li>
</ul>
to display the list of operations defined by
<a href="db.xhtml?Octonion">Octonion</a>.
\getchunk{page foot}
```

numotherbases.xhtml

— numotherbases.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Expansions in other Bases</div>
  <hr/>
It is possible to expand numbers in general bases. Here we expand
111 in base 5. This means
<pre>
    2   1   0      2    1  -
  10 +10 +10  = 4*5 +2*5 +5
</pre>
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="111::RadixExpansion(5)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can expand fractions to form repeating expansions.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="(5/24)::RadixExpansion(2)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="(5/24)::RadixExpansion(3)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
    value="(5/24)::RadixExpansion(8)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut" onclick="makeRequest('p5');"
    value="(5/24)::RadixExpansion(10)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
For bases from 11 to 36 the letters A through Z are used.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut" onclick="makeRequest('p6');"
    value="(5/24)::RadixExpansion(12)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut" onclick="makeRequest('p7');"
    value="(5/24)::RadixExpansion(16)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut" onclick="makeRequest('p8');"
    value="(5/24)::RadixExpansion(36)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
For bases greater than 36, the ragits are separated by blanks.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut" onclick="makeRequest('p9');"
    value="(5/24)::RadixExpansion(38)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
The <a href="db.xhtml?RadixExpansion">RadixExpansion</a> type provides
operations to obtain the individual ragits. Here is a rational number
in base 8.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut" onclick="makeRequest('p10');"
    value="a:=(76543/210)::RadixExpansion(8)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
The operation <a href="dbopwholeragits.xhtml">wholeRagits</a> returns
a list of the ragits for the integral part of the number.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p10','p11']);"
    value="w:=wholeRagits a" />
  <div id="ansp11"><div></div></div>
```

```
 </li>
</ul>
The operations <a href="dbopprefixragits.xhtml">prefixRagits</a> and
<a href="dbopcycleragits.xhtml">cycleRagits</a> returns lists of the
initial and repeating ragist in the fractional part of the number.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p10','p12']);"
    value="f0:=prefixRagits a" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p10','p13']);"
    value="f1:=cycleRagits a" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
You can construct any radix expansion by giving the whole, prefix, and
cycle parts. The declaration is necessary to let Axiom know the base
of the ragits.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p11','p12','p13','p14']);"
    value="u:RadixExpansion(8):=wholeRadix(w)+fractRadix(f0,f1)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
If there is no repeating part, then the list [0] should be used.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut" onclick="makeRequest('p15');"
    value="v:RadixExpansion(12):=fractRadix([1,2,3,11],[0])" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
If you are not interested in the repeating nature of the expansion,
an infinite stream of ragits can be obtained using
<a href="dbopfractragits.xhtml">fractRagits</a>
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
     onclick="handleFree(['p14','p16']);"
    value="fractRagits(u)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Of course, it's possible to recover the fraction representation:n
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p10','p17']);"
```

```
    value="a::Fraction(Integer)" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
Issue the system command
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="showcall('p18');"
   value=")show RadixExpansion"/>
  <div id="ansp18"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?RadixExpansion">RadixExpansion</a>. More examples of
expansions are available in
<a href="numrepeatingdecimals.xhtml">DecimalExpansion</a>,
<a href="numrepeatingbinaryexpansions.xhtml">BinaryExpansion</a>, and
<a href="numrepeatinghexexpansions.xhtml">HexadecimalExpansion</a>
\getchunk{page foot}
```

——

numpartialfractions.xhtml

— numpartialfractions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Partial Fractions</div>
  <hr/>
A partial fraction is a decomposition of a quotient into a sum of quotients
where the denominators of the summand are powers of primes. (Most people
first encounter partial fractions when they are learning integral calculus.
For a technical discussion of partial fractions see, for example, Lang's
Algebra.) For example, the rational number 1/6 is decomposed into 1/2-1/3.
You can compute partial fractions of quotients of objects from domains
belonging to the category
<a href="db.xhtml?EuclideanDomain">EuclideanDomain</a>. For example,
<a href="db.xhtml?Integer">Integer</a>,
<a href="dbcomplexinteger.xhtml">Complex Integer</a>, and
<a href="db.xhtml?UnivariatePolynomial">
UnivariatePolynomial(x,Fraction Integer)</a>
all belong to
<a href="db.xhtml?EuclideanDomain">EuclideanDomain</a>.
In the examples following, we demonstrate how to decompose quotients of
each of these kinds of objects into partial fractions.
```

```
It is necessary that we know how to factor the denominator when we want to
compute a partial fraction. Although the interpreter can often do this
automatically, it may be necessary for you to include a call to
<a href="dbopfactor.xhtml">factor</a>. In these examples, it is not
necessary to factor the denominators explicitly. The main operation for
computing partial fractions is called
<a href="dboppartialfraction.xhtml">partialFraction</a> and we use this
to compute a decomposition of 1/10!. The first argument top
<a href="dboppartialfraction.xhtml">partialFraction</a> is the numerator
of the quotient and the second argument is the factored denominator.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="partialFraction(1,factorial 10)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Since the denominators are powers of primes, it may be possible to expand
the numerators further with respect to those primes. Use the operation
<a href="dboppadicfraction.xhtml">padicFraction</a> to do this.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="f:=padicFraction(%)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The operation <a href="dbopcompactfraction.xhtml">compactFraction</a>
returns an expanded fraction into the usual form. The compacted version
is used internally for computational efficiency.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="compactFraction(f)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
You can add, subtract, multiply, and divide partial fractions. In addition,
you can extract the parts of the decomposition.
<a href="dbopnumberoffractionalterms.xhtml">numberOfFractionalTerms</a>
computes the number of terms in the fractional part. This does not include
the whole part of the fraction, which you get by calling
<a href="dbopwholepart.xhtml">wholePart</a>. In this example, the whole part
is 0.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p2','p4']);"
    value="numberOfFractionalTerms(f)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
```

```
The operation
<a href="dbopnthfractionalterm.xhtml">nthFractionalTerm</a>
returns the individual terms in the decomposition. Notice that the object
returned is a partial fraction itself.
<a href="dbopfirstnumer.xhtml">firstNumer</a> and
<a href="dbopfirstdenom.xhtml">firstDenom</a> extract the numerator and
denominator of the first term of the fraction.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p5']);"
    value="nthFractionalTerm(f,3)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Given two gaussian integers (see <a href="db.xhtml?Complex">Complex</a>),
you can decompose their quotient into a partial fraction.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut" onclick="makeRequest('p6');"
    value="g:=partialFraction(1,-13+14*%i)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
To convert back to a quotient, simply use the conversion
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value="g::Fraction Complex Integer" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
To conclude this section, we compute the decomposition of
<pre>
                  1
     -------------------------------
                  2       3       4
     (x + 1)(x + 2) (a + 3) (x + 4)
</pre>
The polynomials in this object have type
<a href="db.xhtml?UnivariatePolynomial">
UnivariatePolynomial(x,Fraction Integer)</a>.
We use the <a href="dbopprimefactor.xhtml">primeFactor</a> operation
(see <a href="db.xhtml?Factored">Factored</a>) to create the denominator
in factored form directly.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut" onclick="makeRequest('p8');"
   value="u:FR UP(x,FRAC INT):=reduce(*,[primeFactor(x+i,i) for i in 1..4])"/>
  <div id="ansp8"><div></div></div>
 </li>
</ul>
These are the compact and expanded partial fractions for the quotient.
```

```
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p8','p9']);"
    value="pu:=partialFraction(1,u)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p8','p9','p10']);"
    value="padicFraction pu" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
Also see
<a href="db.xhtml?FullPartialFractionExpansion">
FullPartialFractionExpansion</a> for examples of factor-free conversion of
quotients to full partial fractions.

Issue the system
command
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="showcall('p11');"
   value=")show PartialFraction"/>
  <div id="ansp11"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?PartialFraction">PartialFraction</a>.

\getchunk{page foot}
```

numproblems.xhtml

— numproblems.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Problems</div>
  <hr/>
One can show that if an integer of the form 2**k-1 is prime then
k must be prime.
<br/>
```

```
<b>Proof</b>
Suppose that k=m*n is a non-trivial factorization. Then
<pre>
       2^m = 1 (mod (2^m-1))
   2^(m*n) = 1 (mod (2^m-1))
  so 2^m-1 is a non-trivial factor of 2^k-1
</pre>

<b>Problem</b> Find the smallest prime p such that 2**p-1 is not prime
<br/>
<b>Answer</b>
<br/>
First, define a function:
<ul>
 <li>
  <input type="submit" id="p1" class="noresult"
    onclick="makeRequest('p1');"
    value="f(n:NNI):INT == 2^n-1" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can try factoring f(p) as p ranges through the set of primes.
For example,
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="factor f(7)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
This gets tedious after a while, so let's use Axiom's stream facility.
A streamm is essentially an infinite sequence. First, we create a stream
consisting of the positive integers:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="ints:=[n for n in 1..]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Now, we create a stream consisting of the primes:
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p2','p4']);"
    value="primes:=[x for x in ints | prime? x]" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Here is the 25th prime:
<ul>
 <li>
```

```
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p5']);"
    value="primes.25" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Next, create the stream of numbers of the form 2**p-1 with p prime:
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p6']);"
    value="numbers:=[f(n) for n in primes]" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Finally, form the stream of factorizations of the elements of numbers:
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p6','p7']);"
    value="factors:=[factor n for n in numbers]" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
You can see that the fifth number in the stream (2047=23*89) is the first
one that has a non-trivial factorization. Since 2**11=2048, the solution
to the problem is 11.

Here is another way to see that 2047 is the first number in the stream
that is composite:
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p3','p4','p6','p8']);"
    value="nums:=[x for x in numbers | not prime? x]" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
<br/><br/>
<b>Problem</b>: Find the smallest positive integer n such that
n**2-n+41 is not prime.
<br/>
<b>Answer</b>: When n=41, n**2-n+41=41**2, which certainly isn't prime.
Is there any smaller integer that works? Here are the first 40 values:
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="numbs:=[n**2-n+41 for n in 0..40]" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Now have Axiom factor the numbers on this list:
<ul>
```

```
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p9','p10']);"
    value="[factor n for n in numbs]" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
You can see that 41 is the smallest positive integer n such that
n**2-n+41 is not prime.
\getchunk{page foot}
```

numquaternions.xhtml

— numquaternions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Quaternions</div>
  <hr/>
The domain contructor <a href="db.xhtml?Quaternion">Quaternion</a>
implements quaternions over commutative rings.

The basic operation for creating quaternions is
<a href="dbopquatern.xhtml">quatern</a>. This is a quaternion
over the rational numbers.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="q:=quatern(2/11,-8,3/4,1)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The four arguments are the real part, the i imaginary part,
the j imaginary part, and the k imaginary part, respectively.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="[real q, imagI q, imagJ q, imagK q]" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Because q is over the rationals (and nonzero), you can invert it.
<ul>
 <li>
```

```
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="inv q" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The usual arithmetic (ring) operations are available.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="q^6" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"  onclick="makeRequest('p5');"
    value="r:=quatern(-2,3,23/9,-89)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p5','p6']);"
    value="q+r" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
In general, multiplication is not commutative.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p5','p7']);"
    value="q*r-r*q" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
There are no predefined constants for the imaginary i, j, and k parts,
but you can easily define them
<ul>
 <li>
  <input type="submit" id="p8" class="subbut" onclick="makeRequest('p8');"
    value="i:=quatern(0,1,0,0)" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut" onclick="makeRequest('p9');"
    value="j:=quatern(0,0,1,0)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut" onclick="makeRequest('p10');"
    value="k:=quatern(0,0,0,1)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
```

```
These satisfy the normal identities.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p8','p9','p10','p11']);"
    value="[i*i,j*j,k*k,i*j,j*k,k*i,q*i]" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
The norm is the quaternion times its conjugate.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p12']);"
    value="norm q" />
  <div id="ansp12"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p1','p13']);"
    value="c:=conjugate q" />
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p1','p13','p14']);"
    value="q*c" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
For information on
related topics, see <a href="db.xhtml?Complex">Complex</a> and
<a href="db.xhtml?Octonion">Octonion</a>. You can also issue the
system command
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="showcall('p15');"
   value=")show Quaternion"/>
  <div id="ansp15"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?Quaternion">Quaternion</a>.
\getchunk{page foot}
```

numquotientfields.xhtml

— numquotientfields.xhtml —

```
\getchunk{standard head}
```

```
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Quotient Fields</div>
  <hr/>
The <a href="db.xhtml?Fraction">Fraction</a> domain implements quotients.
The elements must belong to a domain of category
<a href="db.xhtml?IntegralDomain">IntegralDomain</a>: multiplication
must be commutative and the product of two non-zero elements must not be
zero. This allows you to make fractions of most things you would think of,
but don't expect to create a fraction of two matrices. The abbreviation for
<a href="db.xhtml?Fraction">Fraction</a> is
<a href="db.xhtml?Fraction">FRAC</a>.

Use <a href="dbopdivide.xhtml">/</a> to create a fraction.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="a:=11/12" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="b:=23/24" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The standard arithmetic operations are available.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="3-a*b^2+a+b/a" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Extract the numerator and denominator by using
<a href="dbopnumer.xhtml">numer</a> and <a href="dbopdenom.xhtml">denom</a>,
respectively.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="numer(a)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
```

```
    onclick="handleFree(['p2','p5']);"
    value="denom(b)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Operations like
<a href="dbopmax.xhtml">max</a>, <a href="dbopmin.xhtml">min</a>,
<a href="dbopnegativeq.xhtml">negative?</a>,
<a href="dboppositiveq.xhtml">positive?</a>, and
<a href="dbopzeroq.xhtml">zero?</a> are all available if they are
provided for the numerators and denominators. See
<a href="numintegers.xhtml?Integer">Integer</a> for examples.

Don't expect a useful answer from
<a href="dbopfactor.xhtml">factor</a>,
<a href="dbopgcd.xhtml">gcd</a>, or
<a href="dboplcm.xhtml">lcm</a> if you apply them to fractions.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="r:=(x^2+2*x+1)/(x^2-2*x+1)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Since all non-zero fractions are invertible, these operations have
trivial definitions.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value="factor(r)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Use <a href="dbopmap.xhtml">map</a> to apply
<a href="dbopfactor.xhtml">factor</a> to the numerator and denominator,
which is probably what you mean.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p6','p8']);"
    value="map(factor,r)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>

Other forms of fractions are available, Use
<a href="dbopcontinuedfraction.xhtml">continuedFraction</a>
to create a continued fraction.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
```

```
    value="continuedFraction(7/12)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
Use <a href="dboppartialfraction.xhtml">partialFraction</a> to create a
partial fraction.
See <a href="numcontinuedfractions.xhtml">continuedFraction</a>
and <a href="numpartialfractions.xhtml">PartialFraction</a> for
additional information and examples.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="makeRequest('p10');"
    value="partialFraction(7,12)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
Use conversion to create alternative views of fractions with objects
moved in and out of the numerator and denominator.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="makeRequest('p11');"
    value="g:=2/3+4/5*%i" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
Conversion is discussed in detail in
<a href="axbook/section-2.7.xhtml">Conversion</a>.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p11','p12']);"
    value="g::FRAC COMPLEX INT" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numrationalnumbers.xhtml

— numrationalnumbers.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
```

```
  <div align="center">Rational Numbers</div>
  <hr/>
Like integers, rational numbers can be arbitrarily large. For example:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="61657^10/999983^12" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Rational numbers will not be converted to decimals unless you explicitly
ask Axiom to do so. To convert a rational number to a decimal, use the
function <a href="dbopnumeric.xhtml">numeric</a>. Here's an example:
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="x:=104348/33215" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p2','p3']);"
    value="numeric x" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
You can find the numerator and denominator of rational numbers using the
functions <a href="dbopnumer.xhtml">numer</a> and
<a href="dbopdenom.xhtml">denom</a>, respectively.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p2','p4']);"
    value="numer(x)" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p2','p5']);"
    value="denom(x)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
To factor the numerator and denominator of a fraction, use the following
command:
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p2','p6']);"
    value="factor(numer x)/factor(denom x)" />
  <div id="ansp6"><div></div></div>
 </li>
```

```
</ul>
\getchunk{page foot}
```

numrepeatingbinaryexpansions.xhtml

— numrepeatingbinaryexpansions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Repeating Binary Expansions</div>
  <hr/>
All rational numbers have repeating binary expansions. Operations to
access the individual bits of a binary expansion can be obtained by
converting the value to
<a href="db.xhtml?RadixExpansion">RadixExpansion(2)</a>. More examples
of expansions are available in
<a href="numrepeatingdecimals.xhtml">DecimalExpansion</a>,
<a href="numrepeatinghexexpansions.xhtml">HexadecimalExpansion</a>, and
<a href="db.xhtml?RadixExpansion">RadixExpansion</a>.

The expansion (of type
<a href="db.xhtml?BinaryExpansion">BinaryExpansion</a>)
of a rational number is returned by the
<a href="dbopbinary.xhtml">binary</a> operation.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="r:=binary(22/7)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Arithmetic is exact.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="r+binary(6/7)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The period of the expansion can be short or long...
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="[binary(1/i) for i in 102..106]" />
```

```
  <div id="ansp3"><div></div></div>
 </li>
</ul>
or very long
<ul>
 <li>
  <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
    value="binary(1/1007)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
These numbers are bona fide algebraic objects.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut" onclick="makeRequest('p5');"
    value="p:=binary(1/4)*x^2+binary(2/3)*x+binary(4/9)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p5','p6']);"
    value="q:=D(p,x)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p5','p6','p7']);"
    value="g:=gcd(p,q)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

numrepeatingdecimals.xhtml

— numrepeatingdecimals.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Repeating Decimals</div>
  <hr/>
All rationals have repeating decimal expansions. Operations to access
the individual digits of a decimal expansion can be obtained by converting
the value to <a href="db.xhtml?RadixExpansion">RadixExpansion(10)</a>.
```

```
The operation <a href="dbopdecimal.xhtml">decimal</a> is used to create
this expansion of type
<a href="db.xhtml?DecimalExpansion">DecimalExpansion</a>.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="r:=decimal(22/7)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Arithmetic is exact.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="r+decimal(6/7)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The period of the expansion can be short or long...
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="[decimal(1/i) for i in 350..354]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
or very long
<ul>
 <li>
  <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
    value="decimal(1/2049)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
These numbers are bona fide algebraic objects.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut" onclick="makeRequest('p5');"
    value="p:=decimal(1/4)*x^2+decimal(2/3)*x+decimal(4/9)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p5','p6']);"
    value="q:=differentiate(p,x)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p5','p6','p7']);"
    value="g:=gcd(p,q)" />
  <div id="ansp7"><div></div></div>
 </li>
```

```
</ul>
More examples of expansions are available in
<a href="numrepeatingbinaryexpansions.xhtml">BinaryExpansion</a>,
<a href="numrepeatinghexexpansions.xhtml">HexadecimalExpansion</a>, and
<a href="db.xhtml?RadixExpansion">RadixExpansion</a>. Issue the system
command
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="showcall('p8');"
   value=")show RadixExpansion"/>
  <div id="ansp8"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?RadixExpansion">RadixExpansion</a>.
\getchunk{page foot}
```

———-

numrepeatinghexexpansions.xhtml

— numrepeatinghexexpansions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Repeating Hexadecimal Expansions</div>
  <hr/>
All rationals have repeating hexadecimals expansions. The operation
<a href="dbophex.xhtml">hex</a> returns these expansions of type
<a href="db.xhtml?HexadecimalExpansion">HexadecimalExpansion</a>.
Operations to access the individual numerals of a hexadecimal expansion
can be obtained by converting the value to
<a href="db.xhtml?RadixExpansion">RadixExpansion(16)</a>. More examples of
expansions are available in
<a href="numrepeatingdecimals.xhtml">DecimalExpansion</a>,
<a href="numrepeatingbinaryexpansions.xhtml">BinaryExpansion</a>, and
<a href="db.xhtml?RadixExpansion">RadixExpansion</a>.

This is a hexadecimal expansion of a rational number.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="r:=hex(22/7)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
```

```
Arithmetic is exact.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree([’p1’,’p2’]);"
    value="r+hex(6/7)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The period of the expansion can be short or long...
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest(’p3’);"
    value="[hex(1/i) for i in 350..354]" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
or very long.
.<ul>
 <li>
  <input type="submit" id="p4" class="subbut" onclick="makeRequest(’p4’);"
    value="hex(1/1007)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
These numbers are bona fide algebraic objects.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut" onclick="makeRequest(’p5’);"
    value="p:=hex(1/4)*x^2+hex(2/3)*x+hex(4/9)" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree([’p5’,’p6’]);"
    value="q:=D(p,x)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree([’p5’,’p6’,’p7’]);"
    value="g:=gcd(p,q)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Issue the system command
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="showcall(’p8’);"
   value=")show HexadecimalExpansion"/>
  <div id="ansp8"><div></div></div>
 </li>
</ul>
```

```
to display the full list of operations defined by
<a href="db.xhtml?HexadecimalExpansion">HexadecimalExpansion</a>.

\getchunk{page foot}
```

numromannumerals.xhtml

— numromannumerals.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Roman Numerals</div>
  <hr/>
The Roman numeral package was added to Axiom in MCMLXXXVI for use in
denoting higher order derivatives.

For example, let f be a symbolic operator.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut" onclick="makeRequest('p1');"
    value="f:=operator 'f" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
This is the seventh derivative of f with respect to x
<ul>
 <li>
  <input type="submit" id="p2" class="subbut" onclick="makeRequest('p2');"
    value="D(f x,x,7)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
You can have integers printed as Roman numerals by declaring variables
to be of type
<a href="db.xhtml?RomanNumeral">RomanNumeral</a>
(abbreviation <a href="db.xhtml?RomanNumeral">ROMAN</a>).
<ul>
 <li>
  <input type="submit" id="p3" class="subbut" onclick="makeRequest('p3');"
    value="a:=roman(1978-1965)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
This package now has a small but devoted group of followers that claim
this domain has shown its efficacy in many other contexts. They claim
```

```
that Roman numerals are every bit as useful as ordinary integers.
In a sense, they are correct, because Roman numerals form a ring and
you can therefore construct polynomials with Roman numeral
coefficients, matrices over Roman numerals,etc..
<ul>
 <li>
  <input type="submit" id="p4" class="subbut" onclick="makeRequest('p4');"
    value="x:UTS(ROMAN,'x,0):=x" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Was Fibonacci Italian or ROMAN?
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="recip(1-x-x^2)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
You can also construct fractions with Roman numeral numerators and
denominators, as this matrix Hilberticus illustrates.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut" onclick="makeRequest('p6');"
    value="m:MATRIX FRAC ROMAN" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value="m:=matrix [ [1/(i+j) for i in 1..3] for j in 1..3]" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Note that the inverse of the matrix has integral
<a href="db.xhtml?RomanNumeral">ROMAN</a> entries.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p6','p7','p8']);"
    value="inverse m" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Unfortunately, the spoil-sports say that the fun stops when the
numbers get big -- mostly because the Romans didn't establish
conventions about representing very large numbers.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut" onclick="makeRequest('p9');"
    value="y:=factorial 10" />
  <div id="ansp9"><div></div></div>
 </li>
```

```
</ul>
You work it out!
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p9','p10']);"
    value="roman y" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
Issue the system command
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="showcall('p11');"
   value=")show RomanNumeral"/>
  <div id="ansp11"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?RomanNumeral">RomanNumeral</a>).
\getchunk{page foot}
```

———-

ocwmit18085.xhtml

— **ocwmit18085.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
18.085 Mathematical Methods for Engineers I Course Notes
<hr/>
These are course notes based on the
<a href="http://ocw.mit.edu/OcwWeb/Mathematics/18-085Fall-2005/VideoLectures/index.htm">
 M.I.T. Open Courseware lectures by Gilbert Strang.
</a>
<ul>
 <li>
  <a href="ocwmit18085lecture1.xhtml">
   Positive Definite Matrices K=A'CA
  </a>
 </li>
 <li>
  <a href="ocwmit18085lecture2.xhtml">
   One-dimensional Applications: A = Difference Matrix
  </a>
 </li>
</ul>
\getchunk{page foot}
```

———-

ocwmit18085lecture1.xhtml

— ocwmit18085lecture1.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
 Positive Definite Matrices K=A'CA
<hr/>
In applied mathematics we have 2 basic tasks:
<ul>
<li>Find the equations</li>
<li>Solve the equations</li>
</ul>
<h4>Positive Definite Matrices</h4>
Certain matrices occur frequently in applied math. These three
matrices (K,T,and M) are canonical examples.
We have 3 3x3 matrices,
<pre>
K:Matrix(Integer):=[[2,-1,0],[-1,2,-1],[0,-1,2]]

        + 2   - 1   0 +
        |             |
        |- 1   2   - 1|
        |             |
        + 0   - 1   2 +
               Type: Matrix Integer
T:Matrix(Integer):=[[1,-1,0],[-1,2,-1],[0,-1,2]]

        + 1   - 1   0 +
        |             |
        |- 1   2   - 1|
        |             |
        + 0   - 1   2 +
               Type: Matrix Integer
B:Matrix(Integer):=[[1,-1,0],[-1,2,-1],[0,-1,1]]

        + 1   - 1   0 +
        |             |
        |- 1   2   - 1|
        |             |
        + 0   - 1   1 +
               Type: Matrix Integer
</pre>
These matrices are similar and can be generalized to square matrices
of order N, with n x n elements. All of these matrices have the same
element along the diagonal. T (aka Top) differs from K in the first row.
B (aka Both) differs from K in the first and last row. These represent
different boundary conditions in the problem.

We can create K(n), T(n) and B(n) with the following commands:
<pre>
```

```
k(n) ==
 M := diagonalMatrix([2 for i in 1..n])
 for i in 1..n-1 repeat M(i,i+1):=-1
 for i in 1..n-1 repeat M(i+1,i):=-1
 M::SquareMatrix(n,Fraction(Integer))
</pre>
<pre>
t(n) ==
 M:=k(n)
 N:=M::Matrix(Fraction(Integer))
 qsetelt!(N,1,1,1)
 N::SquareMatrix(n,Fraction(Integer))
</pre>
<pre>
b(n) ==
 M:=k(n)
 N:=M::Matrix(Fraction(Integer))
 qsetelt!(N,1,1,1)
 qsetelt!(N,n,n,1)
 N::SquareMatrix(n,Fraction(Integer))
</pre>

K:=k(n) has a few key properties:
<ul>
<li> K is symmetric, that is K=K^T</li>
<li> K might be nonsingular, that is, it is invertible</li>
<li> K has a non-zero determinant</li>
<li> K is banded (main diagonal and neighbors)</li>
<li> K is tri-diagonal (main diagonal and nearest neighbors</li>
<li> K is extremely sparse</li>
<li> K has constant diagonals, (shift invariant, time invariant)</li>
<li> K is Toeplitz (constant diagonal, shows up in filters)</li>
<li> K is good for Fourier analysis</li>
</ul>

<h5>The inverse of T</h5>
If we look at the inverse of the T matrix we see:
<pre>
T^-1

        +3  2  1+
        |       |
        |2  2  1|
        |       |
        +1  1  1+
               Type: Matrix Fraction Integer
</pre>
Notice that these are all integers because the determinant of
this matrix is 1
<pre>
determinant T

     1
               Type: Fraction Integer
```

</pre>
We can check that this matrix is the inverse of T.

When computing the inverse the row pattern [-1 2 -1] is a ‘‘second difference’’. The first column of the inverse matrix is [3 2 1] which is linear. When we take the second difference of a linear object we should get 0. Thus,
<pre>
```
[[-1,2,-1]]::MATRIX(INT)*[[3],[2],[1]]

     [0]
              Type: Matrix Integer
```

</pre>
The third column of the T matrix is linear and constant. If we take the second difference of that we also find it is zero:
<pre>
```
 [[-1,2,-1]]::MATRIX(INT)*[[1],[1],[1]]

    [0]
              Type: Matrix Integer
```
</pre>
and the diagonal element of the unit matrix must be one. So the second difference of the second column is:
<pre>
```
 [[-1,2,-1]]::MATRIX(INT)*[[2],[2],[1]]

    [1]
              Type: Matrix Integer
```
</pre>
So these simple checks show that we’re getting the correct row and column values for the identity matrix by multiplying T times its inverse.

<h5>The inverse of B</h5>
If we look for the inverse of the B matrix we can observe that the rows sum to zero which implies that it is not invertible. Thus it is singular.

K and T are positive definite. B is only positive semi-definite.

If we can find a vector that it takes to zero, that is if we can solve for x,y,z in:
<pre>
```
        + 1   - 1   0 + + x +    + 0 +
        |             | |   |    |   |
        |- 1   2   - 1| | y | =  | 0 |
        |             | |   |    |   |
        + 0   - 1   1 + + z +    + 0 +
```

</pre>
The constant vector [1 1 1] solves this equation. When

```
the rows sum to zero we are adding each row by a constant
and thus we add each row times the constant one and we
get zeros. If the matrix takes some vector to zero it
cannot have an inverse since if
<pre>
   B x = 0
</pre>
and x is not zero. If B had an inverse only x=0 would
solve the equation. Since x=1 solves the equation B has
no inverse. The vector x is in the nullspace of B. In
fact any constant vector, e.g. [3 3 3] is in the nullspace.
Thus the nullspace of B is cx for any constant c.

When doing matrix multiplication one way to think about the
work is to consider the problem by columns. Thus in the
multiplication
<pre>
        + 1   - 1   0 + + x +    + 0 +
        |             | |   |    |   |
        |- 1   2   - 1| | y | =  | 0 |
        |             | |   |    |   |
        + 0   - 1   1 + + z +    + 0 +

</pre>
we can think about this as
<pre>
x*(first column) + y*(second column) + z*(third column).
</pre>
and for the constant vector [1 1 1] this means that we
just need to sum the columns.

Alternatively this can be computed by thinking of the
multiplication as
<pre>
 (first row)*(vector)
 (second row)*(vector)
 (third row)*(vector)
</pre>

<br/>
<h5>The inverse of K</h5>
Now we consider the K matrix we see the inverse
<pre>
K

         + 2   - 1   0 +
         |             |
         |- 1   2   - 1|
         |             |
         + 0   - 1   2 +
               Type: SquareMatrix(3,Fraction Integer)
kinv:=K^-1

         +3  1  1+
```

```
         |-  -  -|
         |4  2  4|
         |       |
         |1     1|
         |-  1  -|
         |2     2|
         |       |
         |1  1  3|
         |-  -  -|
         +4  2  4+
               Type: SquareMatrix(3,Fraction Integer)
</pre>
We can take the determinant of k
<pre>
determinant K

    4
               Type: Fraction Integer
</pre>
Thus there is a constant 1/4 which can be factored out
<pre>
4*kinv

         +3  2  1+
         |       |
         |2  4  2|
         |       |
         +1  2  3+
               Type: SquareMatrix(3,Fraction Integer)
</pre>
Notice that the inverse is a symmetric matrix but not tri-diagonal.
The inverse is not a sparse matrix so much more computation would
be involved when using the inverse.

In order to solve the system
<pre>
 K u = f
</pre>
by elimination which implies multiplying and subtracting rows.
<pre>
       K    u  =  f    ==>   U     u  =    f
</pre>
For the 2x2 case we see:
<pre>
                             +2  -1+        +  f1  +
    +2  -1+  +x+   +f1+      |     |  +x+   |      |
    |     |  | | = |  |  ==> |    3|  | | = |   1  |
    +-1  2+  +y+   +f2+      |0   -|  +y+   |f2+-f1|
                             +    2+        +   2  +


</pre>
By multiplying row1 by 1/2 and adding it to row2 we create an
upper triangular matrix U. Since we chose K(1,1), the number 2
```

```
is called the first pivot. K(2,2), the number 3/2, is called
the second pivot.

For K 2x2 above is symmetric and invertible (since the pivots
are all non-zero).

For the K 3x3 case the pivots are 2, 3/2, and 4/3. (The next pivots
would be 5/4, 6/5, etc. for larger matrices).

For the T 3x3 case the pivots are 1, 1, and 1.

For the B 3x3 case the third pivot would be zero.

<hr/>
<h5>Generalizing the matrix pivot operations</h5>
For the 2x2 case we see contruct an elimination matrix E which we can use
to pre-multipy by K to give us the upper triangular matrix U
<pre>
      E     K    =   U
</pre>
In detail we see
<pre>

    +1  0+            +2  -1+
    |    |  +2  -1+   |     |
    |1   |  |     | = |    3|
    |-  1|  +-1  2+   |0   -|
    +2   +            +    2+

</pre>
We wish to rewrite this as
<pre>
       K = L U
</pre>

<hr/>
<h5>The big 4 solve operations in Linear Algebra</h5>
<ol>
<li>Elimination</li>
<li>Gram-Schmidt Orthoginalization</li>
<li>Eigenvalues</li>
<li>Singular Value Decomposition</li>
</ol>
Each of these operations is described by a factorization of K.
Elimination is written
<pre>
  K = L U
</pre>
where L is lower triangular and U is upper triangular.
Thus we need a matrix L which when multiplied by U gives K.
The required matrix is the inverse of the E matrix above since
<pre>

1)      E K =     U
```

```
    -1         -1
2)  E   E K = E   U

               -1
3)      I K = E   U

               -1
4)  but   L = E

5)  so    K = L U
```
</pre>
Given the matrix operations above we had
<pre>
```
      E       K   =   U

    +1  0+            +2  -1+
    |    |  +2  -1+   |     |
    |1   |  |     | = |    3|
    |-  1|  +-1  2+   |0   -|
    +2   +            +    2+

```
</pre>
and the inverse of E is the same matrix with a minus sign in
the second row, thus:
<pre>
```
        +  1  0+
   -1   |      |
  E   = |  1   | = L
        |- -  1|
        +  2   +

```
</pre>

<hr/>
<h5>Making the matrices symmetric</h5>
We would like to preserve the symmetry property which we can
do with a further decomposition of LU as follows:
<pre>
```
      L        U     =     L        D       U'

  +  1  0+  +2  -1+    +  1  0+  +2  0+  +1   1+
  |      |  |     |    |      |  |    |  |  - -|
  |  1   |  |    3|  = |  1   |  |   3|  |    2|
  |- -  1|  |0   -|    |- -  1|  |0  -|  |     |
  +  2   +  +    2+    +  2   +  +   2+  +0   1+

```
</pre>
So now we have 3 matrices; L is the lower triangular,
D is symmetric and contains the pivots, and U' is upper triangular and
is the transpose of the lower. So the real form we have is
<pre>
```
           T
    L  D  L
```

```
</pre>
This result will always be symmetric. We can check this by taking
its transpose. If we get the same matrix we must have a symmetric
matrix. So the transpose of
<pre>
            T  T     TT  T   T        T T        T
  (  L  D  L  )   = L   D   L   =  L D L  = L D L
</pre>
<hr/>
<h5>Positive Definite Matrices</h5>
There are several ways to recognize a positive definite matrix.
First, it must be symmetric. The "positive" aspect comes from
the pivots, all of which must be positive. Note that T is also
positive definite. B is positive semi-definite because one of
the pivots is zero. So
<pre>
   positive definite      == all pivots >  0
   positive semi-definite == all pivots >= 0
</pre>
When all the pivots are positive then all the eigenvalues are positive.

So a positive definite matrix K and any non-zero vector X
<pre>
    T
   X  K X  > 0
</pre>
X transpose is just a row and X is just a column.

\getchunk{page foot}
```

ocwmit18085lecture2.xhtml

— ocwmit18085lecture2.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
 One-dimensional Applications: A = Difference Matrix
<hr/>
<h5>Difference Matrices</h5>
<hr/>
<h5>Second Differences</h5>
<hr/>
<h5>Stiffness Matrix</h5>
<hr/>
<h5>Boundary Conditions</h5>
\getchunk{page foot}
```

operations.xhtml

— operations.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      operations not implemented
\getchunk{page foot}
```

outputfunctions.xhtml

— outputfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <div align="center">Output Functions</div>
  <hr/>
A number of operations exist for specifying how numbers of type
<a href="db.xhtml?Float">Float</a> are to be displayed. By default,
spaces are inserted every ten digits in the output for readability.
(Not that you cannot include spaces in the input form of a floating
point number, though you can use underscores.)

Output spacing can be modified with the
<a href="dbopoutputspacing.xhtml">outputSpacing</a> operation. This
inserts no spaces and then displays the value of x.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="outputSpacing 0; x:=sqrt 0.2" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
Issue this to have the spaces inserted every 5 digits.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="outputSpacing 5; x" />
  <div id="ansp2"><div></div></div>
 </li>
```

```
</ul>
By default, the system displays floats in either fixed format or
scientific format, depending on the magnitude of the number.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="y:=x/10^10" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
A particular format may be requested with the operations
<a href="dbopoutputfloating.xhtml">outputFloating</a> and
<a href="dbopoutputfixed.xhtml">outputFixed</a>.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="outputFloating(); x" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="outputFixed(); y" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Additionally, you can ask for n digits to be displayed after the
decimal point.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p3','p6']);"
    value="outputFloating 2; y" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p7']);"
    value="outputFixed 2; x" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
The <a href="dbopoutputgeneral.xhtml">outputGeneral</a> function resets
the output printing to the default behavior.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="outputGeneral()" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
```

```
\getchunk{page foot}
```

pagelist.xhtml

— pagelist.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pagelist not implemented
\getchunk{page foot}
```

pagematrix.xhtml

— pagematrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pagematrix not implemented
\getchunk{page foot}
```

pageonedimensionalarray.xhtml

— pageonedimensionalarray.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pageonedimensionalarray not implemented
\getchunk{page foot}
```

pageset.xhtml

— pageset.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
```

```
\getchunk{page head}
      pageset not implemented
\getchunk{page foot}
```

———-

pagetable.xhtml

— pagetable.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pagetable not implemented
\getchunk{page foot}
```

———-

pagepermanent.xhtml

— pagepermanent.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pagepermanent not implemented
\getchunk{page foot}
```

———-

pagesquarematrix.xhtml

— pagesquarematrix.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pagesquarematrix not implemented
\getchunk{page foot}
```

———-

pagetwodimensionalarray.xhtml

— pagetwodimensionalarray.xhtml —

```
\getchunk{standard head}
```

```
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">TwoDimensionalArray</div>
  <hr/>
The <a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a> is used for
storing data in a two-dimensional data structure indexed by row and column.
Such an array is a homogeneous data structure in that all the entries of the
array must belong to the same Axiom domain (although see
<a href="axbook/section-2.6.xhtml">The Any Domain</a>). Each array has a fixed
number of rows and columns specified by the user and arrays are not
extensible. In Axiom, the indexing of two-dimensional arrays is one-based.
This means that both the "first" row of an array and the "first" column of
an array are given the index 1. Thus, the entry in the upper left corner
of an array is in position (1,1).

The operation <a href="dbopnew.xhtml">new</a> creates an array with a
specified number of rows and columns and fills the components of that array
with a specified entry. The arguments of this operation specify the number
of rows, the number of columns, and the entry. This creates a five-by-four
array of integers, all of whose entries are zero.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="arr:ARRAY2 INT:=new(5,4,0)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The entries of this array can be set to other integers using the operation
<a href="dbopsetelt.xhtml">setelt</a>.

Issue this to set the element in the upper left corner of this array to 17.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="setelt(arr,1,1,17)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Now the first element of the array is 17.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="arr" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
```

```
Likewise, elements of an array are extracted using the operation
<a href="dbopelt.xhtml">elt</a>.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p4']);"
    value="elt(arr,1,1)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Another way to use these two operations is as follows. This sets the
element in position (3,2) of the array to 15.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p5']);"
    value="arr(3,2):=15" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
This extracts the element in position (3,2) of the array.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p5','p6']);"
    value="arr(3,2)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The operation <a href="dbopelt.xhtml">elt</a> and
<a href="dbopsetelt.xhtml">setelt</a> come equipped with an error check
which verifies that the indices are in the proper ranges. For example,
the above array has five rows and four columns, so if you ask for the
entry in position (6,2) with arr(6,2) Axiom displays an error message.
If there is no need for an error check, you can call the operations
<a href="dbopqelt.xhtml">qelt</a> and
<a href="dbopqseteltbang.xhtml">qsetelt!</a>
which provide the same functionality
but without the error check. Typically, these operations are called in
well-tested programs.

The operations <a href="dboprow.xhtml">row</a> and
<a href="dbopcolumn.xhtml">column</a> extract rows and columns, respectively,
and return objects of
<a href="db.xhtml?OneDimensionalArray">OneDimensionalArray</a> with the
same underlying element type.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p5','p7']);"
    value="row(arr,1)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p5','p8']);"
    value="column(arr,1)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
You can determine the dimensions of an array by calling the operations
<a href="dbopnrows.xhtml">nrows</a> and
<a href="dbopncols.xhtml">ncols</a>, which return the number of rows
and columns, respectively.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p2','p5','p9']);"
    value="nrows(arr)" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p2','p5','p10']);"
    value="ncols(arr)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
To apply an operation to every element of an array, use
<a href="dbopmap.xhtml">map</a>. This creates a new array. This
expression negates every element.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11']);"
    value="map(-,arr)" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
This creates an array where all the elements are doubled.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12']);"
    value="map((x+->x+x),arr)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
To change the array destructively, use
<a href="dbopmapbang.xhtml">map!</a> instead of
<a href="dbopmap.xhtml">map</a>.
If you need to make a copy of an array,
use <a href="dbopcopy.xhtml">copy</a>.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p13']);"
    value="arrc:=copy(arr)" />
```

```
  <div id="ansp13"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p13','p14']);"
    value="map!(-,arrc)" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p13','p14','p15']);"
    value="arrc" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p16']);"
    value="arr" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Use <a href="dbopmemberq.xhtml">member?</a> to see if a given element is in
an array.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p17']);"
    value="member?(17,arr)" />
  <div id="ansp17"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p18']);"
    value="member?(10317,arr)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
To see how many times an element appears in an array, use
<a href="dbopcount.xhtml">count</a>.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p19']);"
    value="count(17,arr)" />
  <div id="ansp19"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree(['p1','p2','p5','p11','p12','p20']);"
    value="count(0,arr)" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
For more information about the operations available for
```

```
<a href="db.xhtml?TwoDimensionalArray">TwoDimensionalArray</a>, issue
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="showcall('p21');"
   value=")show TwoDimensionalArray"/>
  <div id="ansp21"><div></div></div>
 </li>
</ul>
For more information on related topics, see
<a href="pagematrix.xhtml">Matrix</a> and
<a href="pageonedimensionalarray.xhtml">OneDimensionalArray</a>.
\getchunk{page foot}
```

pagevector.xhtml

— pagevector.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      pagevector not implemented
\getchunk{page foot}
```

polybasicfunctions.xhtml

— polybasicfunctions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Basic Operations on Polynomials</div>
  <hr/>
You create polynomials using the usual operations of
<a href="dbopplus.xhtml">+</a>,
<a href="dbopminus.xhtml">-</a>,
<a href="dboptimes.xhtml">*</a>
(for multiplication), and
<a href="dbopstarstar.xhtml">**</a> (or
<a href="dbopstarstar.xhtml">^</a>. Here are two examples:
<ul>
 <li>
```

```
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="p:=a*x**2+b*x*y+c*y**2" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="q:=12*x^2+3*z" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
These operations can also be used to combine polynomials. Try the following:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p2','p3']);"
    value="p+q" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p4']);"
    value="p-3*q" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p5']);"
    value="p**2+p*q" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p6']);"
    value="r:=(p+q)**2" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
As you can see from the above examples, the variables are ordered by defaults
<pre>
  z > y > x > c > b > a
</pre>
That is, z is the main variable, then y and so on in reverse alphabetical
order. You can redefine this ordering (for display purposes) with the
<a href="dbopsetvariableorder.xhtml">setVariableOrder</a>. For example, the
following makes a the main variable, then b, and so on:
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="setVariableOrder [a,b,c,x,y,z]" />
  <div id="ansp7"><div></div></div>
 </li>
```

```
</ul>
Now compare the way polynomials are displayed:
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p7','p8']);"
    value="p" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p2','p7','p9']);"
    value="q" />
  <div id="ansp9"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p2','p6','p7','p10']);"
    value="r" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
To return to the system's default ordering, use
<a href="dbopresetvariableorder.xhtml">resetVariableOrder</a>.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="makeRequest('p11');"
    value="resetVariableOrder()" />
  <div id="ansp11"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p11','p12']);"
    value="p" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
Polynomial coefficients can be pulled out using the function
<a href="dbopcoefficient.xhtml">coefficient</a>. For example:
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p2','p13']);"
    value="coefficient(q,x,2)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
will give you the coefficient of x**2 in the polynomial q. Try these
commands:
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p6','p14']);"
```

```
    value="coefficient(r,x,3)" />
  <div id="ansp14"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p6','p15']);"
    value="c:=coefficient(r,z,1)" />
  <div id="ansp15"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p6','p15','p16']);"
    value="coefficient(c,x,2)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Coefficients of monomials can be obtained as follows:
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p2','p17']);"
    value="coefficient(q**2,[x,z],[2,1])" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
This will return the coefficient of x**2*z in the polynomial q**2. Also,
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="handleFree(['p1','p2','p6','p18']);"
    value="coefficient(r,[x,y],[2,2])" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
will return the coefficient of x**2*y**2 in the polynomial r(x,y).
\getchunk{page foot}
```

———-

polyfactorization.xhtml

— polyfactorization.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Polynomial Factorization</div>
  <hr/>
The Axiom polynomial factorization facilities are available for all
polynomial types and a wide variety of coefficient domains. Here are
some examples.
<ul>
```

```
 <li>
  <a href="polyfactorization1.xhtml">
   Integer and Rational Number Coefficients
  </a>
 </li>
 <li>
  <a href="polyfactorization2.xhtml">
   Finite Field Coefficients
  </a>
 </li>
 <li>
  <a href="polyfactorization3.xhtml">
   Simple Algebraic Extension Field Coefficients
  </a>
 </li>
 <li>
  <a href="polyfactorization4.xhtml">
   Factoring Rational Functions
  </a>
 </li>
</ul>
\getchunk{page foot}
```

polyfactorization1.xhtml

— polyfactorization1.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Integer and Rational Number Coefficients</div>
  <hr/>
Polynomials with integer coefficients can be factored.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="v:=(4*x^3+2*y^2+1)*(12*x^5-(1/2)*x^3+12)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="factor v" />
  <div id="ansp2"><div></div></div>
 </li>
```

```
</ul>
Also, Axiom can factor polynomials with rational number coefficients
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="w:=(4*x^3+(2/3)*x^2+1)*(12*x^5-(1/2)*x^3+12)" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value="factor w" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

polyfactorization2.xhtml

— polyfactorization2.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Finite Field Coefficients</div>
  <hr/>
Polynomials with coefficients in a finite filed can also be factored.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="u:POLY(PF(19)):=3*x^4+2*x^2+15*x+18" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
These include the integers mod p, where p is prime, and extensions of these
fields.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="factor u" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
```

```
Convert this to have coefficients in the finite field with
19**3 elements. See
<a href="axbook/section-8.11.xhtml">FiniteFields</a> for more information
about finite fields.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="factor(u::POLY FFX(PF 19,3))" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

polyfactorization3.xhtml

— polyfactorization3.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Simple Algebraic Extension Field Coefficients</div>
  <hr/>
Polynomials with coefficients in simple algebraic extensions of the
rational numbers can be factored.

Here, aa and bb are symbolic roots of polynomials.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="aa:=rootOf(aa^2+aa+1)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="p:=(x^2+aa^2*x+y)*(aa*x^2+aa*x+aa*y^2)^2" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Note that the second argument to factor can be a list of algebraic
extensions to factor over.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
```

```
    onclick="handleFree(['p1','p2','p3']);"
    value="factor(p,[aa])" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
This factors x^2+3 over the integers.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="factor(x^2+3)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Factor the same polynomial over the field obtained by adjoining aa to the
rational numbers.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="factor(x^2+3,[aa])" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Factor x^6+108 over the same field.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p6']);"
    value="factor(x^6+108,[aa])" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="bb:=rootOf(bb^3-2)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p7','p8']);"
    value="factor(x^6+8,[bb])" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Factor again over the field obtained by adjoining both aa and bb to the
rational numbers.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p7','p9']);"
    value="factor(x^6+108,[aa,bb])" />
  <div id="ansp9"><div></div></div>
 </li>
```

```
</ul>
\getchunk{page foot}
```

polyfactorization4.xhtml

— polyfactorization4.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Factoring Rational Functions</div>
  <hr/>
Since fractions of polynomials form a field, every element (other than zero)
divides any other, so there is no useful notion of irreducible factors.
Thus the <a href="dbopfactor.xhtml">factor</a> operation is not very useful
for fractions of polynomials.

Instead, there is a specific operation
<a href="dbopfactorfraction.xhtml">factorFraction</a> that separately
factors the numerator and denominator and returns a fraction of the
factored results.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="factorFraction((x^2-4)/(y^2-4))" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
You can also use <a href="dbopmap.xhtml">map</a>. This expression applies
the <a href="dbopfactor.xhtml">factor</a> operation to the numerator and
denominator.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="map(factor,(x^2-4)/(y^2-4))" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

polygcdandfriends.xhtml

— polygcdandfriends.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
<div align="center">
 Greatest Common Divisors, Resultants, and Discriminants
</div>
<hr/>
You can compute the greatest common divisor of two polynomials using the
function <a href="dbopgcd.xhtml">gcd</a>. Here’s an example:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest(’p1’);"
    value="p:=3*x^8+2*x^7+6*x^2+7*x+2" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest(’p2’);"
    value="q:=2*x^13+9*x^7+2*x^6+10*x+5" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p3’]);"
    value="gcd(p,q)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
You could also see that p and q have a factor in common by using the
function <a href="dbopresultant.xhtml">resultant</a>:
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree([’p1’,’p2’,’p4’]);"
    value="resultant(p,q,x)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
The resultant of two polynomials vanishes precisely when they have a
factor in common. (In the example above we specified the variable with which
we wanted to compute the resultant because the polynomials could have
involved variables other than x.)
\getchunk{page foot}
```

———-

polynomialpage.xhtml

— polynomialpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Polynomials</div>
  <hr/>
<table>
 <tr>
  <td>
   <a href="polybasicfunctions.xhtml">Basic Functions</a>
  </td>
  <td>
   Create and manipulate polynomials
  </td>
 </tr>
 <tr>
  <td>
   <a href="polysubstitutions.xhtml">Substitutions</a>
  </td>
  <td>
   Evaluate Polynomials
  </td>
 </tr>
 <tr>
  <td>
   <a href="polyfactorization.xhtml">Factorization</a>
  </td>
  <td>
   Factor in different contexts
  </td>
 </tr>
 <tr>
  <td>
   <a href="polygcdandfriends.xhtml">GCD and Friends</a>
  </td>
  <td>
   Greatest Common Divisors, Resultants, and Discriminants
  </td>
 </tr>
 <tr>
  <td>
   <a href="polyroots.xhtml">Roots</a>
  </td>
  <td>
   Work with and solve for roots
  </td>
 </tr>
```

```
 <tr>
  <td>
   <a href="polyspecifictypes.xhtml">Specific Types</a>
  </td>
  <td>
   More specific information
  </td>
 </tr>
</table>
\getchunk{page foot}
```

polyroots.xhtml

— polyroots.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">Roots of Polynomials</div>
  <hr/>
<table>
 <tr>
  <td>
   <a href="polyroots1.xhtml">
    Using a Single Root of a Polynomial
   </a>
  </td>
  <td>
   Working with a single root of a polynomial
  </td>
 </tr>
 <tr>
  <td>
   <a href="polyroots2.xhtml">
    Using All Roots of a Polynomial
   </a>
  </td>
  <td>
   Working with all the roots of a polynomial
  </td>
 </tr>
 <tr>
  <td>
   <a href="polyroots3.xhtml">
    Solution of a Single Polynomial Equation
   </a>
  </td>
  <td>
   Finding the roots of one polynomial
  </td>
```

```
 </tr>
 <tr>
  <td>
   <a href="polyroots4.xhtml">
    Solution of Systems of Polynomial Equations
   </a>
  </td>
  <td>
   Finding the roots of a system of polynomials
  </td>
 </tr>
</table>
\getchunk{page foot}
```

polyroots1.xhtml

— polyroots1.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Using a Single Root of a Polynomial</div>
  <hr/>
Use <a href="dboprootof.xhtml">rootOf</a> to get a symbolic root of a
polynomial. The call rootOf(p,x) returns a root of p(x).

This creates an algebraic number a, which is a root of the polynomial
returned in symbolic form.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="aa:=rootOf(a^4+1,a)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
To find the algebraic relation that defines a, use
<a href="dbopdefiningpolynomial.xhtml">definingPolynomial</a>
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="definingPolynomial aa" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
```

```
You can use a in any further expression, including a nested
<a href="dboprootof.xhtml">rootOf</a>.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="bb:=rootOf(b^2-aa-1,b)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Higher powers of the roots are automatically reduced during calculations.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p3','p4']);"
    value="g:=aa+bb" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p3','p4','p5']);"
    value="g^5" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
The operation <a href="dbopzeroof.xhtml">zeroOf</a> is similar to
<a href="dboprootof.xhtml">rootOf</a>, except that it may express the
root using radicals in some cases.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="rootOf(c^2+c+1,c)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="zeroOf(d^2+d+1,d)" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="makeRequest('p8');"
    value="rootOf(e^5-2,e)" />
  <div id="ansp8"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="zeroOf(f^5-2,f)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
```

```
\getchunk{page foot}
```

———-

polyroots2.xhtml

— polyroots2.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Using All Roots of a Polynomial</div>
  <hr/>
Use <a href="dboprootsof.xhtml">rootsOf</a> to get all symbolic roots
of a polynomial. The call rootsOf(p,x) returns a list of all the roots
of p(x). If p(x) has a multiple root of order n, then that root appears
n times in the list.

Compute all the roots of x^4+1.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="l:=rootsOf(x^4+1,x)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
As a side effect, the variables %x0, %x1, and %x2 are bound to the first
three roots of x^4+1.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="%x0^5" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Although they all satisfy x^4+1=0, %x0, %x1, and %x2 are different
algebraic numbers. To find the algebraic relation that defines each of
them, use <a href="dbopdefiningpolynomial.xhtml">definingPolynomial</a>.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="definingPolynomial %x0" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
```

```
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="definingPolynomial %x1" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p5']);"
    value="definingPolynomial %x2" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
We can check that the sum and product of the roots of x^4+1 are its
trace and norm.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p6']);"
    value="x3:=last l" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p6','p7']);"
    value="%x0+%x1+%x2+x3" />
  <div id="ansp7"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p6','p8']);"
    value="%x0*%x1*%x2*x3" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
Corresponding to the pair of operations
<a href="dboprootof.xhtml">rootOf</a> and
<a href="dbopzeroof.xhtml">zeroOf</a> in
<a href="axbook/section-8.5.xhtml#subsec-8.5.2">
Solution of a Single Polynomial Equation</a>
there is an operations <a href="dbopzerosof.xhtml">zerosOf</a> that, like
<a href="dboprootsof.xhtml">rootsOf</a>, computes all the roots of a given
polynomial, but which expresses some of them in terms of radicals.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="makeRequest('p9');"
    value="zerosOf(y^4+1,y)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
As you see, only one implicit algebraic number was created (%y1), and its
defining equation is this. The other three roots are expressed in radicals.
<ul>
 <li>
```

```
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p9','p10']);"
    value="definingPolynomial %y1" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

polyroots3.xhtml

— polyroots3.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Solution of a Single Polynomial Equation</div>
  <hr/>
Axiom can solve polynomial equations producing either approximate or exact
solutions. Exact solutions are either members of the ground field or can
be presented symbolically as roots of irreducible polynomials.

This returns one rational root along with an irreducible polynomial
describing the other solutions
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="solve(x^3=8,x)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
If you want solutions expressed in terms of radicals you would use this
instead.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="radicalSolve(x^3=8,x)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
The <a href="dbopsolve.xhtml">solve</a> command always returns a value but
<a href="dbopradicalsolve.xhtml">radicalSolve</a> returns only the solutions
that it is able to express in terms of radicals.

If the polynomial equation has rational coefficients you can ask for
```

```
approximations to its real roots by calling solve with a second argument
that specifies the "precision" epsilon. This means that each approximation
will be within plus or minus epsilon of the actual result.

Notice that the type of second argument controls the type of the result.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="solve(x^4-10*x^3+35*x^2-50*x+25,.0001)" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
If you give a floating point precision you get a floating point result.
If you give the precision as a ration number you get a rational result.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="solve(x^2-2,1/1000)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
If you want approximate complex results you should use the command
<a href="dbopcomplexsolve.xhtml">complexSolve</a> that takes the same
precision argument epsilon.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="complexSolve(x^3-2,.0001)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Each approximation will be within plus or minus epsilon of the actual result
in each of the real and imaginary parts.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="complexSolve(x^2-2*%i+1,1/100)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
Note that if you omit the = from the first argument Axiom generates
an equation by equating the first argument to zero. Also, when only one
variable is present in the equation, you do not need to specify the
variable to be solved for, that is, you can omit the second argument.

Axiom can also solve equations involving rational functions. Solutions
where the denominator vanishes are discarded.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
```

```
    onclick="makeRequest('p7');"
    value="radicalSolve(1/x^3+1/x^2+1/x=0,x)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

polyroots4.xhtml

— polyroots4.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Solution of Systems of Polynomial Equations</div>
  <hr/>
Given a system of equations of rational functions with exact coefficients
<pre>
     p1(x1,...,xn)
         .
         .
     pm(x1,...,xn)
</pre>
Axiom can find numeric or symbolic solutions. The system is first split
into irreducible components, then for each component, a triangular system
of equations is found that reduces the problem to sequential solutions of
univariate polynomials resulting from substitution of partial solutions
from the previous stage.
<pre>
     q1(x1,...,xn)
         .
         .
     qm(xn)
</pre>
Symbolic solutions can be presented using "implicit" algebraic numbers
defined as roots of irreducible polynomials or in terms of radicals. Axiom
can also find approximations to the real or complex roots of a system of
polynomial equations to any user specified accuracy.

The operation <a href="dbopsolve.xhtml">solve</a> for systems is used in
a way similar to <a href="dbopsolve.xhtml">solve</a> for single equations.
Instead of a polynomial equation, one has to give a list of equations and
instead of a single variable to solve for, a list of variables. For
solutions of single equations see
<a href="axbook/section-8.5.xhtml#subsec-8.5.2">
Solution of a Single Polynomial Equation</a>
```

```
Use the operation <a href="dbopsolve.xhtml">solve</a> if you want
implicitly presented solutions.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="solve([3*x^2+y+1,y^2-4],[x,y])" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="solve([x=y^2-19,y=z^2+x+3,z=3*x],[x,y,z])" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
Use <a href="dbopradialsolve.xhtml">radicalSolve</a> if you want your
solutions expressed in terms of radicals.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="radicalSolve([3*x^3+y+1,y^2-4],[x,y])" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
To get numeric solutions you only need to give the list of equations and
the precision desired. The list of variables would be redundant information
since there can be no parameters for the numerical solver.

If the precision is expressed as a floating point number you get results
expressed as floats.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="solve([x^2*y-1,x*y^2-2],.01)" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
To get complex numeric solutions, use the operation
<a href="dbopcomplexsolve.xhtml">complexSolve</a>, which takes the same
arguments as in the real case.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="makeRequest('p5');"
    value="complexSolve([x^2*y-1,x*y^2-2],1/1000)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
It is also possible to solve systems of equations in rational functions
over the rational numbers. Note that [x=0.0,a=0.0] is not returned as
```

```
a solution since the denominator vanishes there.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="solve([x^2/a=a,a=a*x],.001)" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
When solving equations with denominators, all solutions where the
denominator vanishes are discarded.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="radicalSolve([x^2/a+a+y^3-1,a*y+a+1],[x,y])" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

polyspecifictypes.xhtml

— polyspecifictypes.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <div align="center">The Specific Polynomial Types</div>
  <hr/>
<table>
 <tr>
  <td>
   <a href="polyspecifictypes1.xhtml">
    Polynomial
   </a>
  </td>
  <td>
   The general type
  </td>
 </tr>
 <tr>
  <td>
   <a href="polyspecifictypes2.xhtml">
    UnivariatePolynomial
   </a>
  </td>
  <td>
   One variable polynomials
  </td>
```

```
 </tr>
 <tr>
  <td>
   <a href="polyspecifictypes3.xhtml">
    MultivariatePolynomial
   </a>
  </td>
  <td>
   Multiple variable polynomials, recursive structure
  </td>
 </tr>
 <tr>
  <td>
   <a href="polyspecifictypes4.xhtml">
    DistributedMultivariatePolynomial
   </a>
  </td>
   Multiple variable polynomials, non-recursive structure
  <td>
  </td>
 </tr>
</table>
\getchunk{page foot}
```

———-

polyspecifictypes1.xhtml

— polyspecifictypes1.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Polynomial</div>
  <hr/>
The domain constructor <a href="db.xhtml?Polynomial">Polynomial</a>
(abbreviation: <a href="db.xhtml?Polynomial">POLY</a>) provides polynomials
with an arbitrary number of unspecified variables.

It is used to create the default polynomial domains in Axiom. Here the
coefficients are integers.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="x+1" />
  <div id="ansp1"><div></div></div>
 </li>
```

```
</ul>
Here the coefficients have type <a href="db.xhtml?Float">Float</a>.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="z-2.3" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
And here we have a polynomial in two variables with coefficients which
have type <a href="dbfractioninteger.xhtml">Fraction Integer</a>
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="y^2-z+3/4" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
The representation of objects of domains created by
<a href="db.xhtml?Polynomial">Polynomial</a> is that of recursive univariate
polynomials. (The term univariate means "one variable". The term
multivariate means "possibly more than one variable".) This recursive
structure is sometimes obvious from the display of a polynomial.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="r:=y^2+x*y+y" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
In this example, you see that the polynomial is stored as a polynomial in y
with coefficients that are polynomials in x with integer coefficients. In
fact, you really don't need to worry about the representation unless you are
working on an advanced application where it is critical. The polynomial
types created from
<a href="db.xhtml?DistributedMultivariatePolynomial">
DistributedMultivariatePolynomial</a> and
<a href="db.xhtml?XDistributedPolynomial">XDistributedPolynomial</a>
(discussed in
<a href="axbook/section-9.16.xhtml">"DistributedMultivariatePolynomial"</a>
are stored and displayed in a
non-recursive manner. You see a "flat" display of the above polynomial by
converting to one of those types.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="r::DMP([y,x],INT)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
```

```
We will demonstrate many of the polynomial facilities by using two
polynomials with integer coefficients. By default, the interpreter
expands polynomial expressions, even if they are written in a factored
format.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="p:=(y-1)^2*x*z" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
See <a href="axbook/section-9.22.xhtml">Factored</a>
to see how to create objects in factored form directly.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="makeRequest('p7');"
    value="q:=(y-1)*x*(z+5)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
The fully factored form can be recovered by using
<a href="dbopfactor.xhtml">factor</a>
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p7','p8']);"
    value="factor(q)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
This is the same name used for the operation to factor integer.
Such reuse of names is called
<a href="glossarypage.xhtml#p36465">overloading</a> and makes it much
easier to think of solving problems in general ways. Axiom facilities
for factoring polynomials created with
<a href="db.xhtml?Polynomial">Polynomial</a>
are currently restricted to the integer and rational number coefficients
cases. There are more complete facilities for factoring univariate
polynomials (see
<a href="axbook/section-8.2.xhtml">Polynomial Factorization</a>)

The standard arithmetic operations are available for polynomials.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p6','p7','p9']);"
    value="p-q^2" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
The operation <a href="dbopgcd.xhtml">gcd</a> is used to compute the
greated common divisor of two polynomials.
```

```
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree([’p6’,’p7’,’p10’]);"
    value="m:=gcd(p,q)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
In the case of p and q, the gcd is obvious from their definitions.
We factor the gcd to show this relationship better.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree([’p6’,’p7’,’p10’,’p11’]);"
    value="factor m" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
The least common multiple is computed by using
<a href="dboplcm.xhtml">lcm</a>.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree([’p6’,’p7’,’p12’]);"
    value="lcm(p,q)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
Use <a href="dbopcontent.xhtml">content</a> to compute the greatest common
divisor of the coefficients of the polynomial.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree([’p6’,’p13’]);"
    value="content p" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
Many of the operations on polynomials require you to specify a variable.
For example, <a href="dbopresultant.xhtml">resultant</a> requires you to
give the variable in which the polynomials should be expressed. This
computes the resultant of the values of p and q, considering them as
polynomials in the variable z. They do not share a root when thought
of as polynomials in z.
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree([’p6’,’p7’,’p14’]);"
    value="resultant(p,q,z)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
This value is 0 because as polynomials in x the polynomials have a
common root.
```

```
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree([’p6’,’p7’,’p15’]);"
    value="resultant(p,q,x)" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
The data type used for the variables created by
<a href="db.xhtml?Polynomial">Polynomial</a> is
<a href="db.xhtml?Symbol">Symbol</a>. As mentioned above, the representation
used by <a href="db.xhtml?Polynomial">Polynomial</a> is recursive and so
there is a main variable for nonconstant polynomials. The operation
<a href="dbopmainvariable.xhtml">makeVariable</a> returns this variable.
The return type is actually a union of <a href="db.xhtml?Symbol">Symbol</a>
and "failed".
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree([’p6’,’p16’]);"
    value="mainVariable p" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
The latter branch of the union is used if the polynomial has no
variables, that is, is a constant.
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree([’p6’,’p17’]);"
    value="ground? p" />
  <div id="ansp17"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest(’p18’);"
    value="ground?(1::POLY INT)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
The complete list of variables actually used in a particular polynomial
is returned by <a href="dbopvariables.xhtml">variables</a>. For constant
polynomials, this list is empty.
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree([’p6’,’p19’]);"
    value="variables p" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
The <a href="dbopdegree.xhtml">degree</a> operation returns the degree
of a polynomial in a specific variable.
<ul>
```

```
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="handleFree([’p6’,’p20’]);"
    value="degree(p,x)" />
  <div id="ansp20"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="handleFree([’p6’,’p21’]);"
    value="degree(p,y)" />
  <div id="ansp21"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="handleFree([’p6’,’p22’]);"
    value="degree(p,z)" />
  <div id="ansp22"><div></div></div>
 </li>
</ul>
If you give a list of variables for the second argument, a list of the
degrees in those variables is returned.
<ul>
 <li>
  <input type="submit" id="p23" class="subbut"
    onclick="handleFree([’p6’,’p23’]);"
    value="degree(p,[x,y,z])" />
  <div id="ansp23"><div></div></div>
 </li>
</ul>
The minimum degree of a variable in a polynomial is computed using
<a href="dbopminimumdegree.xhtml">minimumDegree</a>.
<ul>
 <li>
  <input type="submit" id="p24" class="subbut"
    onclick="handleFree([’p6’,’p24’]);"
    value="minimumDegree(p,z)" />
  <div id="ansp24"><div></div></div>
 </li>
</ul>
The total degree of a polynomial is returned by
<a href="dboptotaldegree.xhtml">totalDegree</a>.
<ul>
 <li>
  <input type="submit" id="p25" class="subbut"
    onclick="handleFree([’p6’,’p25’]);"
    value="totalDegree p" />
  <div id="ansp25"><div></div></div>
 </li>
</ul>
It is often convenient to think of a polynomial as a leading monomial
plus the remaining terms, using the operation
<a href="dbopleadingmonomial.xhtml">leadingMonomial</a>
<ul>
 <li>
```

```
  <input type="submit" id="p26" class="subbut"
    onclick="handleFree(['p6','p26']);"
    value="leadingMonomial p" />
  <div id="ansp26"><div></div></div>
 </li>
</ul>
The <a href="dbopreductum.xhtml">reductum</a> operation returns a polynomial
consisting of the sum of the monomials after the first.
<ul>
 <li>
  <input type="submit" id="p27" class="subbut"
    onclick="handleFree(['p6','p27']);"
    value="reductum p" />
  <div id="ansp27"><div></div></div>
 </li>
</ul>
These have the obvious relationship that the original polynomial is equal
to the leading monomial plus the reductum.
<ul>
 <li>
  <input type="submit" id="p28" class="subbut"
    onclick="handleFree(['p6','p28']);"
    value="p-leadingMonomial p - reductum p" />
  <div id="ansp28"><div></div></div>
 </li>
</ul>
The value returned by <a href="dbopleadingmonomial.xhtml">leadingMonomial</a>
includes the coefficient of that term. This is extracted by using
<a href="dbopleadingcoefficient.xhtml">leadingCoefficient</a> on the
original polynomial.
<ul>
 <li>
  <input type="submit" id="p29" class="subbut"
    onclick="handleFree(['p6','p29']);"
    value="leadingCoefficient p" />
  <div id="ansp29"><div></div></div>
 </li>
</ul>
The operation <a href="dbopeval.xhtml">eval</a> is used to substitute a
value for a varialbe in a polynomial.
<ul>
 <li>
  <input type="submit" id="p30" class="subbut"
    onclick="handleFree(['p6','p30']);"
    value="p" />
  <div id="ansp30"><div></div></div>
 </li>
</ul>
This value may be another variable, a constant or a polynomial.
<ul>
 <li>
  <input type="submit" id="p31" class="subbut"
    onclick="handleFree(['p6','p31']);"
    value="eval(p,x,w)" />
```

```
  <div id="ansp31"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p32" class="subbut"
    onclick="handleFree(['p6','p32']);"
    value="eval(p,x,1)" />
  <div id="ansp32"><div></div></div>
 </li>
</ul>
Actually, all the things being substituted are just polynomials, some
more trivial than others.
<ul>
 <li>
  <input type="submit" id="p33" class="subbut"
    onclick="handleFree(['p6','p33']);"
    value="eval(p,x,y^2-1)" />
  <div id="ansp33"><div></div></div>
 </li>
</ul>
Derivatives are computed using the <a href="dbopd.xhtml">D</a> operation.
<ul>
 <li>
  <input type="submit" id="p34" class="subbut"
    onclick="handleFree(['p6','p34']);"
    value="D(p,x)" />
  <div id="ansp34"><div></div></div>
 </li>
</ul>
The first argument is the polynomial and the second is the variable.
<ul>
 <li>
  <input type="submit" id="p35" class="subbut"
    onclick="handleFree(['p6','p35']);"
    value="D(p,y)" />
  <div id="ansp35"><div></div></div>
 </li>
</ul>
Even if the polynomial has only one variable, you must specify it.
<ul>
 <li>
  <input type="submit" id="p36" class="subbut"
    onclick="handleFree(['p6','p36']);"
    value="D(p,z)" />
  <div id="ansp36"><div></div></div>
 </li>
</ul>
Integration of polynomials is similar and the
<a href="dbopintegrate.xhtml">integrate</a> operation is used.

Integration requires that the coefficients support division.
Consequently, Axiom converts polynomials over the integers to polynomials
over the rational numbers before integrating them.
<ul>
 <li>
```

```
  <input type="submit" id="p37" class="subbut"
    onclick="handleFree(['p6','p37']);"
    value="integrate(p,y)" />
  <div id="ansp37"><div></div></div>
 </li>
</ul>
It is not possible, in general, to divide two polynomials. In our example
using polynomials over the integers, the operation
<a href="dbopmonicdivide.xhtml">monicDivide</a> divides a polynomial by a
monic polynomial (that is, a polynomial with leading coefficient equal to
1). The result is a record of the quotient and remainder of the division.
You must specify the variable in which to express the polynomial.
<ul>
 <li>
  <input type="submit" id="p38" class="subbut"
    onclick="handleFree(['p6','p38']);"
    value="qr:=monicDivide(p,x+1,x)" />
  <div id="ansp38"><div></div></div>
 </li>
</ul>
The selectors of the components of the record are quotient and
remainder. Issue this to extract the remainder:
<ul>
 <li>
  <input type="submit" id="p39" class="subbut"
    onclick="handleFree(['p6','p38','p39']);"
    value="qr.remainder" />
  <div id="ansp39"><div></div></div>
 </li>
</ul>
Now that we can extract the components, we can demonstrate the
relationship among them and the arguments to our original expression
<pre>
  qr:=monicDivide(p,x+1,x)
</pre>
<ul>
 <li>
  <input type="submit" id="p40" class="subbut"
    onclick="handleFree(['p6','p38','p40']);"
    value="p-((x+1)*qr.quotient+qr.remainder)" />
  <div id="ansp40"><div></div></div>
 </li>
</ul>
If the <a href="dbopdivide.xhtml">/</a> operator is used with polynomials,
a fraction object is created. In this example, the result is an object of
type
<a href="dbfractionpolynomialinteger.xhtml">Fraction Polynomial Integer</a>.
<ul>
 <li>
  <input type="submit" id="p41" class="subbut"
    onclick="handleFree(['p6','p7','p41']);"
    value="p/q" />
  <div id="ansp41"><div></div></div>
 </li>
```

```
</ul>
If you use rational numbers as polynomial coefficients, the resulting
object is of type
<a href="dbpolynomialfractioninteger.xhtml">Polynomial Fraction Integer</a>
<ul>
 <li>
  <input type="submit" id="p42" class="subbut"
    onclick="makeRequest('p42');"
    value="pfi:=(2/3)*x^2-y+4/5" />
  <div id="ansp42"><div></div></div>
 </li>
</ul>
This can be converted to a fraction of polynomials and back again, if
required.
<ul>
 <li>
  <input type="submit" id="p43" class="subbut"
    onclick="handleFree(['p42','p43']);"
    value="fpi:=pfi::FRAC POLY INT" />
  <div id="ansp43"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p44" class="subbut"
    onclick="handleFree(['p42','p43','p44']);"
    value="fpi::POLY FRAC INT" />
  <div id="ansp44"><div></div></div>
 </li>
</ul>
To convert the coefficients to floating point, map the
<a href="dbopnumeric.xhtml">numeric</a> operation on the coefficients
of the polynomial.
<ul>
 <li>
  <input type="submit" id="p45" class="subbut"
    onclick="handleFree(['p42','p45']);"
    value="map(numeric,pfi)" />
  <div id="ansp45"><div></div></div>
 </li>
</ul>
For more information on related topcis, see
<a href="axbook/section-9.83.xhtml">UnivariatePolynomial</a>,
<a href="axbook/section-9.54.xhtml">MultivariatePolynomial</a>, and
<a href="axbook/section-9.16.xhtml">DistributedMultivariatePolynomial</a>.
You can also issue the system command
<ul>
 <li>
  <input type="submit" id="p46" class="subbut"
    onclick="showcall('p46');"
   value=")show Polynomial"/>
  <div id="ansp46"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?Polynomial">Polynomial</a>.
```

```
\getchunk{page foot}
```

———-

polyspecifictypes2.xhtml

— polyspecifictypes2.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">UnivariatePolynomial</div>
  <hr/>
The domain constructor
<a href="db.xhtml?UnivariatePolynomial">UnivariatePolynomial</a>
(abbreviated <a href="db.xhtml?UnivariatePolynomial">UP</a>)
creates domains of univariate polynomials in a specified variable.
For example, the domain UP(a1,POLY FRAC INT) provides polynomials in
the single variable a1 whose coefficients are general polynomials with
rational number coefficients.
<hr/>
<b>Restriction:</b><br/>
Axiom does not allow you to create types where
<a href="db.xhtml?UnivariatePolynomial">UnivariatePolynomial</a>
is contained in the coefficient type of
<a href="db.xhtml?Polynomial">Polynomial</a>.
Therefore, UP(x,POLY INT) is legal but POLY UP(x,INT) is not.
<hr/>
UP(x,INT) is the domain of polynomials in the single variable x with
integer coefficients.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="(p,q):UP(x,INT)" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="makeRequest('p2');"
    value="p:=(3*x-1)^2*2*(2*x+8)" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="q:=(1-6*x+9*x^2)^2" />
  <div id="ansp3"><div></div></div>
```

```
 </li>
</ul>
The usual arithmetic operations are available for univariate polynomials.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4']);"
    value="p^2+p*q" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
The operation
<a href="dbopleadingcoefficient.xhtml">leadingCoefficient</a>
extracts the coefficient of the term of highest degree.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p5']);"
    value="leadingCoefficient p" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
The operation <a href="dbopdegree.xhtml">degree</a> returns the degree of
the polynomial. Since the polynomial has only one variable, the variable
is not supplied to operations like <a href="dbopdegree.xhtml">degree</a>.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p2','p6']);"
    value="degree p" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
The reductum of the polynomial, the polynomial obtained by subtracting
the term of highest order, is returned by
<a href="dbopreductum.xhtml">reductum</a>.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p7']);"
    value="reductum p" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
The operation <a href="dbopgcd.xhtml">gcd</a> computes the greatest common
divisor of two polynomials.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p3','p8']);"
    value="gcd(p,q)" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
```

```
The operation <a href="dboplcm.xhtml">lcm</a> computes the least common
multiple.
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="handleFree(['p1','p2','p3','p9']);"
    value="lcm(p,q)" />
  <div id="ansp9"><div></div></div>
 </li>
</ul>
The operation <a href="dbopresultant.xhtml">resultant</a> computes the
resultant of two univariate polynomials. In the case of p and q, the
resultant is 0 because they share a common root.
<ul>
 <li>
  <input type="submit" id="p10" class="subbut"
    onclick="handleFree(['p1','p2','p3','p10']);"
    value="resultant(p,q)" />
  <div id="ansp10"><div></div></div>
 </li>
</ul>
To compute the derivative of a univariate polynomial with respect to
its variable, use <a href="dbopd.xhtml">D</a>.
<ul>
 <li>
  <input type="submit" id="p11" class="subbut"
    onclick="handleFree(['p1','p2','p11']);"
    value="D p" />
  <div id="ansp11"><div></div></div>
 </li>
</ul>
Univariate polynomials can also be used as if they were functions.
To evaluate a univariate polynomial at some point, apply the polynomial
to the point.
<ul>
 <li>
  <input type="submit" id="p12" class="subbut"
    onclick="handleFree(['p1','p2','p12']);"
    value="p(2)" />
  <div id="ansp12"><div></div></div>
 </li>
</ul>
The same syntax is used for composing two univariate polynomials, i.e.
substituting one polynomial for the variable in another. This substitutes q
for the variable in p.
<ul>
 <li>
  <input type="submit" id="p13" class="subbut"
    onclick="handleFree(['p1','p2','p3','p13']);"
    value="p(q)" />
  <div id="ansp13"><div></div></div>
 </li>
</ul>
This substitutes p for the variable in q.
```

```
<ul>
 <li>
  <input type="submit" id="p14" class="subbut"
    onclick="handleFree(['p1','p2','p3','p14']);"
    value="q(p)" />
  <div id="ansp14"><div></div></div>
 </li>
</ul>
To obtain a list of coefficients of the polynomial, use
<a href="dbopcoefficients.xhtml">coefficients</a>.
<ul>
 <li>
  <input type="submit" id="p15" class="subbut"
    onclick="handleFree(['p1','p2','p15']);"
    value="l:=coefficients p" />
  <div id="ansp15"><div></div></div>
 </li>
</ul>
From this you can use <a href="dbopgcd.xhtml">gcd</a> and
<a href="dbopreduce.xhtml">reduce</a> to compute the contents of the
polynomial.
<ul>
 <li>
  <input type="submit" id="p16" class="subbut"
    onclick="handleFree(['p1','p2','p15','p16']);"
    value="reduce(gcd,l)" />
  <div id="ansp16"><div></div></div>
 </li>
</ul>
Alternatively (and more easily), you can just call
<a href="dbopcontent.xhtml">content</a>
<ul>
 <li>
  <input type="submit" id="p17" class="subbut"
    onclick="handleFree(['p1','p2','p17']);"
    value="content p" />
  <div id="ansp17"><div></div></div>
 </li>
</ul>
Note that the operation <a href="dbopcoefficients.xhtml">coefficients</a>
omits the zero coefficients from the list. Sometimes it is useful to
convert a univariate polynomial to a vector whose i-th position contains
the degree i-1 coefficient of the polynomial.
<ul>
 <li>
  <input type="submit" id="p18" class="subbut"
    onclick="makeRequest('p18');"
    value="ux:=(x^4+2*x+3)::UP(x,INT)" />
  <div id="ansp18"><div></div></div>
 </li>
</ul>
To get a complete vector of coefficients, use the operation
<a href="dbopvectorise.xhtml">vectorise</a>, which takes a univariate
polynomial and an integer denoting the length of the desired vector.
```

```
<ul>
 <li>
  <input type="submit" id="p19" class="subbut"
    onclick="handleFree(['p18','p19']);"
    value="vectorise(ux,5)" />
  <div id="ansp19"><div></div></div>
 </li>
</ul>
It is common to want to do something to every term of a polynomial,
creating a new polynomial in the process. This is a function for
iterating across the terms of a polynomial, squaring each term.
<ul>
 <li>
  <input type="submit" id="p20" class="subbut"
    onclick="makeRequest('p20');"
    value="squareTerms(m)==reduce(+,[t^2 for t in monomials m])" />
  <div id="ansp20"><div></div></div>
 </li>
</ul>
Recall what p looked like.
<ul>
 <li>
  <input type="submit" id="p21" class="subbut"
    onclick="handleFree(['p1','p2','p21']);"
    value="p" />
  <div id="ansp21"><div></div></div>
 </li>
</ul>
We can demonstrate squareTerms on p.
<ul>
 <li>
  <input type="submit" id="p22" class="subbut"
    onclick="handleFree(['p1','p2','p20','p22']);"
    value="squareTerms p" />
  <div id="ansp22"><div></div></div>
 </li>
</ul>
When the coefficients of the univariate polynomial belong to a field,
(for example, when the coefficients are rational numbers, as opposed to
integers. The important property of a field is that non-zero elements can
be divided and produce another element. The quotient of the integers 2 and 3
is not another integer.) It is possible to compute quotients and remainders.
<ul>
 <li>
  <input type="submit" id="p23" class="subbut"
    onclick="makeRequest('p23');"
    value="(r,s):UP(a1,FRAC INT)" />
  <div id="ansp23"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p24" class="subbut"
    onclick="handleFree(['p23','p24']);"
    value="r:=a1^2-2/3" />
  <div id="ansp24"><div></div></div>
```

```
 </li>
 <li>
  <input type="submit" id="p25" class="subbut"
    onclick="handleFree(['p23','p25']);"
    value="s:=a1+4" />
  <div id="ansp25"><div></div></div>
 </li>
</ul>
When the coefficients are rational numbers or rational expressions, the
operation <a href="dbopquo.xhtml">quo</a> computes the quotient of two
polynomials.
<ul>
 <li>
  <input type="submit" id="p26" class="subbut"
    onclick="handleFree(['p23','p24','p25','p26']);"
    value="r quo s" />
  <div id="ansp26"><div></div></div>
 </li>
</ul>
The operation <a href="dboprem.xhtml">rem</a> computes the remainder.
<ul>
 <li>
  <input type="submit" id="p27" class="subbut"
    onclick="handleFree(['p23','p24','p25','p27']);"
    value="r rem s" />
  <div id="ansp27"><div></div></div>
 </li>
</ul>
The operation <a href="dbopdivide.xhtml">divide</a> can be used to return
a record of both components.
<ul>
 <li>
  <input type="submit" id="p28" class="subbut"
    onclick="handleFree(['p23','p24','p25','p28']);"
    value="d:=divide(r,s)" />
  <div id="ansp28"><div></div></div>
 </li>
</ul>
Now we check the arithmetic.
<ul>
 <li>
  <input type="submit" id="p29" class="subbut"
    onclick="handleFree(['p23','p24','p25','p28','p29']);"
    value="r-(d.quotient*s+d.remainder)" />
  <div id="ansp29"><div></div></div>
 </li>
</ul>
It is also possible to integrate univariate polynomials when the
coefficients belong to a field.
<ul>
 <li>
  <input type="submit" id="p30" class="subbut"
    onclick="handleFree(['p23','p24','p30']);"
    value="integrate r" />
```

```
  <div id="ansp30"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p31" class="subbut"
    onclick="handleFree(['p23','p25','p31']);"
    value="integrate s" />
  <div id="ansp31"><div></div></div>
 </li>
</ul>
One application of univariate polynomials is to see expressions in terms of
a specific variable. We start with a polynomial in a1 whose coefficients are
quotients of polynomials in b1 and b2.
<ul>
 <li>
  <input type="submit" id="p32" class="subbut"
    onclick="makeRequest('p32');"
    value="t:UP(a1,FRAC POLY INT)" />
  <div id="ansp32"><div></div></div>
 </li>
</ul>
Since in this case we are not talking about using multivariate polynomials
in only two variables, we use <a href="db.xhtml?Polynomial">Polynomial</a>.
We also use <a href="db.xhtml?Fraction">Fraction</a> because we want fractions.
<ul>
 <li>
  <input type="submit" id="p33" class="subbut"
    onclick="handleFree(['p32','p33']);"
    value="t:=a1^2-a1/b2+(b1^2-b1)/(b2+3)" />
  <div id="ansp33"><div></div></div>
 </li>
</ul>
We push all the variables into a single quotient of polynomials.
<ul>
 <li>
  <input type="submit" id="p34" class="subbut"
    onclick="handleFree(['p32','p33','p34']);"
    value="u:FRAC POLY INT:=t" />
  <div id="ansp34"><div></div></div>
 </li>
</ul>
Alternatively, we can view this as a polynomial in the variable. This is a
mode-directed conversion: You indicate as much of the structure as you care
about and let Axiom decide on the full type and how to do the transformation.
<ul>
 <li>
  <input type="submit" id="p35" class="subbut"
    onclick="handleFree(['p32','p33','p34','p35']);"
    value="u::UP(b1,?)" />
  <div id="ansp35"><div></div></div>
 </li>
</ul>
See <a href="axbook/section-8.2.xhtml">Polynomial Factorization</a> for a
discussion of the factorization facilities in Axiom for univariate
polynomials. For more information on related topics, see
```

```
<a href="axbook/section-1.8.xhtml">Polynomials</a>,
<a href="axbook/section-2.7.xhtml">Conversion</a>,
<a href="polyspecifictypes1.xhtml">Polynomial</a>,
<a href="polyspecifictypes3.xhtml">MultivariatePolynomial</a>, and
<a href="polyspecifictypes4.xhtml">DistributedMultivariatePolynomial</a>.
Issue the system command
<ul>
 <li>
  <input type="submit" id="p36" class="subbut"
    onclick="showcall('p36');"
   value=")show UnivariatePolynomial"/>
  <div id="ansp36"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?UnivariatePolynomial">UnivariatePolynomial</a>.
\getchunk{page foot}
```

polyspecifictypes3.xhtml

— polyspecifictypes3.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">MultivariatePolynomial</div>
  <hr/>
The domain constructor
<a href="db.xhtml?MultivariatePolynomial">MultivariatePolynomial</a> is
similar to <a href="db.xhtml?Polynomial">Polynomial</a> except that it
specifies the variables to be used.
<a href="db.xhtml?Polynomial">Polynomial</a> are available for
<a href="db.xhtml?MultivariatePolynomial">MultivariatePolynomial</a>.
The abbreviation for
<a href="db.xhtml?MultivariatePolynomial">MultivariatePolynomial</a> is
<a href="db.xhtml?MultivariatePolynomial">MPOLY</a>. The type expressions
<pre>
   MultivariatePolynomial([x,y],Integer)
</pre>
and
<pre>
         MPOLY([x,y],INT)
</pre>
refer to the domain of multivariate polynomials in the variables x and y
where the coefficients are restricted to be integers. The first variable
specified is the main variable and the display of the polynomial reflects
```

```
this. This polynomial appears with terms in descending powers of the
variable x.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="m:MPOLY([x,y],INT):=(x^2-x*y^3+3*y)^2" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
It is easy to see a different variable ordering by doing a conversion.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="m::MPOLY([y,x],INT)" />
  <div id="ansp2"><div></div></div>
 </li>
</ul>
You can use other, unspecified variables, by using
<a href="db.xhtml?Polynomial">Polynomial</a> in the coefficient type of
<a href="db.xhtml?MultivariatePolynomial">MPOLY</a>.
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="makeRequest('p3');"
    value="p:MPOLY([x,y],POLY INT):=(a^2*x-b*y^2+1)^2" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Conversions can be used to re-express such polynomials in terms of the
other variables. For example, you can first push all the variables into a
polynomial with integer coefficients.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p3','p4']);"
    value="u:=p::POLY INT" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
Now pull out the variables of interest.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p3','p4','p5']);"
    value="u::MPOLY([a,b],POLY INT)" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
<hr/>
<b>Restriction:</b> Axiom does not allow you to create types where
<a href="db.xhtml?MultivariatePolynomial">MultivariatePolynomial</a> is
contained in the coefficient type of
```

```
<a href="db.xhtml?Polynomial">Polynomial</a>. Therefore,
<pre>
     MPOLY([x,y],POLY INT)
</pre>
is legal but this is not:
<pre>
     POLY MPOLY([x,y],INT)n
</pre>
<hr/>
Multivariate polynomials may be combined with univariate polynomials to
create types with special structures.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="q:UP(x,FRAC MPOLY([y,z],INT)):=(x^2-x*(z+1)/y+2)^2" />
  <div id="ansp6"><div></div></div>
 </li>
</ul>
This is a polynomial in x whose coefficients are quotients of polynomials
in y and z. Use conversions for the structural rearrangements. z does not
appear in a denominator and so it can be made the main variable.
<ul>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p6','p7']);"
    value="q::UP(z,FRAC MPOLY([x,y],INT))" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Or you can make a multivariate polynomial in x and z whose coefficients
are fractions in polynomials in y
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p6','p8']);"
    value="q::MPOLY([x,z],FRAC UP(y,INT))" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
A conversion like
<pre>
  q::MPOLY([x,y],FRAC UP(z,INT))
</pre>
is not possible in this example because y appears in the denominator of
a fraction. As you can see, Axiom provides extraordinary flexibility in
the manipulation and display of expressions via its conversion facility.

For more information on related topics, see
<a href="polyspecifictypes1.xhtml">Polynomial</a>,
<a href="polyspecifictypes2.xhtml">UnivariatePolynomial</a>, and
<a href="polyspecifictypes4.xhtml">DistributedMultivariatePolynomial</a>.
Issue the system command
<ul>
```

```
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="showcall('p9');"
   value=")show MultivariatePolynomial"/>
  <div id="ansp9"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?MultivariatePolynomial">MultivariatePolynomial</a>.
\getchunk{page foot}
```

polyspecifictypes4.xhtml

— polyspecifictypes4.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">DistributedMultivariatePolynomial</div>
  <hr/>
<a href="db.xhtml?DistributedMultivariatePolynomial">
DistributedMultivariatePolynomial</a> and
<a href="db.xhtml?HomogeneousDistributedMultivariatePolynomial">
HomogeneousDistributedMultivariatePolynomial</a>, abbreviated
<a href="db.xhtml?DistributedMultivariatePolynomial">DMP</a> and
<a href="db.xhtml?HomogeneousDistributedMultivariatePolynomial">HDMP</a>
repspectively, are very similar to
<a href="db.xhtml?MultivariatePolynomial">MultivariatePolynomial</a>
except that they are represented and displayed in a non-recursive manner.
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="(d1,d2,d3):DMP([z,y,x],FRAC INT)" />
  <div id="ansp1"><div></div></div>
 </li>
</ul>
The construction
<a href="db.xhtml?DistributedMultivariatePolynomial">DMP</a> orders its
monomials lexicographically while
<a href="db.xhtml?HomogeneousDistributedMultivariatePolynomial">HDMP</a>
orders them by total order refined by reverse lexicographic order.
<ul>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
```

```
    value="d1:=-4*z+4*y^2*x+16*x^2+1" />
  <div id="ansp2"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="d2:=2*z*y^2+4*x+1" />
  <div id="ansp3"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="handleFree(['p1','p4']);"
    value="d3:=2*z*x^2-2*y^2-x" />
  <div id="ansp4"><div></div></div>
 </li>
</ul>
These constructors are mostly used in Groebner basis calculations.
<ul>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p5']);"
    value="groebner [d1,d2,d3]" />
  <div id="ansp5"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="makeRequest('p6');"
    value="(n1,n2,n3):HDMP([z,y,x],FRAC INT)" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p6','p7']);"
    value="(n1,n2,n3):=(d1,d2,d3)" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
Note that we get a different Groebner basis when we use the
<a href="db.xhtml?HomogeneousDistributedMultivariatePolynomial">HDMP</a>
polynomials, as expected.
<ul>
 <li>
  <input type="submit" id="p8" class="subbut"
    onclick="handleFree(['p1','p2','p3','p4','p6','p7','p8']);"
    value="groebner [n1,n2,n3]" />
  <div id="ansp8"><div></div></div>
 </li>
</ul>
<a href="db.xhtml?GeneralDistributedMultivariatePolynomial">
GeneralDistributedMultivariatePolynomial</a> is somewhat more flexible in
the sense that as well as accepting a list of variables to specify the
variable ordering, it also takes a predicate on exponent vectors to specify
the term ordering. With this polynomial type the user can experiment with
the effect of using completely arbitrary term orderings. This flexibility
```

```
is mostly important for algorithms such as Groebner basis calculations
which can be very sensitive to term orderings.

For more information on related topics, see
<a href="axbook/section-1.8.xhtml">Polynomials</a>,
<a href="axbook/section-2.7.xhtml">Conversion</a>,
<a href="polyspecifictypes1.xhtml">Polynomial</a>,
<a href="polyspecifictypes2.xhtml">UnivariatePolynomial</a>. and
<a href="polyspecifictypes3.xhtml">MultivariatePolynomial</a>,
Issue the system command
<ul>
 <li>
  <input type="submit" id="p9" class="subbut"
    onclick="showcall('p9');"
   value=")show DistributedMultivariatePolynomial"/>
  <div id="ansp9"><div></div></div>
 </li>
</ul>
to display the full list of operations defined by
<a href="db.xhtml?DistributedMultivariatePolynomial">
DistributedMultivariatePolynomial</a> and
\getchunk{page foot}
```

polysubstitutions.xhtml

— polysubstitutions.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
\getchunk{handlefreevars}
\getchunk{axiom talker}
  </script>
 </head>
 <body onload="resetvars();">
\getchunk{page head}
  <div align="center">Polynomial Evaluation and Substitution</div>
  <hr/>
The function <a href="dbopeval.xhtml">eval</a> is used to substitute values
into polynomials. Here's an example of how to use it:
<ul>
 <li>
  <input type="submit" id="p1" class="subbut"
    onclick="makeRequest('p1');"
    value="p:=x^2+y^2" />
  <div id="ansp1"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p2" class="subbut"
    onclick="handleFree(['p1','p2']);"
    value="eval(p,x=5)" />
  <div id="ansp2"><div></div></div>
```

```
 </li>
</ul>
This example would give you the value of the polynomial p at 5. You can
also substitute into polynomials with several variables. First, specify
the polynomial, then give a list of the bindings of the form
<pre>
  variable = value
</pre>
For examples:
<ul>
 <li>
  <input type="submit" id="p3" class="subbut"
    onclick="handleFree(['p1','p3']);"
    value="eval(p,[x=a+b,y=c+d])" />
  <div id="ansp3"><div></div></div>
 </li>
</ul>
Here x was replaced by a+b, and y was replaced by c+d.
<ul>
 <li>
  <input type="submit" id="p4" class="subbut"
    onclick="makeRequest('p4');"
    value="q:=x^3+5*x-y^4" />
  <div id="ansp4"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p5" class="subbut"
    onclick="handleFree(['p4','p5']);"
    value="eval(q,[x=y,y=x])" />
  <div id="ansp5"><div></div></div>
 </li>
</ul>
Substitution is done "in parallel". That is, Axiom takes q(x,y) and
returns q(y,x).

You can also substitute numerical values for some or all of the variables.
<ul>
 <li>
  <input type="submit" id="p6" class="subbut"
    onclick="handleFree(['p1','p6']);"
    value="px:=eval(p,y=sin(2.0))" />
  <div id="ansp6"><div></div></div>
 </li>
 <li>
  <input type="submit" id="p7" class="subbut"
    onclick="handleFree(['p1','p6','p7']);"
    value="eval(px,x=cos(2.0))" />
  <div id="ansp7"><div></div></div>
 </li>
</ul>
\getchunk{page foot}
```

puiseuxseries.xhtml

— puiseuxseries.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function').value;
    myivar = document.getElementById('ivar').value;
    mypvar = document.getElementById('pvar').value;
    myevar = document.getElementById('evar').value;
    myival = document.getElementById('ival').value;
    mysval = document.getElementById('sval').value;
    ans = 'series('+myivar+'+->'+myfunc+','+mypvar+'='+myevar+','+
         myival+'..,'+mysval+')';
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
      Enter the formula for the general coefficient of the series:
    </td>
   </tr>
   <tr>
    <td>
     <input type="text" id="function" size="80" tabindex="10"
       value="(-1)^((3*n-4)/6)/factorial(n-1/3)"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the index variable for your formula:
     <input type="text" id="ivar" size="10" tabindex="20" value="n"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the power series variable:
     <input type="text" id="pvar" size="10" tabindex="30" value="x"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the point about which to expand:
     <input type="text" id="evar" size="10" tabindex="40" value="0"/>
    </td>
```

```
   </tr>
  </table>
For Puiseux Series, the exponent of the power series variable ranges
from an initial value, an arbitrary rational number, to plus
infinity; the step size is any positive rational number.
  <table>
   <tr>
    <td>
     Enter the initial value of the index (a rational number):
     <input type="text" id="ival" size="10" tabindex="50" value="4/3"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the step size (a positive rational number):
     <input type="text" id="sval" size="10" tabindex="60" value="2"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

reallimit.xhtml

— reallimit.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    var myform = document.getElementById("form2");
    var myfunct = myform.expr.value;
    var myvar = myform.vars.value;
    var mypoint = "";
    // decide what the limit point should be
    var finite = document.getElementById('finite').checked;
    if (finite == true)
      mypoint = document.getElementById('fpoint').value;
    if (document.getElementById('plus').checked == true)
      mypoint = "%plusInfinity";
    if (document.getElementById('minus').checked == true)
      mypoint = "%minusInfinity";
    // decide what the limit statement is
    if (document.getElementById('both').checked == true)
      ans = 'limit('+myform.expr.value+','+myvar+'='+mypoint+')';
    // note: ignore direction if limit is %plusInfinity
    if (document.getElementById('right').checked == true) {
     if (finite == true) {
       ans = 'limit('+myform.expr.value+','+myvar+'='+mypoint+',"right")';
     } else {
```

```
       ans = 'limit('+myform.expr.value+','+myvar+'='+mypoint+')';
     };
    };
    // note: ignore direction if limit is %minutInfinity
    if (document.getElementById('left').checked == true) {
     if (finite == true) {
       ans = 'limit('+myform.expr.value+','+myvar+'='+mypoint+',"left")';
     } else {
       ans = 'limit('+myform.expr.value+','+myvar+'='+mypoint+')';
     };
    };
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <form id="form2">
   Enter the function you want to compute the limit of:<br/>
   <input type="text" id="expr" tabindex="10" size="50"
     value="x*sin(1/x)"/><br/>
   Enter the name of the variable:<br/>
   <input type="text" id="vars" tabindex="20" value="x"/><br/>
   <input type="radio" id="finite" tabindex="30" checked="checked"
     name="point"/>
    A finite point
    <input type="text" id="fpoint" tabindex="20" value="0"/><br/>
   <input type="radio" id="plus" tabindex="40" name="point"/>
    %plusInfinity<br/>
   <input type="radio" id="minus" tabindex="50" name="point"/>
    %minusInfinity<br/><br/><br/>
   Compute the limit from:<br/>
   <input type="radio" id="both" tabindex="60" name="direction"
     checked="checked"/>
    both directions<br/>
   <input type="radio" id="right" tabindex="70" name="direction"/>
    the right<br/>
   <input type="radio" id="left" tabindex="80" name="direction"/>
    the left<br/>
  </form>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

refsearchpage.xhtml

— refsearchpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      refsearchpage not implemented
\getchunk{page foot}
```

releasenotes.xhtml

— releasenotes.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
The <b>November 2007</b> release of Axiom contains
<ul>
 <li>
   New MathML output mode. This mode allows Axiom to output expressions
   using standard MathML format. This complements the existing ability
   to output Fortran, IBM script, Latex, OpenMath, and algebra formats.
 </li>
 <li>
   Ninety-five domains have been documented for the )help command.
   Type )help to see the list.
 </li>
 <li>
   New regression tests were added to improve the release testing.
 </li>
 <li>
   Hyperdoc can now be restarted. Type )hd
 </li>
 <li>
   Testing has begun against Spiegel's Mathematical Handbook from the
   Schaum's Outline Series. These tests include Axiom's solutions and
   have uncovered mistakes in the published text.
 </li>
</ul>
Bug fixes
<ul>
 <li>
   <b>Bug100</b> integrate((z^a+1)^b,z) no longer loops infinitely.
 </li>
 <li>
   <b>Bug101</b> laplace(log(z),z,w) returns "failed" instead of crashing.
 </li>
 <li>
   <b>Bug103</b> solve(z=z,z) returns the correct answer
 </li>
</ul>
Additional information sources:
```

```
<table>
 <tr>
  <td>
   <a href="http://axiom.axiom-developer.org">
    <b>Online information is available here</b>
   </a>
  </td>
 </tr>
 <tr>
  <td>
   <a href="CHANGELOG.xhtml">
    The changelog file contains specific file-by-file changes.
   </a>
  </td>
 </tr>
</table>
\getchunk{page foot}
```

rootpage.xhtml

```
books/tanglec books/bookvol11.pamphlet rootpage.xhtml > rootpage.xhtml
```

— **rootpage.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <b>What would you like to do?</b><br/><br/>
  <table>
   <tr>
    <td>
     <a href="commandline.xhtml">
      <b>Any Command</b>
     </a>
    </td>
    <td>Try command line input</td>
   </tr>
   <tr>
    <td>
     <a href="basiccommand.xhtml">
      <b>Basic Commands</b>
     </a>
    </td>
    <td>Solve problems by filling in templates</td>
   </tr>
   <tr>
    <td>
     <a href="jenks.xhtml">
      <b>Axiom Textbook</b>
     </a>
    </td>
```

```
 <td>Read Volume 0 -- The Jenks/Sutor Book</td>
</tr>
<tr>
 <td>
  <a href="tutorial.xhtml">
   <b>Axiom Tutorial</b>
  </a>
 </td>
 <td>Read Volume 1 -- The Tutorial</td>
</tr>
<tr>
 <td>
  <a href="topreferencepage.xhtml">
   <b>Reference</b>
  </a>
 </td>
 <td>Scan on-line documentation for AXIOM<br/></td>
</tr>
<tr>
 <td>
  <a href="topicspage.xhtml">
   <b>Topics</b>
  </a>
 </td>
 <td> Learn how to use Axiom, by topic<br/></td>
</tr>
<tr>
 <td>
  <a href="man0page.xhtml">
   <b>Browser</b>
  </a>
 </td>
 <td> Browse through the AXIOM library<br/></td>
</tr>
<tr>
 <td>
  <a href="topexamplepage.xhtml">
   <b>Examples</b>
  </a>
 </td>
 <td> See examples of use of the library<br/></td>
</tr>
<tr>
 <td>
  <a href="topsettingspage.xhtml">
   <b>Settings</b>
  </a>
 </td>
 <td> Display and change the system environment<br/></td>
</tr>
<tr>
 <td>
  <a href="releasenotes.xhtml">
   <b>What's New</b>
```

```
    </a>
   </td>
   <td> Enhancements in this version of Axiom<br/></td>
  </tr>
  <tr>
   <td>
    <a href="axiomfonts.xhtml">
     <b>Fonts</b>
    </a>
   </td>
   <td> Test Axiom Fonts in your Browser<br/></td>
  </tr>
 </table>
\getchunk{page foot}
```

series.xhtml

— series.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  Create a series by
  <table>
   <tr>
    <td width="100">
     <a href="seriesexpand.xhtml">
      <b>Expansion</b>
     </a>
    </td>
    <td>
     Expand a function in a series around a point
    </td>
   </tr>
   <tr>
    <td width="100">
     <a href="taylorseries.xhtml">
      <b>Taylor Series</b>
     </a>
    </td>
    <td><br/>
     Series where the exponent ranges over the integers from a
     non-negative integer value to plus infinity by an arbitrary
     positive integer step size.
    </td>
   </tr>
   <tr>
    <td width="100">
     <a href="laurentseries.xhtml">
      <b>Laurent Series</b>
```

```
     </a>
    </td>
    <td><br/>
     Series where the exponent ranges from an arbitrary integer value
     to plus infinity by an arbitrary positive integer step size.
    </td>
   </tr>
   <tr>
    <td width="100">
     <a href="puiseuxseries.xhtml">
      <b>Puiseux Series</b>
     </a>
    </td>
    <td><br/>
     Series where the exponent ranges from an arbitrary rational value
     to plus infinity by an arbitrary positive rational number step size.
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

seriesexpand.xhtml

— seriesexpand.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function').value;
    myvar = document.getElementById('var').value;
    mypoint = document.getElementById('point').value;
    ans = 'series('+myfunc+','+myvar+'='+mypoint+')';
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
      What function would you like to expand in a power series?
    </td>
   </tr>
   <tr>
    <td>
     <input type="text" id="function" size="80" tabindex="10"
       value="log(cot(x))"/>
```

```
    </td>
   </tr>
   <tr>
    <td>
     Enter the power series variable:
     <input type="text" id="var" size="10" tabindex="20" value="x"/>
    </td>
   </tr>
   <tr>
    <td>
     Expand around the point:
     <input type="text" id="point" size="10" tabindex="30" value="%pi/2"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

solve.xhtml

— solve.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
 What do you want to solve?
  <table>
   <tr>
    <td>
     <a href="solvelinearequations.xhtml">
      A System of Linear Equations in equation form
     </a>
    </td>
   </tr>
   <tr>
    <td>
     <a href="solvelinearmatrix.xhtml">
      A System of Linear Equations in matrix form
     </a>
    </td>
   </tr>
   <tr>
    <td>
     <a href="solvesystempolynomials.xhtml">
      A System of Polynomial Equations
     </a>
    </td>
   </tr>
```

```
   <tr>
    <td>
     <a href="solvesinglepolynomial.xhtml">
      A Single Polynomial Equation
     </a>
    </td>
   </tr>
  </table>
\getchunk{page foot}
```

———-

solvelinearequations.xhtml

— solvelinearequations.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
<![CDATA[
   function indeps(i) {
    var ans="";
    for (var j = 0 ; j < i ; j++) {
     ans=ans+'x'+j
     if (j != (i - 1)) ans=ans+',';
    }
    return(ans);
   }
   function equation(i) {
    var ans="";
    for (var j = 0 ; j < i ; j++) {
     ans=ans+Math.floor(Math.random()*100)+'*x'+j;
     if (j != (i - 1)) ans=ans+'+';
    }
    ans=ans+"="+Math.floor(Math.random()*100);
    return(ans);
   }
   function byelement() {
      // find out how many rows and columns, must be positive and nonzero
    var rcnt = parseInt(document.getElementById('rowcnt').value);
    if (rcnt <= 0) {
      alert("Rows must be positive and non-zero -- defaulting to 1");
      rcnt = 1;
      document.getElementById('rowcnt').value=1;
      return(false);
    }
      // remove the question and the buttons
    var quest = document.getElementById('question');
    var clicks = document.getElementById('clicks');
    quest.removeChild(clicks);
      // write "Elements"
    var tbl = document.getElementById('form2');
    var tblsize = tbl.rows.length;
    var row = tbl.insertRow(tblsize);
```

```
 var thecell = row.insertCell(0);
 var tnode = document.createTextNode("Enter the equations:");
 thecell.appendChild(tnode);
   // create input boxes for the matrix values
 for (var i = 0 ; i < rcnt ; i++) {
  tblsize = tblsize + 1;
  row = tbl.insertRow(tblsize);
  thecell = row.insertCell(0);
  tnode = document.createTextNode('equation '+i+': ');
  thecell.appendChild(tnode);
  thecell = row.insertCell(1);
  tnode = document.createElement('input');
  tnode.type = 'text';
  tnode.name = 'a'+i;
  tnode.id = 'a'+i;
  tnode.size=50;
  tnode.value=equation(rcnt);
  tnode.tabindex=20+i;
  thecell.appendChild(tnode);
 }
   // insert the request for the unknown
 tblsize = tblsize + 1;
 row = tbl.insertRow(tblsize);
 thecell = row.insertCell(0);
 tnode = document.createTextNode("Enter the unknowns (comma separated):");
 thecell.appendChild(tnode);
 thecell = row.insertCell(1);
 tnode = document.createElement('input');
 tnode.type = 'text';
 tnode.name = 'unk';
 tnode.id = 'unk';
 tnode.size=10;
 tnode.value=indeps(rcnt);
 tnode.tabindex=2000;
 thecell.appendChild(tnode);
 tblsize = tblsize + 1;
   // insert a blank line
 row = tbl.insertRow(tblsize);
 thecell = row.insertCell(0);
 tnode = document.createTextNode("");
 thecell.appendChild(tnode);
   // insert the continue button
 var centnode = document.createElement('center');
 tbl.parentNode.appendChild(centnode);
 tnode = document.createElement('input');
 tnode.type = 'button';
 tnode.id = 'contbutton';
 tnode.value = 'Continue';
 tnode.setAttribute("onclick","makeRequest('');");
 centnode.appendChild(tnode);
 return(false);
}
function commandline(arg) {
  var rcnt = parseInt(document.getElementById('rowcnt').value);
```

```
     var cmdhead = 'solve(';
     var cmdtail = '])';
     var listbody = '[';
     for (var j = 0 ; j < rcnt ; j++) {
      var aj = document.getElementById('a'+j).value;
      listbody = listbody+aj;
      if (j != (rcnt - 1)) listbody = listbody+',';
     }
     listbody = listbody+']';
     cmdhead = cmdhead+listbody;
     var ans = cmdhead+',['+document.getElementById('unk').value+cmdtail;
     alert(ans);
     return(ans);
   }
]]>
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
 <table id="form2">
  <tr>
   <td>
    Enter the number of equations:
    <input type="text" id="rowcnt" tabindex="10" size="10" value="2"/>
   </td>
  </tr>
 </table>
 <div id="question">
  <div id="clicks">
   <center>
    <input type="button" value="Continue" onclick="byelement();"/>
   </center>
  </div>
 </div>
\getchunk{answer field}
\getchunk{page foot}
```

solvelinearmatrix.xhtml

— **solvelinearmatrix.xhtml** —

```
\getchunk{standard head}
  <script type="text/javascript">
<![CDATA[
   function byformula() {
      // find out how many rows and columns, must be positive and nonzero
    var rcnt = parseInt(document.getElementById('rowcnt').value);
     if (rcnt <= 0) {
      alert("Rows must be positive and non-zero -- defaulting to 1");
```

```
  rcnt = 1;
  document.getElementById('rowcnt').value=1;
  return(false);
 }
var ccnt = parseInt(document.getElementById('colcnt').value);
 if (ccnt <= 0) {
  alert("Columns must be positive and non-zero -- defaulting to 1");
  ccnt = 1;
  document.getElementById('colcnt').value=1;
  return(false);
 }
  // remove the question and the buttons
var quest = document.getElementById('question');
var clicks = document.getElementById('clicks');
quest.removeChild(clicks);
var tbl = document.getElementById('form2');
var tblsize = tbl.rows.length;
  // make the row variable question
  // row variable left cell
var row = tbl.insertRow(tblsize);
var cell = row.insertCell(0);
var tnode = document.createTextNode("Enter the row variable");
cell.appendChild(tnode);
  // row variable right cell
cell = row.insertCell(1);
tnode = document.createElement('input');
tnode.type = 'text';
tnode.name = 'rowvar';
tnode.id = 'rowvar';
tnode.size=10;
tnode.value='i';
tnode.tabindex=21;
cell.appendChild(tnode);
  // make the column variable question
  // column variable left cell
tblsize = tblsize + 1;
row = tbl.insertRow(tblsize);
cell = row.insertCell(0);
tnode = document.createTextNode("Enter the column variable");
cell.appendChild(tnode);
  // column variable right cell
cell = row.insertCell(1);
tnode = document.createElement('input');
tnode.type = 'text';
tnode.name = 'colvar';
tnode.id = 'colvar';
tnode.size=10;
tnode.tabindex=22;
tnode.value='j';
cell.appendChild(tnode);
  // make the formula question
  // column variable left cell
tblsize = tblsize + 1;
row = tbl.insertRow(tblsize);
```

```
 cell = row.insertCell(0);
 tnode = document.createTextNode("Enter the formulas for the elements");
 cell.appendChild(tnode);
   // formula input field
 tblsize = tblsize + 1;
 row = tbl.insertRow(tblsize);
 cell = row.insertCell(0);
 tnode = document.createElement('input');
 tnode.type = 'text';
 tnode.name = 'formula1';
 tnode.id = 'formula1';
 tnode.size=50;
 tnode.value = '1/(x-i-j-1)';
 tnode.tabindex=23;
 cell.appendChild(tnode);
 tblsize = tblsize + 1;
 row = tbl.insertRow(tblsize);
 cell = row.insertCell(0);
 tnode = document.createTextNode("Enter the vector, one per row:");
 cell.appendChild(tnode);
   // formula input field
 tblsize = tblsize + 1;
 row = tbl.insertRow(tblsize);
 cell = row.insertCell(0);
 tnode = document.createElement('input');
 tnode.type = 'text';
 tnode.name = 'vec1';
 tnode.id = 'vec1';
 tnode.size=70;
 tnode.value = '3,5';
 tnode.tabindex=24;
 cell.appendChild(tnode);
   // insert the continue button
 tblsize = tblsize + 1;
 row = tbl.insertRow(tblsize);
 cell = row.insertCell(0);
 tnode = document.createElement('input');
 tnode.type = 'button';
 tnode.id = 'contbutton';
 tnode.value = 'Continue';
 tnode.setAttribute("onclick","makeRequest('formula');");
 tnode.tabindex=24;
 cell.appendChild(tnode);
 return(false);
}
function byelement() {
   // find out how many rows and columns, must be positive and nonzero
 var rcnt = parseInt(document.getElementById('rowcnt').value);
  if (rcnt <= 0) {
   alert("Rows must be positive and non-zero -- defaulting to 1");
   rcnt = 1;
   document.getElementById('rowcnt').value=1;
   return(false);
  }
```

```
var ccnt = parseInt(document.getElementById('colcnt').value);
 if (ccnt <= 0) {
  alert("Columns must be positive and non-zero -- defaulting to 1");
  ccnt = 1;
  document.getElementById('colcnt').value=1;
  return(false);
 }
  // remove the question and the buttons
var quest = document.getElementById('question');
var clicks = document.getElementById('clicks');
quest.removeChild(clicks);
  // write "Elements"
var tbl = document.getElementById('form2');
var tblsize = tbl.rows.length;
var row = tbl.insertRow(tblsize);
var thecell = row.insertCell(0);
var tnode = document.createTextNode("Elements");
thecell.appendChild(tnode);
  // create input boxes for the matrix values
tblsize = tblsize + 1;
for (var i = 0 ; i < rcnt ; i++) {
 row = tbl.insertRow(tblsize);
 for (var j = 0 ; j < ccnt ; j++) {
  thecell = row.insertCell(j);
  tnode = document.createElement('input');
  tnode.type = 'text';
  tnode.name = 'a'+i+'c'+j;
  tnode.id = 'a'+i+'c'+j;
  tnode.size=10;
  tnode.tabindex=20+(i*10)+j;
  thecell.appendChild(tnode);
 }
  thecell = row.insertCell(j);
  tnode = document.createTextNode(' = ');
  thecell.appendChild(tnode);
  thecell = row.insertCell(j+1);
  tnode = document.createElement('input');
  tnode.type = 'text';
  tnode.name = 'k'+i;
  tnode.id = 'k'+i;
  tnode.size=10;
  tnode.value='0';
  tnode.tabindex=20+(i*10)+j+10;
  thecell.appendChild(tnode);
  tblsize = tblsize + 1;
}
  // insert a blank line
row = tbl.insertRow(tblsize);
thecell = row.insertCell(0);
tnode = document.createTextNode("");
thecell.appendChild(tnode);
  // insert the continue button
var centnode = document.createElement('center');
tbl.parentNode.appendChild(centnode);
```

```
 tnode = document.createElement('input');
 tnode.type = 'button';
 tnode.id = 'contbutton';
 tnode.value = 'Continue';
 tnode.setAttribute("onclick","makeRequest('element');");
 centnode.appendChild(tnode);
 return(false);
}
function commandline(arg) {
 if (arg == 'element') {
  var rcnt = parseInt(document.getElementById('rowcnt').value);
  var ccnt = parseInt(document.getElementById('colcnt').value);
   // get the right side vector into list form
  var vecbody = '[';
  var homogeneous = true;
  for (var k = 0 ; k < rcnt ; k++) {
    var ki = document.getElementById('k'+k).value;
      // is it homogeneous?
    if (parseInt(ki) != 0) homogeneous = false;
    vecbody = vecbody+ki;
    if (k != (rcnt - 1)) vecbody = vecbody+',';
  }
  vecbody = vecbody+']';
  alert('vecbody='+vecbody);
    // get the matrix elements, make them into lists of lists
  var listbody = '';
  for (var i = 0 ; i < rcnt ; i++) {
   var listbody = listbody+'[';
   for (var j = 0 ; j < ccnt ; j++) {
    var aij = document.getElementById('a'+i+'c'+j).value;
    listbody = listbody+aij;
    if (j != (ccnt - 1)) listbody = listbody+',';
   }
   listbody = listbody+']';
   if (i != (rcnt - 1)) listbody = listbody+',';
  }
  var matcmd = 'matrix(['+listbody+'])';
  alert('matcmd='+matcmd);
   // now we decide whether to compute the nullSpace or solve
  if (homogeneous == true)
    cmd = 'nullSpace('+matcmd+')';
  else
    cmd = 'solve('+matcmd+','+vecbody+')';
  alert(cmd);
  return(cmd);
 } else {
  var rcnt = parseInt(document.getElementById('rowcnt').value);
  var ccnt = parseInt(document.getElementById('colcnt').value);
  var vec = '['+document.getElementById('vec1').value+']';
  var cmdhead = 'matrix([[';
  var cmdtail = '])';
  var formula = document.getElementById('formula1').value;
  var rowv = document.getElementById('rowvar').value;
  var colv = document.getElementById('colvar').value;
```

```
     var cmd = cmdhead+formula+' for '+colv+' in 1..'+ccnt+']'+
                              ' for '+rowv+' in 1..'+rcnt+cmdtail;
     return(cmd);
    }
   }
]]>
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
Enter the size of the matrix:
<table id="form2">
 <tr>
  <td size="10">Rows</td>
  <td><input type="text" id="rowcnt" tabindex="10" size="10" value="2"/></td>
 </tr>
 <tr>
  <td>Columns</td>
  <td><input type="text" id="colcnt" tabindex="20" size="10" value="3"/></td>
 </tr>
</table>
<div id="question">
 <div id="clicks">
  How would you like to enter the matrix elements?
  <center>
   <input type="button" value="By Formula" onclick="byformula();"/>
   <input type="button" value="By Element" onclick="byelement();"/>
  </center>
 </div>
</div>
\getchunk{answer field}
\getchunk{page foot}
```

———

solvesinglepolynomial.xhtml

— solvesinglepolynomial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      solvesinglepolynomial.xhtml not implemented
\getchunk{page foot}
```

———

solvesystempolynomials.xhtml

— solvesystempolynomials.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
     solvesystempolynomials.xhtml  not implemented
\getchunk{page foot}
```

———-

summation.xhtml

— summation.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    var myform = document.getElementById("form2");
    return('sum('+myform.expr.value+','+myform.vars.value+'='+
                 myform.lower.value+'..'+myform.upper.value+')');
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <form id="form2">
   Enter the function you want to sum:<br/>
   <input type="text" id="expr" tabindex="10" size="50" value="i^3"/><br/>
   Enter the summation index:
   <input type="text" id="vars" tabindex="20" value="i" size="5"/><br/>
   Enter the limits of the sum: From:
   <input type="text" id="lower" tabindex="30" value="1" size="5"/>
   To:
   <input type="text" id="upper" tabindex="40" value="n" size="5"/><br/>
  </form>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

systemvariables.xhtml

— systemvariables.xhtml —

```
\getchunk{standard head}
```

```
 </head>
 <body>
\getchunk{page head}
      systemvariables not implemented
\getchunk{page foot}
```

taylorseries.xhtml

— taylorseries.xhtml —

```
\getchunk{standard head}
  <script type="text/javascript">
   function commandline(arg) {
    myfunc = document.getElementById('function').value;
    myivar = document.getElementById('ivar').value;
    mypvar = document.getElementById('pvar').value;
    myevar = document.getElementById('evar').value;
    myival = document.getElementById('ival').value;
    mysval = document.getElementById('sval').value;
    ans = 'series('+myivar+'+->'+myfunc+','+mypvar+'='+myevar+','+
         myival+'..,'+mysval+')';
    alert(ans);
    return(ans);
   }
\getchunk{showfullanswer}
\getchunk{axiom talker}
  </script>
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td>
      Enter the formula for the general coefficient of the series:
    </td>
   </tr>
   <tr>
    <td>
     <input type="text" id="function" size="80" tabindex="10"
       value="1/factorial(i)"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the index variable for your formula:
     <input type="text" id="ivar" size="10" tabindex="20" value="i"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the power series variable:
```

```
     <input type="text" id="pvar" size="10" tabindex="30" value="x"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the point about which to expand:
     <input type="text" id="evar" size="10" tabindex="40" value="0"/>
    </td>
   </tr>
  </table>
For Taylor Series, the exponent of the power series variable ranges
from an initial value, an arbitrary non-negative integer, to plus
infinity; the step size is any positive integer.
  <table>
   <tr>
    <td>
     Enter the initial value of the index (an integer):
     <input type="text" id="ival" size="10" tabindex="50" value="0"/>
    </td>
   </tr>
   <tr>
    <td>
     Enter the step size (a positive integer):
     <input type="text" id="sval" size="10" tabindex="60" value="1"/>
    </td>
   </tr>
  </table>
\getchunk{continue button}
\getchunk{answer field}
\getchunk{page foot}
```

———-

topexamplepage.xhtml

— topexamplepage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td><a href="graphicsexamplepage.xhtml"><b>Graphics</b></a></td>
    <td>Examples of Axiom Graphics</td>
   </tr>
   <tr>
    <td><a href="examplesexposedpage.xhtml"><b>Domains</b></a></td>
    <td>Examples of use of Axiom domains and packages</td>
   </tr>
   <tr>
    <td><a href="examplecoverpage.xhtml"><b>Operations</b></a></td>
```

```
    <td>Examples of Axiom Operations, by topic</td>
   </tr>
  </table>
\getchunk{page foot}
```

topicspage.xhtml

— topicspage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td><a href="numberspage.xhtml"><b>Numbers</b></a></td>
    <td>A look at different types of numbers</td>
   </tr>
   <tr>
    <td><a href="polynomialpage.xhtml"><b>Polynomials</b></a></td>
    <td>Polynomials in Axiom</td>
   </tr>
   <tr>
    <td><a href="functionpage.xhtml"><b>Functions</b></a></td>
    <td>Built-in and user-defined functions</td>
   </tr>
   <tr>
    <td><a href="equationpage.xhtml"><b>Solving Equations</b></a></td>
    <td>Facilities for solving equations</td>
   </tr>
   <tr>
    <td><a href="calculuspage.xhtml"><b>Calculus</b></a></td>
    <td>Using Axiom to do calculus</td>
   </tr>
   <tr>
    <td><a href="linalgpage.xhtml"><b>Linear Algebra</b></a></td>
    <td>Axiom's linear algebra facilities</td>
   </tr>
   <tr>
    <td><a href="graphicspage.xhtml"><b>Graphics</b></a></td>
    <td>Axiom's graphics facilities</td>
   </tr>
   <tr>
    <td><a href="algebrapage.xhtml"><b>Algebra</b></a></td>
    <td>Axiom's abstract algebra facilities</td>
   </tr>
   <tr>
    <td><a href="cryptopage.xhtml"><b>Cryptography</b></a></td>
    <td>Alasdair McAndrew's Crytography Course Notes</td>
   </tr>
   <tr>
```

```
    <td><a href="ocwmit18085.xhtml"><b>Mathematical Methods</b></a></td>
    <td>MIT 18-08 Mathematical Methods for Engineers Course Notes</td>
   </tr>
   <tr>
    <td><a href="cats.xhtml"><b>CATS</b></a></td>
    <td>Computer Algebra Test Suite</td>
   </tr>
  </table>
\getchunk{page foot}
```

———-

topreferencepage.xhtml

— **topreferencepage.xhtml** —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
  <table>
   <tr>
    <td><a href="axbook/book-contents.xhtml"><b>AXIOM Book</b></a></td>
    <td>The on-line version of the Jenks/Sutor book.</td>
   </tr>
   <tr>
    <td><a href="foundationlibrarydocpage.xhtml"><b>NAG Library</b></a></td>
    <td>The on-line NAG Library documentation.</td>
   </tr>
   <tr>
    <td><a href="topicspage.xhtml"><b>Topics</b></a></td>
    <td>Learn how to use Axiom, by topic.</td>
   </tr>
   <tr>
    <td><a href="uglangpage.xhtml"><b>Language</b></a></td>
    <td>Introduction to the Axiom language.</td>
   </tr>
   <tr>
    <td><a href="examplesexposedpage.xhtml"><b>Examples</b></a></td>
    <td>Examples for exposed domains and packages</td>
   </tr>
   <tr>
    <td><a href="ugsyscmdpage.xhtml"><b>Commands</b></a></td>
    <td>System commands that control your workspace.</td>
   </tr>
   <tr>
    <td><a href="operations.xhtml"><b>Operations</b></a></td>
    <td>A guide to useful operations</td>
   </tr>
   <tr>
     <td><a href="systemvariables.xhtml"><b>System Variables</b></a></td>
    <td>View and change a system-defined variable</td>
   </tr>
```

```
   <tr>
    <td><a href="glossarypage.xhtml"><b>Glossary</b></a></td>
    <td>A glossary of Axiom terms.</td>
   </tr>
   <tr>
    <td><a href="htxtoppage.xhtml"><b>HyperDoc</b></a></td>
    <td>How to write your own HyperDoc pages.</td>
   </tr>
   <tr>
    <td><a href="refsearchpage.xhtml"><b>Search</b></a></td>
    <td>Reference pages for occurrences of a string.</td>
   </tr>
  </table>
\getchunk{page foot}
```

topsettingspage.xhtml

— topsettingspage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
System commands are used to perform Axiom environment
management and change Axiom system variables.
  <hr/>
  <table>
   <tr>
    <td><a href="ugsyscmdpage.xhtml"><b>Commands</b></a></td>
    <td>System commands that control your environment.</td>
   </tr>
   <tr>
    <td><a href="htSystemVariables.js"><b>Settings</b></a></td>
    <td>Change an Axiom variable.</td>
   </tr>
  </table>
\getchunk{page foot}
```

tutorial.xhtml

— tutorial.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      tutorial not implemented
\getchunk{page foot}
```

uglangpage.xhtml

— uglangpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      uglangpage not implemented
\getchunk{page foot}
```

ugsyscmdpage.xhtml

— ugsyscmdpage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      ugsyscmdpage not implemented
\getchunk{page foot}
```

usersguidepage.xhtml

— usersguidepage.xhtml —

```
\getchunk{standard head}
 </head>
 <body>
\getchunk{page head}
      usersguidepage not implemented
\getchunk{page foot}
```

rcm3720.input

— rcm3720.input —

```
str2lst(str) == [ord(str.i)-65 for i in 1..#str]

lst2str(lst) == concat [char(lst.i+65)::String for i in 1..#lst]

str2num(str) ==
```

```
  local strlst
  strlst:=[ord(str.i) for i in 1..#str]
  return wholeRadix(strlst)$RadixExpansion(256)::INT

num2str(n) ==
  local tmp
  tmp:=wholeRagits(n::RadixExpansion(256))
  return concat [char(tmp.i)::String for i in 1..#tmp]

superIncreasing?(lst) ==
  reduce(/\\,[lst.i>reduce(+,[lst.j for j in 1..i-1]) for i in 2..#lst])

siSolve(lst,n) ==
  local res,m,i
  if not superIncreasing?(lst) then error "The list is not super-increasing"
  m := n
  res := [0 for i in 1..#lst]
  for i in #lst..1 by -1 repeat
    if lst.i <= m then
      res.i := 1
      m := m - lst.i
      if m = 0 then return res
  error "Unsolvable"

subsetsum(L:List(INT),N:INT):List(INT) ==
  local x,Y
  if N=0 then return([])
  if N<0 or #L=0 then return([-1])
  for x in L repeat
    Y:=subsetsum(remove(x,L),N)
    if Y~=[-1] then return(Y)
    Y:=subsetsum(remove(x,L),N-x)
    if Y~=[-1] then return(cons(x,Y))
    return([-1])
```

signatures.txt

— signatures.txt —

```
RSA ---
n = 2^137-1 e = 17
message = "This is my text."
signature = 68767027465671577191073128495082795700768
n = (6^67-1)/5 e = 17
message = "Please feed my dog!"
signature = 1703215098456351993605104919259566435843590978852633

Rabin -----
n = (3^59-1)/2
message = "Leave now."
signature =
```

```
n = (7^47-1)/6
message = "Arrive Thursday."
signature = 189479723122534414019783447271411895509

El Gamal --------
p = next prime after 2^150
a = 2
B = 1369851585774063312693119161120024351761244461
message = "Leave AT ONCE!"
signature r = 1389080525305754392111976715361069425353578198
s = 1141326468070168229982976133801721430306004477

DSS ---
p = next prime after 2^170
q = 143441505468590696209
g = 672396402136852996799074813867123583326389281120278
B = 1394256880659595564848116770226045673904445792389839
message = "Now's your chance!"
signature r = 64609209464638355801
s = 13824808741200493330
```

strang.input

— strang.input —

```
rowmatrix(r:List(Fraction(Integer))):Matrix(Fraction(Integer)) ==
 [r]::Matrix(Fraction(Integer))

columnmatrix(c:List(Fraction(Integer))):Matrix(Fraction(Integer)) ==
 [[i] for i in c]::Matrix(Fraction(Integer))

k(n) ==
 M := diagonalMatrix([2 for i in 1..n])
 for i in 1..n-1 repeat M(i,i+1):=-1
 for i in 1..n-1 repeat M(i+1,i):=-1
 M::SquareMatrix(n,Fraction(Integer))

t(n) ==
 M:=k(n)
 N:=M::Matrix(Fraction(Integer))
 qsetelt!(N,1,1,1)
 N::SquareMatrix(n,Fraction(Integer))

b(n) ==
 M:=k(n)
 N:=M::Matrix(Fraction(Integer))
 qsetelt!(N,1,1,1)
 qsetelt!(N,n,n,1)
 N::SquareMatrix(n,Fraction(Integer))

K:=k(3)
```

```
T:=t(3)
B:=b(3)
```

———-

bitmaps/axiom1.bitmap

— axiom1.bitmap —

```
#define axiom_width 270
#define axiom_height 100
static char axiom_bits[] = {
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x7e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x80, 0xff, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0,
   0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf0, 0xff, 0x1f,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0x1f, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0x1f, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0xf8, 0xff, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0xe0, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
   0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x80,
   0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x01,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xfe, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x38, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x80, 0x3f, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0xf2, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x92, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x92, 0x02,
0x00, 0xe0, 0x01, 0xf0, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x07, 0x00, 0x00, 0xe0, 0xff, 0x01, 0x00, 0x00, 0xe0,
0x00, 0xc0, 0x7f, 0x00, 0x00, 0xc0, 0x1f, 0x00, 0xd2, 0x02, 0x00, 0xe0,
0xff, 0xff, 0xff, 0x0f, 0xfe, 0xff, 0xff, 0x03, 0xf8, 0xff, 0xff, 0x03,
0xe0, 0x07, 0x00, 0x00, 0xfe, 0xff, 0x0f, 0x00, 0x00, 0xf8, 0x01, 0xf8,
0xff, 0x03, 0x00, 0xf8, 0xff, 0x01, 0x00, 0x00, 0x00, 0xe0, 0xff, 0xff,
0xff, 0x3f, 0xff, 0xff, 0xff, 0x03, 0xf8, 0xff, 0xff, 0x01, 0xf8, 0x07,
0x00, 0x80, 0xff, 0xff, 0x3f, 0x00, 0x00, 0xff, 0x01, 0xfe, 0xff, 0x07,
0x00, 0xfe, 0xff, 0x03, 0x00, 0x00, 0x00, 0xf0, 0xff, 0xff, 0xff, 0x7f,
0xff, 0xff, 0xff, 0x01, 0xf8, 0xff, 0xff, 0xfd, 0xff, 0x07, 0x00, 0xe0,
0xff, 0xff, 0x7f, 0x00, 0xff, 0xff, 0x00, 0xff, 0xff, 0x0f, 0x00, 0xff,
0xff, 0x03, 0x00, 0x00, 0x00, 0xf0, 0xff, 0x8f, 0xff, 0xff, 0xe0, 0xff,
0x7f, 0x00, 0x80, 0xff, 0x3f, 0xfe, 0xff, 0x07, 0x00, 0xf0, 0xff, 0xff,
0xff, 0x80, 0xff, 0xff, 0xc0, 0xff, 0xff, 0x1f, 0xc0, 0xff, 0xff, 0x07,
0x00, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0xf0, 0xff, 0x81, 0xff, 0x3f, 0x00,
0x00, 0xff, 0x07, 0xff, 0xff, 0x03, 0x00, 0xfe, 0xff, 0xff, 0xff, 0xc3,
0xff, 0xff, 0xf0, 0xff, 0xff, 0x3f, 0xf0, 0xff, 0xff, 0x0f, 0x00, 0x00,
0x00, 0xf0, 0x1f, 0x00, 0xe0, 0xff, 0x01, 0xff, 0x3f, 0x00, 0x00, 0xfe,
0x03, 0xe0, 0xff, 0x03, 0x00, 0xff, 0xff, 0xff, 0xff, 0x83, 0xff, 0xff,
0xf8, 0xff, 0xff, 0x3f, 0xf8, 0xff, 0xff, 0x1f, 0x00, 0x00, 0x00, 0xf0,
0x0f, 0x00, 0xc0, 0xff, 0x01, 0xfe, 0x3f, 0x00, 0x00, 0xfe, 0x01, 0xc0,
0xff, 0x03, 0x80, 0xff, 0x00, 0xfc, 0xff, 0x07, 0xf8, 0xff, 0xfc, 0x01,
0xff, 0x3f, 0xfe, 0x80, 0xff, 0x1f, 0x00, 0x00, 0x00, 0xf0, 0x07, 0x00,
0xc0, 0xff, 0x03, 0xfe, 0x3f, 0x00, 0x00, 0xff, 0x00, 0x80, 0xff, 0x03,
0xc0, 0x3f, 0x00, 0xe0, 0xff, 0x0f, 0xe0, 0xff, 0x3f, 0x00, 0xfe, 0xbf,
```

```
0x3f, 0x00, 0xff, 0x1f, 0x00, 0x00, 0x00, 0xf8, 0x01, 0x00, 0x80, 0xff,
0x03, 0xf8, 0x3f, 0x00, 0x80, 0x7f, 0x00, 0x80, 0xff, 0x03, 0xe0, 0x0f,
0x00, 0x80, 0xff, 0x1f, 0xe0, 0xff, 0x0f, 0x00, 0xf8, 0xff, 0x0f, 0x00,
0xfc, 0x1f, 0x00, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x00, 0xff, 0x03, 0xf8,
0x7f, 0x00, 0x80, 0x3f, 0x00, 0x80, 0xff, 0x03, 0xf0, 0x0f, 0x00, 0x00,
0xff, 0x1f, 0xe0, 0xff, 0x07, 0x00, 0xf8, 0xff, 0x07, 0x00, 0xfc, 0x1f,
0x00, 0x00, 0x00, 0xf8, 0x00, 0x00, 0x00, 0xff, 0x03, 0xf0, 0xff, 0x00,
0xc0, 0x1f, 0x00, 0x80, 0xff, 0x03, 0xf8, 0x07, 0x00, 0x00, 0xfe, 0x3f,
0xe0, 0xff, 0x07, 0x00, 0xf8, 0xff, 0x03, 0x00, 0xf8, 0x3f, 0x00, 0x00,
0x00, 0x78, 0x00, 0x00, 0x00, 0xff, 0x03, 0xe0, 0xff, 0x00, 0xc0, 0x0f,
0x00, 0x80, 0xff, 0x03, 0xfc, 0x03, 0x00, 0x00, 0xfc, 0x3f, 0xe0, 0xff,
0x03, 0x00, 0xf0, 0xff, 0x01, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x3c,
0x00, 0x00, 0x00, 0xff, 0x03, 0xc0, 0xff, 0x01, 0xe0, 0x0f, 0x00, 0x80,
0xff, 0x03, 0xfc, 0x03, 0x00, 0x00, 0xfc, 0x3f, 0xe0, 0xff, 0x01, 0x00,
0xf0, 0xff, 0x01, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x1c, 0x00, 0x00,
0x00, 0xff, 0x03, 0xc0, 0xff, 0x03, 0xf0, 0x07, 0x00, 0x80, 0xff, 0x03,
0xfe, 0x01, 0x00, 0x00, 0xf8, 0x7f, 0xe0, 0xff, 0x01, 0x00, 0xf0, 0xff,
0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff,
0x03, 0x80, 0xff, 0x07, 0xf8, 0x01, 0x00, 0x80, 0xff, 0x03, 0xff, 0x01,
0x00, 0x00, 0xf8, 0x7f, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00,
0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x03, 0x00,
0xff, 0x0f, 0xf8, 0x01, 0x00, 0x80, 0xff, 0x03, 0xff, 0x01, 0x00, 0x00,
0xf0, 0x7f, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x03, 0x00, 0xfe, 0x1f,
0xf8, 0x00, 0x00, 0x80, 0xff, 0x83, 0xff, 0x00, 0x00, 0x00, 0xf0, 0xff,
0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x03, 0x00, 0xfc, 0x3f, 0x7e, 0x00,
0x00, 0x80, 0xff, 0xc3, 0xff, 0x00, 0x00, 0x00, 0xe0, 0xff, 0xe0, 0xff,
0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0xff, 0x03, 0x00, 0xf8, 0x7f, 0x3e, 0x00, 0x00, 0x80,
0xff, 0xc3, 0xff, 0x00, 0x00, 0x00, 0xe0, 0xff, 0xe0, 0xff, 0x00, 0x00,
0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0xff, 0x03, 0x00, 0xf8, 0x7f, 0x3f, 0x00, 0x00, 0x80, 0xff, 0xc3,
0xff, 0x00, 0x00, 0x00, 0xe0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f,
0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff,
0x03, 0x00, 0xf0, 0xff, 0x1f, 0x00, 0x00, 0x80, 0xff, 0xc3, 0xff, 0x00,
0x00, 0x00, 0xe0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00,
0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff, 0x03, 0x00,
0xe0, 0xff, 0x0f, 0x00, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00,
0xc0, 0xff, 0xe1, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f,
0x00, 0x00, 0x00, 0x00, 0x80, 0xff, 0xff, 0xff, 0x03, 0x00, 0xc0, 0xff,
0x07, 0x00, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0xc0, 0xff,
0xe1, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00,
0x00, 0x00, 0xe0, 0xff, 0x7f, 0xff, 0x03, 0x00, 0x80, 0xff, 0x07, 0x00,
0x00, 0x80, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xe1, 0xff,
0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00,
0xfc, 0xff, 0x07, 0xff, 0x03, 0x00, 0x80, 0xff, 0x07, 0x00, 0x00, 0x80,
0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xe1, 0xff, 0x00, 0x00,
0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00, 0x00, 0x00, 0xff, 0x3f,
0x00, 0xff, 0x03, 0x00, 0x00, 0xff, 0x0f, 0x00, 0x00, 0x80, 0xff, 0xe3,
0xff, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f,
0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x00, 0xc0, 0xff, 0x07, 0x00, 0xff,
0x03, 0x00, 0x00, 0xfe, 0x1f, 0x00, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x00,
0x00, 0x00, 0xc0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00,
```

```
0xf8, 0x3f, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x01, 0x00, 0xff, 0x03, 0x00,
0x00, 0xfe, 0x1f, 0x00, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00,
0xc0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f,
0x00, 0x00, 0x00, 0xf0, 0xff, 0x00, 0x00, 0xff, 0x03, 0x00, 0x00, 0xff,
0x3f, 0x00, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0xc0, 0xff,
0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00,
0x00, 0xf8, 0x3f, 0x00, 0x00, 0xff, 0x03, 0x00, 0x80, 0xff, 0x7f, 0x00,
0x00, 0x80, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xe0, 0xff,
0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x00, 0xfc,
0x1f, 0x00, 0x00, 0xff, 0x03, 0x00, 0x80, 0xff, 0xff, 0x00, 0x00, 0x80,
0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xe0, 0xff, 0x00, 0x00,
0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x00, 0xfe, 0x1f, 0x00,
0x00, 0xff, 0x03, 0x00, 0xc0, 0xef, 0xff, 0x01, 0x00, 0x80, 0xff, 0xe3,
0xff, 0x00, 0x00, 0x00, 0xc0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f,
0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x00, 0xfe, 0x0f, 0x00, 0x00, 0xff,
0x03, 0x00, 0xe0, 0xc7, 0xff, 0x01, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x01,
0x00, 0x00, 0xc0, 0xff, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00,
0xf8, 0x1f, 0x00, 0x00, 0x00, 0xff, 0x07, 0x00, 0x00, 0xff, 0x03, 0x00,
0xf0, 0x83, 0xff, 0x07, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x01, 0x00, 0x00,
0xc0, 0x7f, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f,
0x00, 0x00, 0x80, 0xff, 0x07, 0x00, 0x00, 0xff, 0x03, 0x00, 0xf8, 0x83,
0xff, 0x0f, 0x00, 0x80, 0xff, 0xe3, 0xff, 0x01, 0x00, 0x00, 0xc0, 0x7f,
0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00,
0x80, 0xff, 0x07, 0x00, 0x80, 0xff, 0x03, 0x00, 0xf8, 0x01, 0xff, 0x0f,
0x00, 0x80, 0xff, 0xc3, 0xff, 0x01, 0x00, 0x00, 0xc0, 0x3f, 0xe0, 0xff,
0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff,
0x03, 0x00, 0x80, 0xff, 0x03, 0x00, 0xfc, 0x00, 0xfe, 0x1f, 0x00, 0x80,
0xff, 0xc3, 0xff, 0x03, 0x00, 0x00, 0xe0, 0x3f, 0xe0, 0xff, 0x00, 0x00,
0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff, 0x03, 0x00,
0x80, 0xff, 0x03, 0x00, 0xfe, 0x00, 0xfc, 0x3f, 0x00, 0x80, 0xff, 0xc3,
0xff, 0x03, 0x00, 0x00, 0xe0, 0x3f, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f,
0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff, 0x03, 0x00, 0xc0, 0xff,
0x03, 0x00, 0x7f, 0x00, 0xf8, 0x7f, 0x00, 0x80, 0xff, 0xc3, 0xff, 0x07,
0x00, 0x00, 0xe0, 0x1f, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00,
0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff, 0x03, 0x00, 0xc0, 0xff, 0x03, 0x80,
0x3f, 0x00, 0xf8, 0x7f, 0x00, 0x80, 0xff, 0x83, 0xff, 0x07, 0x00, 0x00,
0xf0, 0x0f, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f,
0x00, 0x00, 0x80, 0xff, 0x03, 0x00, 0xe0, 0xff, 0x03, 0x80, 0x1f, 0x00,
0xf0, 0xff, 0x00, 0x80, 0xff, 0x83, 0xff, 0x0f, 0x00, 0x00, 0xf0, 0x0f,
0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00,
0x80, 0xff, 0x07, 0x00, 0xe0, 0xff, 0x03, 0xc0, 0x1f, 0x00, 0xf0, 0xff,
0x01, 0x80, 0xff, 0x83, 0xff, 0x0f, 0x00, 0x00, 0xf0, 0x07, 0xe0, 0xff,
0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff,
0x07, 0x00, 0xf8, 0xff, 0x03, 0xe0, 0x0f, 0x00, 0xe0, 0xff, 0x03, 0x80,
0xff, 0x03, 0xff, 0x3f, 0x00, 0x00, 0xf8, 0x03, 0xe0, 0xff, 0x00, 0x00,
0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff, 0x0f, 0x00,
0xf8, 0xff, 0x03, 0xf0, 0x07, 0x00, 0xc0, 0xff, 0x07, 0x80, 0xff, 0x03,
0xfe, 0x7f, 0x00, 0x00, 0xfc, 0x01, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f,
0x00, 0x00, 0xf8, 0x1f, 0x00, 0x00, 0x80, 0xff, 0x1f, 0x00, 0xfe, 0xff,
0x07, 0xf8, 0x07, 0x00, 0xc0, 0xff, 0x0f, 0x80, 0xff, 0x03, 0xfe, 0xff,
0x00, 0x00, 0xfe, 0x00, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00,
0xf8, 0x3f, 0x00, 0x00, 0x80, 0xff, 0x1f, 0x00, 0x7f, 0xff, 0x0f, 0xf8,
0x07, 0x00, 0x80, 0xff, 0x1f, 0x80, 0xff, 0x03, 0xfc, 0xff, 0x01, 0x00,
0x7f, 0x00, 0xe0, 0xff, 0x00, 0x00, 0xf0, 0x7f, 0x00, 0x00, 0xf8, 0x3f,
```

```
0x00, 0x00, 0x00, 0xff, 0xff, 0xf1, 0x1f, 0xfe, 0xff, 0xff, 0x03, 0x00,
0x80, 0xff, 0x3f, 0xc0, 0xff, 0x07, 0xf8, 0xff, 0x1f, 0xf0, 0x3f, 0x00,
0xf0, 0xff, 0x00, 0x00, 0xf8, 0xff, 0x00, 0x00, 0xf8, 0x3f, 0x00, 0x00,
0x00, 0xfe, 0xff, 0xff, 0x0f, 0xfe, 0xff, 0xff, 0x03, 0x00, 0x80, 0xff,
0x7f, 0xc0, 0xff, 0x07, 0xf8, 0xff, 0xff, 0xff, 0x1f, 0x00, 0xf0, 0xff,
0x01, 0x00, 0xf8, 0xff, 0x00, 0x00, 0xfc, 0x7f, 0x00, 0x00, 0x00, 0xfc,
0xff, 0xff, 0x07, 0xfe, 0xff, 0xff, 0x07, 0x00, 0xc0, 0xff, 0xff, 0xe0,
0xff, 0x1f, 0xf0, 0xff, 0xff, 0xff, 0x0f, 0x00, 0xf8, 0xff, 0x03, 0x00,
0xf8, 0xff, 0x01, 0x00, 0xfc, 0xff, 0x00, 0x00, 0x00, 0xf8, 0xff, 0xff,
0x01, 0xfc, 0xff, 0xff, 0x0f, 0x00, 0xe0, 0xff, 0xff, 0xfb, 0xff, 0x3f,
0xe0, 0xff, 0xff, 0xff, 0x03, 0x00, 0xfe, 0xff, 0x07, 0x00, 0xfc, 0xff,
0x07, 0x00, 0xfe, 0xff, 0x01, 0x00, 0x00, 0xf0, 0xff, 0x7f, 0x00, 0xfc,
0xff, 0xff, 0x7f, 0x00, 0xf8, 0xff, 0xff, 0xff, 0xff, 0xff, 0x87, 0xff,
0xff, 0xff, 0x00, 0xc0, 0xff, 0xff, 0xff, 0x01, 0xff, 0xff, 0x7f, 0x80,
0xff, 0xff, 0x3f, 0x00, 0x00, 0xc0, 0xff, 0x3f, 0x00, 0xf8, 0xff, 0xff,
0x3f, 0x00, 0xf8, 0xff, 0xff, 0xff, 0xff, 0xff, 0x03, 0xff, 0xff, 0x3f,
0x00, 0xc0, 0xff, 0xff, 0xff, 0x80, 0xff, 0xff, 0x7f, 0xc0, 0xff, 0xff,
0x3f, 0x00, 0x00, 0x00, 0xff, 0x0f, 0x00, 0xf0, 0xff, 0x00, 0x1f, 0x00,
0xfc, 0x0f, 0xfe, 0xff, 0xcf, 0xff, 0x03, 0xfc, 0xff, 0x0f, 0x00, 0xe0,
0xff, 0xff, 0x7f, 0x80, 0xff, 0xff, 0x3f, 0xc0, 0xff, 0xff, 0x3f, 0x00,
0x00, 0x00, 0xf8, 0x03, 0x00, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x01, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x18, 0xc0, 0x01, 0x00, 0x1f, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x3c, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xff, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0xe0, 0xff, 0x07, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
```

```
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0xf0, 0xff, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8,
0xff, 0x1f, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0x1f,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xf8, 0xff, 0x1f, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0xe0, 0xff, 0x07, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0xc0, 0xff, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0xff, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x7e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x10, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00};
```

——————

License


```
--Copyright (c) 2007 Arthur C. Ralfs
--All rights reserved.
--
--Redistribution and use in source and binary forms, with or without
--modification, are permitted provided that the following conditions are
--met:
--
--    - Redistributions of source code must retain the above copyright
--      notice, this list of conditions and the following disclaimer.
--
--    - Redistributions in binary form must reproduce the above copyright
--      notice, this list of conditions and the following disclaimer in
--      the documentation and/or other materials provided with the
--      distribution.
--
--    - Neither the name of Arthur C. Ralfs nor the
--      names of its contributors may be used to endorse or promote products
--      derived from this software without specific prior written permission.
--
```

```
--THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS
--IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED
--TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A
--PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER
--OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL,
--EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO,
--PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR
--PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF
--LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING
--NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
--SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

Bibliography