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 6: AXIOM COMMAND

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.

Cyril Alberga	Roy Adler	Richard Anderson
George Andrews	Henry Baker	Stephen Balzac
Yurij Baransky	David R. Barton	Gerald Baumgartner
Gilbert Baumslag	Fred Blair	Vladimir Bondarenko
Mark Botch	Alexandre Bouyer	Peter A. Broadbery
Martin Brock	Manuel Bronstein	Florian Bundschuh
William Burge	Quentin Carpent	Bob Caviness
Bruce Char	Cheekai Chin	David V. Chudnovsky
Gregory V. Chudnovsky	Josh Cohen	Christophe Conil
Don Coppersmith	George Corliss	Robert Corless
Gary Cornell	Meino Cramer	Claire Di Crescenzo
Timothy Daly Sr.	Timothy Daly Jr.	James H. Davenport
Jean Della Dora	Gabriel Dos Reis	Michael Dewar
Claire DiCrescendo	Sam Dooley	Lionel Ducos
Martin Dunstan	Brian Dupee	Dominique Duval
Robert Edwards	Heow Eide-Goodman	Lars Erickson
Richard Fateman	Bertfried Fauser	Stuart Feldman
Brian Ford	Albrecht Fortenbacher	George Frances
Constantine Frangos	Timothy Freeman	Korrinn Fu
Marc Gaetano	Rudiger Gebauer	Kathy Gerber
Patricia Gianni	Holger Gollan	Teresa Gomez-Diaz
Laureano Gonzalez-Vega	Stephen Gortler	Johannes Grabmeier
Matt Grayson	James Griesmer	Vladimir Grinberg
Oswald Gschnitzer	Jocelyn Guidry	Steve Hague
Vilya Harvey	Satoshi Hamaguchi	Martin Hassner
Ralf Hemmecke	Henderson	Antoine Hersen
Pietro Iglio	Richard Jenks	Kai Kaminski
Grant Keady	Tony Kennedy	Paul Kosinski
Klaus Kusche	Bernhard Kutzler	Larry Lambe
Frederic Lehobey	Michel Levaud	Howard Levy
Rudiger Loos	Michael Lucks	Richard Luczak
Camm Maguire	Bob McElrath	Michael McGettrick
Ian Meikle	David Mentre	Victor S. Miller
Gerard Milmeister	Mohammed Mobarak	H. Michael Moeller
Michael Monagan	Marc Moreno-Maza	Scott Morrison
Mark Murray	William Naylor	C. Andrew Neff
John Nelder	Godfrey Nolan	Arthur Norman
Jinzhong Niu	Michael O'Connor	Kostas Oikonomou
Julian A. Padget	Bill Page	Jaap Weel
Susan Pelzel	Michel Petitot	Didier Pinchon
Claude Quitte	Norman Ramsey	Michael Richardson
Renaud Rioboo	Jean Rivlin	Nicolas Robidoux
Simon Robinson	Michael Rothstein	Martin Rubey
Philip Santas	Alfred Scheerhorn	William Schelter
Gerhard Schneider	Martin Schoenert	Marshall Schor
Fritz Schwarz	Nick Simicich	William Sit
Elena Smirnova	Jonathan Steinbach	Christine Sundaresan
Robert Sutor	Moss E. Sweedler	Eugene Surowitz
James Thatcher	Baldir Thomas	Mike Thomas
Dylan Thurston	Barry Trager	Themos T. Tsikas
Gregory Vanuxem	Bernhard Wall	Stephen Watt
Juergen Weiss	M. Weller	Mark Wegman
James Wen	Thorsten Werther	Michael Wester
John M. Wiley	Berhard Will	Clifton J. Williamson
Stephen Wilson	Shmuel Winograd	Robert Wisbauer
Sandra Wityak	Waldemar Wiwianka	Knut Wolf
Clifford Yapp	David Yun	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

The axiom system consists of a set of processes managed by the superman process. The superman process, called sman, is normally invoked from the axiom shell script in order to start a tree of subprocesses.

The `axiom` command is a shell script that collects the command line options for the `sman` process, sets some shell variables, and then invokes `sman`.

The `sman` process starts the following tree of processes:

```
--xterm---bash---sman-|-AXIOMsys
                      |-clef---spadclient
                      |-hypertex
                      |-session
                      |-sman
                      |-viewman
```

Chapter 2

The axiom Command

The `axiom` command starts everything for Axiom. The options for the `axiom` command are:

```
axiom
  [-ht    |-noht]      whether to use HyperDoc
  [-gr    |-nogr]      whether to use Graphics
  [-clef  |-noclef]    whether to use Clef
  [-nonag |-nag]       whether to use NAG
  [-noiw  |-iw]        start in interpreter in a separate window
  [-ihere |-noihere]   start an interpreter in this window
  [-nox]               don't use X Windows
  [-go  |-nogo]        whether to start system
  [-ws wsname]         use named workspace
  [-list]              list workspaces only
  [-grprog fname]      use named program for Graphics
  [-nagprog fname]     use named program for Nag
  [-htprog fname]      use named program for HyperDoc
  [-clefprog fname]    use named program for Clef
  [-sessionprog fname] use named program for session
  [-clientprog fname]  use named program for spadclient
  [-h]                 show usage
```

In detail, the command options are:

2.0.1 [-ht | -noht]

```
  [-ht    |-noht]      whether to use HyperDoc
```

`Hyperdoc`[7] is the documentation tool for Axiom. The `-ht` option, enabled by default, will start this tool. See Jenks[1] Chapter 3 for further information on the `hyperdoc` subsystem.

2.0.2 [-gr | -nogr]

```
[-gr    |-nogr]      whether to use Graphics
```

The `graphics`[8] subsystem is enabled using the `-gr` option, enabled by default. Graphics will appear as a result of a draw command, such as

```
draw(sin(x),x=0..1)
```

Note that attempting to use draw commands when the graphics is disabled will simply hang the interpreter waiting for a response. See Jenks[1] Chapter 7 for further information on the `graphics` subsystem.

2.0.3 [-clef | -noclef]

```
[-clef  |-noclef]    whether to use Clef
```

The `clef` (Command Line Edit Facility) allows for command completion. The list of command completion strings is in the last chapter of this document. If `clef`, enabled by default, is running then you can type:

```
x:Dena<tab>
```

and this will automatically be expanded to:

```
x:DenavitHartenbergMatrix
```

The `clef` program also allows command line editing. The commands are special keyboard keys.

- HOME move to beginning of the line
- END move to the end of the line
- CTRL-END delete to end of the line
- TAB command completion (multiple tabs give new choices)
- UPARROW move back thru commands
- DOWNARROW move forward thru commands
- LEFTARROW move left on the line
- RIGHTARROW move right on the line
- INSERT toggle insert/overstrike

See Jenks[1] page 21 for further information on the `clef` command.

2.0.4 [-nonag | -nag]

```
[-nonag |-nag]       whether to use NAG
```

The `nag` option, disabled by default, will attempt to start the `nagman` program in the $AXIOM/lib subdirectory. Since the open source version does not include the NAG numeric libraries this option does not work.

2.0.5 [-noiw | -iw]

```
[-noiw  |-iw]        start in interpreter in a separate window
```

The `iw` option, disabled by default, will start a second interpreter in its own window with its own frame. The fact that the second interpreter is in its own frame can be seen using the `)frame` command. For instance, if you type

```
axiom -iw
```

there will be two interpreter windows available, one in the current window and one in a new window. In the current window if you type:

```
)frame names
```

you will see:

```
   The names of the existing frames are:
            frame0
            frame1
            initial
      The current frame is the first one listed.
```

In the second window, if you type

```
)frame names
```

you will see:

```
   The names of the existing frames are:
            frame1
            frame0
            initial
      The current frame is the first one listed.
```

Setting

```
x:=3
```

in the second window will set the variable x in the frame `frame1`. Switching to the first window and typing:

```
x
```

gives:

```
   (1)  x
                                Type: Variable x
```

since the first window is in `frame0` and the variable x is defined in `frame1`. But we can switch frames in the first window using

```
)frame next
```

and then

```
x
```

gives:

```
   (2)  3
                                Type: PositiveInteger
```

and now the two windows share the same frame space. See Jenks[1] page 579 for further information on the `frame` command.

2.0.6 [-ihere | -noihere]

```
  [-ihere |-noihere]   start an interpreter in this window
```

This option determines whether Axiom will start in the current window. Using this option alone is not particularly useful and it is generally used in combination with the `-iw` option:

```
axiom -noihere -iw &
```

However, used alone, as in:

```
axiom -noihere &
```

it will start Axiom and show the Hyperdoc window. Graphics will also work from the Hyperdoc pages.

2.0.7 [-nox]

```
  [-nox]               don't use X Windows
```

allows Axiom to start the interpreter without Hyperdoc or the graphics subsystem. This is useful for starting Axiom in an emacs buffer.

2.0.8 [-go | -nogo]

```
[-go  |-nogo]        whether to start system
```

uses the `-go` option, enabled by default, controls whether the system starts from the command line. If the `-nogo` option is chosen the system prints the command line that would have been issued. This is useful for finding out what the command line options to `sman` will be. For instance:

```
axiom -nogo -iw
```

does not start Axiom but types out:

```
Would now start the processes.
exec ~/mnt/linux/bin/sman -iw -ws ~/mnt/linux/bin/AXIOMsys
```

2.0.9 [-ws wsname]

```
[-ws wsname]         use named workspace
```

In the `-nogo` command above you can see that the default workspace name is

```
-ws ~/mnt/linux/bin/AXIOMsys
```

This option allows you to change that. This is useful for debugging new system builds. During build a debugging version of Axiom is created in the `obj/linux/bin` directory. The `debugsys` image uses interpreted lisp code rather than compiled code. This makes it possible to do deep debugging. To use this workspace you would incant:

```
cd youraxiombuild
export AXIOM=‘pwd‘/mnt/linux
export PATH=$AXIOM/bin:$PATH
axiom -ws obj/linux/bin/debugsys
```

2.0.10 [-list]

```
[-list]              list workspaces only
```

shows you the executable workspaces. Generally in a built system there is only one, called `$AXIOM/bin/AXIOMsys`.

2.0.11 [-grprog fname]

```
[-grprog fname]      use named program for Graphics
```

allows you to specify which program to use for the graphics. By default this is `$AXIOM/lib/viewman`.

2.0.12 [-nagprog fname]

```
[-nagprog fname]     use named program for Nag
```

allows you to specify which program to use for the NAG library connection. By default this is
`$AXIOM/lib/nagman` but it is disabled by default.

2.0.13 [-htprog fname]

```
[-htprog fname]      use named program for Hyperdoc
```

allows you tp specify which program to use for Hyperdoc. By default it is
`$AXIOM/bin/hypertex -s`.

2.0.14 [-clefprog fname]

```
[-clefprog fname]    use named program for Clef
```

allows you to specify which program to use for clef. By default it is
`$AXIOM/bin/clef -f $AXIOM/lib/command.list -e`.

2.0.15 [-sessionprog fname]

```
[-sessionprog fname] use named program for session
```

allows you to specify the session manager program. By default it is
`$AXIOM/lib/session`.

2.0.16 [-clientprog fname]

```
[-clientprog fname]  use named program for spadclient
```

allows you to specify the spadclient program. By default it is
`$AXIOM/lib/spadclient`.

2.0.17 [-h]

```
[-h]                 show usage
```

— **axiomcmd** —

```
#!/bin/sh
```

———-

The `MALLOCTYPE` shell variable is an `IBM AIX` shell variable that controls buckets based extensions in the default memory allocator which may enhance performance. AIX uses a new memory management routine that does not zero `malloc` memory and does not round up to the nearest power of 2, unlike most non-AIX systems. This can cause failures so we protect against that here. See the AIX Performance Tuning Guide[9] for details.

— **axiomcmd** —

```
MALLOCTYPE=3.1
export MALLOCTYPE
```

———-

The `nagman` process needs to know the hostname

— **axiomcmd** —

```
HOST=‘hostname‘
export HOST
```

———-

There are 4 basic utilities used by this script. The `ciao` script for immediate exit:

— **axiomcmd** —

```
ciao() {
echo "Goodbye."
exit 1
}
```

———-

The `needsubopt` script which is used to issue an error message when one of the command line options requires an option:

— **axiomcmd** —

```
needsubopt () {
echo "The $1 option requires an argument."
ciao
}
```

———-

The `showuse` script which gives basic command line help:

— **axiomcmd** —

```
showuse() {
echo "axiom"
echo "  [-ht    |-noht]      whether to use HyperDoc"
echo "  [-gr    |-nogr]      whether to use Graphics"
echo "  [-clef  |-noclef]    whether to use Clef"
echo "  [-nonag |-nag]       whether to use NAG"
echo "  [-noiw  |-iw]        start in interpreter in a separate window"
echo "  [-ihere |-noihere]   start an interpreter in this window"
echo "  [-nox]               don't use X Windows"
echo "  [-go  |-nogo]        whether to start system"
echo "  [-ws wsname]         use named workspace"
echo "  [-list]              list workspaces only"
echo "  [-grprog fname]      use named program for Graphics"
echo "  [-nagprog fname]     use named program for Nag"
echo "  [-htprog fname]      use named program for HyperDoc"
echo "  [-clefprog fname]    use named program for Clef"
echo "  [-sessionprog fname] use named program for session"
echo "  [-clientprog fname]  use named program for spadclient"
echo "  [-h]                 show usage"
}
```

————-

List the various workspaces if asked.

— **axiomcmd** —

```
listwspaces()
{
        echo "$1"
        ls -l $2 | grep "sys$"
        echo ""
}
```

————-

Step 1. Ensure the environment is set.

Just process "-h". If it exists in the command line then we print out the simple command line help menu.

— **axiomcmd** —

```
if [ "$*" = "-h" ] ; then
     showuse
fi
```

————-

We assume that Axiom is installed in the standard place on a linux system. We will modify this assumption as we process the environment and command line. The term `spad` is an

historical shortened version of the name `scratchpad`, the original name of the `Axiom` system.

— **axiomcmd** —

```
SPADDEFAULT=/usr/local/axiom/mnt/linux
```

————-

If the `$AXIOM` shell variable is set then we use it.
If not, then if the `$SPAD` shell variable is set then we use it.
If not, then we try to use the default value above.
If not, we simply fail.

— **axiomcmd** —

```
if [ "$SPAD" = "" ] ; then
  if [ "$AXIOM" = "" ] ; then
    SPAD=$SPADDEFAULT
    echo "AXIOM variable is not set"
    echo "assuming AXIOM = $SPAD"
    AXIOM=$SPAD
    export AXIOM
  else
    SPAD=$AXIOM
  fi
  export SPAD
else
  if [ "$AXIOM" = "" ] ; then
    echo "AXIOM variable is not set"
    echo "but SPAD = $SPAD"
    echo "Using AXIOM = $SPAD"
    AXIOM=$SPAD
    export AXIOM
  else
    if [ ! "$SPAD" = "$AXIOM" ] ; then
       echo "ignoring SPAD variable"
       SPAD=$AXIOM
    fi
  fi
fi
```

————-

If we get here then all attempts to find axiom have failed so we complain and exit.

— **axiomcmd** —

```
if [ ! -d "$SPAD" ] ; then
  echo "The directory for Axiom, $SPAD, does not exist."
  ciao
fi
```

———-

Step 2. Process command line arguments.

Name the workspace directories

— **axiomcmd** —

```
rootwsdir=$SPAD/bin
```

———-

We set up the defaults for command-line arguments. We don't want just a list by default

— **axiomcmd** —

```
list=no
```

———-

We default to actually executing the workspace.

— **axiomcmd** —

```
go=yes
```

———-

We default to the `AXIOMsys` workspace.

— **axiomcmd** —

```
wsname=AXIOMsys
```

———-

And all other options are unset.

— **axiomcmd** —

```
otheropts=""
```

———-

For each option on the command line do

— **axiomcmd** —

```
while [ "$*" != "" ] ; do
```

———-

— **axiomcmd** —

```
case $1 in
```

———-

If the user specified list anywhere then we give the workspace list and exit.

— **axiomcmd** —

```
        -list)  list=yes
                go=no;;
```

———-

If the user specified `go` or `nogo` we handle that case

— **axiomcmd** —

```
-go) go=yes ;;
-nogo) go=no ;;
```

———-

The workspace option requires an argument which follows immediately. If the argument is missing we complain and exit.

— **axiomcmd** —

```
-ws)
if [ "$2" = "" ] ; then needsubopt "$1" ; fi
shift
wsname="$1"
;;
```

———-

We can specify the various subprograms to use.

— **axiomcmd** —

```
-nagprog|-grprog|-htprog|-clefprog|-sessionprog|-clientprog)
if [ "$2" = "" ] ; then needsubopt "$1" ; fi
otheropts="$otheropts  $1 $2"
shift
;;
```

———-

These options were not explained earlier and are only for developer use.

— **axiomcmd** —

```
        -paste|-rm|-rv)
if [ "$2" = "" ] ; then needsubopt "$1" ; fi
otheropts="$otheropts  $1 $2"
shift
;;
```

———-

We handle the various [-option | -nooption] cases

— **axiomcmd** —

```
-clef|-noclef|-gr|-nogr|-ht|-noht|-iw|-noiw)
otheropts="$otheropts $1"
;;
        -ihere|-noihere|-nox|-nag|-nonag)
otheropts="$otheropts $1"
;;
```

———-

The user wanted help so we will not execute.

— **axiomcmd** —

```
-h)
go=no
;;
```

———-

The user is confused. Complain and exit.

— **axiomcmd** —

```
*) echo "Unknown option: $1"
echo "To use a specific workspace use, e.g.: spad -ws $1"
ciao
;;
esac
```

———-

Move to the next option and loop.

— **axiomcmd** —

```
shift
done
```

———-

Step 3. Handle options that require special case handling.

The user just wanted to know what workspaces are available.

— **axiomcmd** —

```
if [ $list = yes ] ; then
 listwspaces "AXIOM workspaces in \$AXIOM/bin = $rootwsdir: " $rootwsdir
fi
```

———-

Try to ensure a suitable workspace on this host.

— **axiomcmd** —

```
if [ `expr $wsname : '.*/.*'` = 0 ] ; then
serverws=$rootwsdir/$wsname
else
serverws=$wsname
fi
```

———-

If we can't find the executable then we complain and exit.

— **axiomcmd** —

```
if [ ! -x $serverws ] ; then
        echo "Cannot find the executable $serverws"
showuse
ciao
fi
```

———-

The user just wanted to see what would happen so we output the command line and exit.

— **axiomcmd** —

```
if [ $go = no ] ; then
echo "Would now start the processes."
echo exec $SPAD/bin/sman $otheropts -ws $serverws
exit 0
fi
```

———-

All of the options have been processed so we start `sman`

— **axiomcmd** —

```
exec $SPAD/bin/sman $otheropts -ws $serverws
```

———-

Chapter 3

The sman program

3.1 sman.h

The spad_proc structure holds information about the process id of a child process, what to do when it dies, and the shell command line necessary to restart the process. There is a linked list of these structures which maintains the process list for axiom.

— **sman.h** —

```
/* Process control definitions.  Used by fork_you and spawn_of_hell */

/* When a process dies it kills off everything else */
#define Die 1
/* When a process dies, do nothing */
#define NadaDelShitsky  2
/* When a process dies start it up again */
#define DoItAgain       3
/* When hypertex dies, clean its socket */
#define CleanHypertexSocket 4

typedef struct spad_proc {
  int proc_id; /* process id of child */
  int death_action; /* one of the above constants */
  char *command; /* sh command line to restart the process */
  struct spad_proc *next;
} SpadProcess;
```

———-

3.2 sman

3.2.1 includes

— sman.includes —

```
#include <stdlib.h>
#include <unistd.h>
#include <string.h>
#include <stdio.h>
#include <pwd.h>
#include <fcntl.h>
#include <termios.h>
#include <errno.h>
#include <sys/time.h>
#include <sys/wait.h>
#include <signal.h>

#if defined(SUN4OS5platform) || defined(HP10platform)
#include <sys/stropts.h>
#endif

#include "com.h"
#include "bsdsignal.h"
#include "sman.h"

#include "bsdsignal.h1"
#include "sockio-c.h1"
#include "openpty.h1"
#include "sman.h1"
```

————-

3.2.2 variables

— sman.variables —

```
char *ws_path;                  /* location of the AXIOM executable */
int start_clef; /* start clef under spad */
int start_graphics; /* start the viewman */
int start_nagman;               /* start the nagman */
int start_ht; /* start hypertex */
int start_spadclient; /* Start the client spad buffer */
int start_local_spadclient; /* Start the client spad buffer */
int use_X; /* Use the X windows environment */
```

```
int server_num; /* AXIOM server number */
```

———-

We add a debug flag so we can print information about what sman is trying to do. This change is pervasive as it touches nearly every routine.

— **sman.variables** —

```
int tpd=0;                      /* to-print-debug information */

/*************************************************/
/* definitions of programs which sman can start */
/*************************************************/

\getchunk{the viewman command line}
\getchunk{the nagman command line}
\getchunk{the hypertex command line}
\getchunk{the clef command line}
\getchunk{the session manager command line}
\getchunk{the spadclient command line}
char *PasteFile = NULL;
char *MakeRecordFile = NULL;
char *VerifyRecordFile = NULL;

SpadProcess *spad_process_list = NULL;
/***************************/
/* sman defaults file name */
/***************************/

#define SpadDefaultFile "spadprof.input"

char ClefCommandLine[256];

#define BufSize      4096  /* size of communication buffer */
char big_bad_buf[BufSize];      /* big I/O buffer */

Sock *session_io = NULL;        /* socket connecting to session manager */

/*********************************************************/
/* Some characters used and externally defined in edible.h */
/*********************************************************/

unsigned char  _INTR, _QUIT, _ERASE, _KILL, _EOF, _EOL, _RES1, _RES2;

/************************************/
/* Stuff for opening pseudo-terminal */
/************************************/

int ptsNum, ptcNum;
char ptsPath[20], ptcPath[20];
```

```
char **new_envp;                /* new environment for AXIOM */
int child_pid;                  /* child's process id */
struct termios oldbuf;           /* the original settings */
struct termios childbuf;         /* terminal structure for user i/o */

int nagman_signal=0;
int death_signal = 0;
```

———-

3.2.3 process_arguments

— **sman.processarguments** —

```
static void
process_arguments(int argc,char ** argv)
{
  int arg;
  if (tpd == 1) fprintf(stderr,"sman:process_arguments entered\n");
  for (arg = 1; arg < argc; arg++) {
    if      (strcmp(argv[arg], "-debug")      == 0)
      tpd = 1;
    else if (strcmp(argv[arg], "-noclef")      == 0)
      start_clef = 0;
    else if (strcmp(argv[arg], "-clef")        == 0)
      start_clef = 1;
    else if (strcmp(argv[arg], "-gr")          == 0)
      start_graphics = 1;
    else if (strcmp(argv[arg], "-nogr")        == 0)
      start_graphics = 0;
    else if (strcmp(argv[arg], "-nag")          == 0)
      start_nagman = 1;
    else if (strcmp(argv[arg], "-nonag")        == 0)
      start_nagman = 0;
    else if (strcmp(argv[arg], "-ht")          == 0)
      start_ht = 1;
    else if (strcmp(argv[arg], "-noht")        == 0)
      start_ht = 0;
    else if (strcmp(argv[arg], "-iw")          == 0)
      start_spadclient = 1;
    else if (strcmp(argv[arg], "-ihere")       == 0)
      start_local_spadclient = 1;
    else if (strcmp(argv[arg], "-noihere")     == 0)
      start_local_spadclient = 0;
    else if (strcmp(argv[arg], "-noiw")        == 0)
      start_spadclient = 0;
```

```
    else if (strcmp(argv[arg], "-ws")          == 0)
      ws_path = argv[++arg];
    else if (strcmp(argv[arg], "-comp")        == 0)
      ws_path = "$AXIOM/etc/images/comp";
    else if (strcmp(argv[arg], "-nox")         == 0)
      {
use_X = 0;
start_local_spadclient = 1;
start_spadclient = 0;
start_ht = 0;
start_graphics = 0;
      }
    else if (strcmp(argv[arg], "-grprog")      == 0)
      GraphicsProgram = argv[++arg];
    else if (strcmp(argv[arg], "-nagprog")      == 0)
      NagManagerProgram = argv[++arg];
    else if (strcmp(argv[arg], "-htprog")      == 0)
      HypertexProgram = argv[++arg];
    else if (strcmp(argv[arg], "-clefprog")    == 0) {
      strcpy(ClefCommandLine,argv[++arg]);
    ClefProgram =
        strcat(ClefCommandLine, " -f $AXIOM/lib/command.list -e ");
    }
    else if (strcmp(argv[arg], "-sessionprog") == 0)
      SessionManagerProgram = argv[++arg];
    else if (strcmp(argv[arg], "-clientprog")  == 0)
      SpadClientProgram = argv[++arg];
    else if (strcmp(argv[arg], "-rm")  == 0)
      MakeRecordFile = argv[++arg];
    else if (strcmp(argv[arg], "-rv")  == 0)
      VerifyRecordFile = argv[++arg];
    else if (strcmp(argv[arg], "-paste")  == 0)
      PasteFile = argv[++arg];
    else {
      fprintf(stderr, "Usage: sman <-clef|-noclef> <-gr|-nogr> <-ht|-noht>");
      fprintf(stderr, " <-iw|-noiw> <-nag|-nonag> <-nox> <-comp>");
      fprintf(stderr, " <-ws spad_workspace> <-grprog path> <-htprog path>");
      fprintf(stderr, " <-clefprog path> <-sessionprog path> <-nagprog path>");
      fprintf(stderr, " <-clientprog path>\n");
      exit(-1);
    }
  }
  if (tpd == 1)
  { fprintf(stderr,"  sman ");
    if (start_clef == 0)
      fprintf(stderr,"-noclef ");
    else
      fprintf(stderr,"-clef ");
    if (start_graphics == 0)
      fprintf(stderr,"-nogr ");
```

```
    else
      fprintf(stderr,"-gr ");
    if (start_nagman == 0)
      fprintf(stderr,"-nonag ");
    else
      fprintf(stderr,"-nag ");
    if (start_ht == 0)
      fprintf(stderr,"-noht ");
    else
      fprintf(stderr,"-ht ");
    if (start_spadclient == 0)
      fprintf(stderr,"-noiw ");
    else
      fprintf(stderr,"-iw ");
    if (start_local_spadclient == 0)
      fprintf(stderr,"-noihere ");
    else
      fprintf(stderr,"-ihere ");
    if (start_local_spadclient == 0)
      fprintf(stderr,"-noihere ");
    else
      fprintf(stderr,"-ihere ");
    if (use_X == 0)
      fprintf(stderr,"-nox ");
    fprintf(stderr,"-ws ");
    fprintf(stderr,"'%s' ",ws_path);
    fprintf(stderr,"-grprog ");
    fprintf(stderr,"'%s' ",GraphicsProgram);
    fprintf(stderr,"-nagprog ");
    fprintf(stderr,"'%s' ",NagManagerProgram);
    fprintf(stderr,"-htprog ");
    fprintf(stderr,"'%s' ",HypertexProgram);
    fprintf(stderr,"-clefprog ");
    fprintf(stderr,"'%s' ",ClefCommandLine);
    fprintf(stderr,"-sessionprog ");
    fprintf(stderr,"'%s' ",SessionManagerProgram);
    fprintf(stderr,"-clientprog ");
    fprintf(stderr,"'%s' ",SpadClientProgram);
    fprintf(stderr,"-rm ");
    fprintf(stderr,"'%s' ",MakeRecordFile);
    fprintf(stderr,"-rv ");
    fprintf(stderr,"'%s' ",VerifyRecordFile);
    fprintf(stderr,"-paste ");
    fprintf(stderr,"'%s' ",PasteFile);
    fprintf(stderr,"\n");
  }
  if (tpd == 1) fprintf(stderr,"sman:process_arguments exit\n");
}
```

———-

3.2.4 should_I_clef

— **sman.shouldIclef** —

```
static int
should_I_clef(void)
{
  return(1);
}
```

———-

3.2.5 in_X

— **sman.inX** —

```
static int
in_X(void)
{
  if (getenv("DISPLAY")) return 1;
  return 0;
}
```

———-

3.2.6 set_up_defaults

These are the default values for sman. A '1' value means that sman will try to start the given process, a '0' value means not starting the process.

We do not have replacement code for the nagman process nor do we have a copy of the nag fortran library to test the process. Until this changes we set start_nagman = 0 in order to disable starting this process by default.

— **sman.setupdefaults** —

```
static  void
set_up_defaults(void)
{
  if (tpd == 1) fprintf(stderr,"sman:set_up_defaults entered\n");
  start_clef = should_I_clef();
```

```
  start_graphics = 1;
  start_nagman = 0;
  start_ht = 1;
  start_spadclient = 0;
  start_local_spadclient = 1;
  use_X = isatty(0) && in_X();
  ws_path = "$AXIOM/bin/AXIOMsys";
  if (tpd == 1) fprintf(stderr,"sman:set_up_defaults exit\n");
}
```

———-

3.2.7 process_options

— **sman.processoptions** —

```
static void
process_options(int argc, char **argv)
{
  if (tpd == 1) fprintf(stderr,"sman:process_options entered\n");
  set_up_defaults();
  process_arguments(argc, argv);
  if (tpd == 1) fprintf(stderr,"sman:process_options exit\n");
}
```

———-

3.2.8 death_handler

— **sman.deathhandler** —

```
static void
death_handler(int sig)
{
  death_signal = 1;
}
```

———-

3.2.9 nagman_handler

— **sman.nagmanhandler** —

```
static void
nagman_handler(int sig)
{
  nagman_signal=1;
}
```

———-

3.2.10 sman_catch_signals

— **sman.smancatchsignals** —

```
static void
sman_catch_signals(void)
{

  /* Set up the signal handlers for sman */
  bsdSignal(SIGINT,  SIG_IGN,RestartSystemCalls);
  bsdSignal(SIGTERM, death_handler,RestartSystemCalls);
  bsdSignal(SIGQUIT, death_handler,RestartSystemCalls);
  bsdSignal(SIGHUP,  death_handler,RestartSystemCalls);
  bsdSignal(SIGILL,  death_handler,RestartSystemCalls);
  bsdSignal(SIGTRAP, death_handler,RestartSystemCalls);
  bsdSignal(SIGIOT,  death_handler,RestartSystemCalls);
  bsdSignal(SIGBUS,  death_handler,RestartSystemCalls);
  bsdSignal(SIGSEGV, death_handler,RestartSystemCalls);
  /* don't restart wait call on SIGUSR1  */
  bsdSignal(SIGUSR1, nagman_handler,DontRestartSystemCalls);
  /* ONLY nagman should send this.
     If an error (such as C-c) interrupts a NAGLINK call, nagman
     gets a signal to clean up. We need to start another nagman
     almost immediately to process the next NAGLINK request.
     Since nagman takes a while to clean up, we treat it specially.
     nagman should send a signal (USR1) to sman.
     sman should respond by spawning a new nagman.

     so nagman is NOT a DoItAgain but a NadaDelShitsky.

     The USR1 mechanism does not work for HPUX 9 - use DoItAgain
     */

}
```

———-

3.2.11 fix_env

insert SPADSERVER and SPADNUM variables into the environemnt

— **sman.fixenv** —

```
static void
fix_env(char **envp, int spadnum)
{
  int len, i;
  char *sn;
  for(len = 0; envp[len] != NULL; len++);
  new_envp = (char **) malloc((len + 3) * sizeof(char *));
  new_envp[0] = "SPADSERVER=TRUE";
  sn = (char *) malloc(20 * sizeof(char));
  sprintf(sn, "SPADNUM=%d", spadnum);
  new_envp[1] = sn;
  for(i=0; i<=len; i++)
    new_envp[i+2] = envp[i];
}
```

———-

3.2.12 init_term_io

— **sman.inittermio** —

```
static void
init_term_io(void)
{
  if(!isatty(0)) return;
  if( tcgetattr(0, &oldbuf) == -1) {
    perror("getting termios");
    return ;  /*  exit(-1); */
  }
  if( tcgetattr(0, &childbuf) == -1) {
    perror("getting termios");
    return ;  /*   exit(-1); */
  }
  _INTR = oldbuf.c_cc[VINTR];
  _QUIT = oldbuf.c_cc[VQUIT];
  _ERASE = oldbuf.c_cc[VERASE];
  _KILL = oldbuf.c_cc[VKILL];
  _EOF = oldbuf.c_cc[VEOF];
  _EOL = oldbuf.c_cc[VEOL];
}
```

———-

3.2.13 strPrefix

— **sman.strPrefix** —

```
static char *
strPrefix(char *prefix,char * s)
{
  while (*prefix != '\0' && *prefix == *s) {
    prefix++;
    s++;
  }
  if (*prefix == '\0') return s;
  return NULL;
}
```

———-

3.2.14 check_spad_proc

— **sman.checkspadproc** —

```
static void
check_spad_proc(char *file, char *prefix)
{
  char *num;
  int pid;
  if ((num = strPrefix(prefix, file))) {
    pid = atoi(num);
    if (pid > 2) {
      kill(pid, 0);
      if (kill(pid, 0) == -1 && errno == ESRCH) {
unlink(file);
      }
    }
  }
}
```

———-

3.2.15 clean_up_old_sockets

— sman.cleanupoldsockets —

```
static void
clean_up_old_sockets(void)
{
  char com[512], tmp_file[128];
  FILE *file;
  int len;
  sprintf(tmp_file, "/tmp/socks.%d", server_num);
  sprintf(com, "ls /tmp/.d* /tmp/.s* /tmp/.i* /tmp/.h* 2> %s > %s",
  tmp_file, tmp_file);
  system(com);
  file = fopen(tmp_file, "r");
  if (file == NULL) {
    fprintf(stderr, "Can't open socket listing file\n");
    return;
  }
  while(fgets(com, 512, file) != NULL) {
    len = strlen(com);
    if (len) com[len-1] = '\0';
    else break;
    check_spad_proc(com, "/tmp/.d");
    check_spad_proc(com, "/tmp/.s");
    check_spad_proc(com, "/tmp/.i");
    check_spad_proc(com, "/tmp/.h");
  }
  fclose(file);
  unlink(tmp_file);
}
```

————-

3.2.16 fork_you

— sman.forkyou —

```
static SpadProcess *
fork_you(int death_action)
{
  /* fork a new process, giving it a default death action */
  /* return NULL in child, SpadProcess in parent          */
  int child_pid = fork();
  SpadProcess *proc;
```

```
  if (!child_pid) return NULL;
  proc = (SpadProcess *) malloc(sizeof(SpadProcess));
  proc->proc_id = child_pid;
  proc->death_action = death_action;
  proc->command = NULL;
  proc->next = spad_process_list;
  spad_process_list = proc;
  return proc;
}
```

———-

3.2.17 exec_command_env

Note that the next-to-last argument of `execle` must be an explicit NULL pointer. The previous naked 0 value was not correct.

— **sman.execcommandenv** —

```
static void
exec_command_env(char *command,char ** env)
{
  char new_command[512];
  sprintf(new_command, "exec %s", command);
  execle("/bin/sh","/bin/sh", "-c", new_command, (char *)0, env);
}
```

———-

3.2.18 spawn_of_hell

— **sman.spawnofhell** —

```
static SpadProcess *
spawn_of_hell(char *command, int death_action)
{
  SpadProcess *proc = fork_you(death_action);
  if (proc != NULL) {
    proc->command = command;
    return proc;
  }
  exec_command_env(command, new_envp);
  return NULL;
}
```

———-

3.2.19 start_the_spadclient

run a AXIOM client in the main process

— **sman.startthespadclient** —

```
static void
start_the_spadclient(void)
{
  char command[256];
  if (start_clef)
#ifdef RIOSplatform
    sprintf(command,
    "aixterm -sb -sl 500 -name axiomclient -n AXIOM -T AXIOM -e %s %s",
    ClefProgram, SpadClientProgram);
#else
  sprintf(command,
  "xterm -sb -sl 500 -name axiomclient -n AXIOM -T AXIOM -e %s %s",
  ClefProgram, SpadClientProgram);
#endif
  else
#ifdef RIOSplatform
    sprintf(command,
    "aixterm -sb -sl 500 -name axiomclient -n AXIOM -T AXIOM -e %s",
    SpadClientProgram);
#else
  sprintf(command,
  "xterm -sb -sl 500 -name axiomclient -n AXIOM -T AXIOM -e %s",
  SpadClientProgram);
#endif
  if (tpd == 1)
    fprintf(stderr,"sman:start_the_spadclient: %s\n",command);
  spawn_of_hell(command, NadaDelShitsky);
}
```

———-

3.2.20 start_the_local_spadclient

— **sman.startthelocalspadclient** —

```
static void
start_the_local_spadclient(void)
{
```

```
  char command[256];
  if (start_clef)
    sprintf(command, "%s  %s", ClefProgram, SpadClientProgram);
  else
    sprintf(command, "%s", SpadClientProgram);
  if (tpd == 1)
    fprintf(stderr,"sman:start_the_local_spadclient: %s\n",command);
  spawn_of_hell(command, NadaDelShitsky);
}
```

———-

3.2.21 start_the_nagman

— **sman.startthenagman** —

```
static void
start_the_nagman(void)
{
#if defined(HP9platform)
  spawn_of_hell(NagManagerProgram,DoItAgain);
#else
  spawn_of_hell(NagManagerProgram,NadaDelShitsky );
#endif
}
```

———-

3.2.22 start_the_session_manager

— **sman.startthesessionmanager** —

```
static void
start_the_session_manager(void)
{
  spawn_of_hell(SessionManagerProgram, Die);
}
```

———-

3.2.23 start_the_hypertex

— sman.startthehypertex —

```
static void
start_the_hypertex(void)
{
  char prog[512];

  if (PasteFile){
    sprintf(prog, "%s -k -ip %s", HypertexProgram, PasteFile);
    spawn_of_hell(prog, NadaDelShitsky);
  }
  else if (MakeRecordFile){
    sprintf(prog, "%s -k -rm %s", HypertexProgram,MakeRecordFile );
    spawn_of_hell(prog, NadaDelShitsky);
  }
  else if (VerifyRecordFile){
    sprintf(prog, "%s -k -rv %s", HypertexProgram, VerifyRecordFile);
    spawn_of_hell(prog, NadaDelShitsky);
  }
  /* If we restart hyperdoc from the axiom command prompt */
  else spawn_of_hell(HypertexProgram, CleanHypertexSocket);
}
```

————-

3.2.24 start_the_graphics

— sman.startthegraphics —

```
static void
start_the_graphics(void)
{
  spawn_of_hell(GraphicsProgram, DoItAgain);
}
```

————-

3.2.25 fork_Axiom

— sman.forkAxiom —

```
/* Start the AXIOM session in a separate process, */
/* using a pseudo-terminal to catch all input and output */
static void
fork_Axiom(void)
{
  char augmented_ws_path[256];  /* will append directory path */
  char *tmp_pointer;
  SpadProcess *proc;

  proc =  fork_you(Die);
  child_pid = (proc == NULL ? 0 : proc->proc_id);
  switch(child_pid) {
  case -1 :
    fprintf(stderr, "Can't create a new process \n");
    exit(0);
  case 0:
    /* Dissasociate from my parents group so all my child processes */
    /* look at my terminal as the controlling terminal for the      */
    /* group                                                        */

    if(setsid() < 0) {
      perror("Dissassociating from parents group");
      exit(-1);
    }

    close(ptsNum);
    /* Now reopen the server side, so that pg, su, etc. work properly */

    if ((ptsNum =  open(ptsPath, O_RDWR)) < 0 ) {
      perror("fork_Axiom: Failed to reopen server");
      exit(-1);
    }
#if defined(SUN4OS5platform) || defined(HP10platform)
    ioctl(ptsNum,I_PUSH,"ptem");
    ioctl(ptsNum,I_PUSH,"ldterm");
#endif

    /* since I am the child, I can close ptc, and dup pts for all its */
    /* standard descriptors                                           */

    if( (dup2(ptsNum, 0) == -1) ||
        (dup2(ptsNum, 1) == -1) ||
        (dup2(ptsNum, 2) == -1)  ) {
      perror("trying to dupe the child");
      exit(-1);
    }
    close(ptcNum);
    close(ptsNum);
```

```
    /* I also have to turn off echoing, since I am echoing all the */
    /* input myself                  */

    childbuf.c_lflag &= ~ECHO;
    if( tcsetattr(0, TCSAFLUSH, &childbuf) == -1) {
      perror("setting the term buffer");
      exit(-1);
    }
    strcpy(augmented_ws_path,ws_path);          /* write the name    */
    strcat(augmented_ws_path," ");              /* space             */
    strcat(augmented_ws_path,ws_path);          /* name again        */
    tmp_pointer = (char *)
      strrchr(augmented_ws_path,'/');      /*pointer to last /  */
    *(++tmp_pointer) = '\0';
    exec_command_env(augmented_ws_path, new_envp);

    /*    fprintf(stderr, "Cannot execute the %s system.\n", ws_path); */

    exit(0);
  }
}
```

———-

3.2.26 start_the_Axiom

— **sman.starttheAxiom** —

```
static void
start_the_Axiom(char **envp)
{
  server_num = make_server_number();
  clean_up_old_sockets();
  if (server_num == -1) {
    fprintf(stderr, "could not get an AXIOM server number\n");
    exit(-1);
  }
  if (ptyopen(&ptcNum, &ptsNum, ptcPath, ptsPath) == -1) {
    perror("start_the_Axiom: ptyopen failed");
    exit(-1);
  }
  fix_env(envp, server_num);
  fork_Axiom();
  close(ptsNum);
}
```

———-

3.2.27 clean_up_sockets

In order to be able to restart hyperdoc from the axiom command prompt we need to remove the socket for this server.

— **sman.cleanupsockets** —

```
static void
clean_hypertex_socket(void)
{
   char name[256];
   sprintf(name, "%s%d", MenuServerName, server_num);
   unlink(name);
}

static void
clean_up_sockets(void)
{
  char name[256];
  sprintf(name, "%s%d", SpadServer, server_num);
  unlink(name);
  sprintf(name, "%s%d", SessionServer, server_num);
  unlink(name);
  sprintf(name, "%s%d", SessionIOName, server_num);
  unlink(name);
  clean_hypertex_socket();
}
```

———-

3.2.28 read_from_spad_io

— **sman.readfromspadio** —

```
static void
read_from_spad_io(int ptcNum)
{
  int ret_code = 0, i=0;
  static int mes_len =0;
  ret_code = read(ptcNum, big_bad_buf, BufSize);
  if (ret_code == -1) {
    clean_up_sockets();
    exit(-1);
  }
```

```
  if (session_io == NULL) {
    if (ret_code < mes_len)
      mes_len -= ret_code;
    else {
      if (mes_len > 0) {
i = mes_len;
mes_len = 0;
      }
      else
i = 0;
      ret_code = write(1, big_bad_buf+i, ret_code-i);
    }
  }
  else
    ret_code = swrite(session_io, big_bad_buf, ret_code,
      "writing to session man");
  if (ret_code == -1) {
    perror("writing output to session manager");
    clean_up_sockets();
    exit(-1);
  }
}
```

———-

3.2.29 read_from_manager

— **sman.readfrommanager** —

```
static void
read_from_manager(int ptcNum)
{
  int ret_code;
  ret_code = sread(session_io, big_bad_buf, BufSize, "reading session io");
  if (ret_code == -1) {
    return;
  }
  ret_code = write(ptcNum, big_bad_buf, ret_code);
  if (ret_code == -1) {
    return;
  }
}
```

———-

3.2.30 manage_spad_io

— **sman.managespadio** —

```
static void
manage_spad_io(int ptcNum)
{
  int ret_code, i, p;
  fd_set rd;
  while (1) {
    rd = socket_mask;
    FD_SET(ptcNum, &rd);
    if (session_io != NULL)
      FD_SET(session_io->socket, &rd);
    ret_code = sselect(FD_SETSIZE, &rd, 0, 0, NULL);
    if (ret_code == -1) {
      perror("Session manager select");
      clean_up_sockets();
      exit(-1);
    }
    if (FD_ISSET(ptcNum, &rd)) {
      read_from_spad_io(ptcNum);
    }
    for(i=0; i<2; i++) {
      if (server[i].socket > 0 && FD_ISSET(server[i].socket, &rd)) {
p = accept_connection(server+i);
switch(p) {
case SessionIO:
  session_io = purpose_table[SessionIO];
  /*  printf("connected session manager\n\r");*/
  printf("\n");
  break;
default:
  printf("sman: Unkown connection request type: %d\n", p);
  break;
}
      }
    }
    if (session_io != NULL && FD_ISSET(session_io->socket, &rd)) {
      read_from_manager(ptcNum);
    }
  }
}
```

———-

3.2.31 init_spad_process_list

— **sman.initspadprocesslist** —

```
static void
init_spad_process_list(void)
{
  spad_process_list = NULL;
}
```

———-

3.2.32 print_spad_process_list

— **sman.printspadprocesslist** —

```
#if 0
static void
print_spad_process_list()
{
  SpadProcess *proc;
  for(proc = spad_process_list; proc != NULL; proc = proc->next)
    fprintf(stderr, "proc_id = %d, death_action = %d\n", proc->proc_id,
    proc->death_action);
}
#endif
```

———-

3.2.33 find_child

— **sman.findchild** —

```
static SpadProcess *
find_child(int proc_id)
{
  SpadProcess *proc;
  for(proc = spad_process_list; proc != NULL; proc = proc->next)
    if (proc->proc_id == proc_id) return proc;
  return NULL;
}
```

———-

3.2.34 kill_all_children

— **sman.killallchildren** —

```
static void
kill_all_children(void)
{
  char name[256];
  SpadProcess *proc;


  for(proc = spad_process_list; proc != NULL; proc = proc->next) {
    kill(proc->proc_id, SIGTERM);
  }
  sprintf(name, "/tmp/hyper%d.input",server_num);
  unlink(name);

}
```

————-

3.2.35 clean_up_terminal

— **sman.cleanupterminal** —

```
static void
clean_up_terminal(void)
{
  tcsetattr(0, TCSAFLUSH, &oldbuf);
}
```

————-

3.2.36 monitor_children

— **sman.monitorchildren** —

```
static void
monitor_children(void)
{
  int dead_baby, stat;
  SpadProcess *proc;
```

```
while (1) {
  stat = 0;
  dead_baby = wait(&stat);
  /* Check the value of dead_baby, since wait may have returned
     a pid but subsequently we have received a signal.  Yeuch!
     In order to restart hyperdoc from the axiom command prompt
     we no longer call clean_up_terminal */
  if (dead_baby == -1 && death_signal) {
    kill_all_children();
    clean_up_sockets();
    sleep(2);
    exit(0);
  }
  /* Check the value of dead_baby, since wait may have returned
     a pid but subsequently we have received a signal.  Yeuch! */
  if(dead_baby == -1 && nagman_signal) {
    nagman_signal=0;
    spawn_of_hell(NagManagerProgram,NadaDelShitsky);
    continue;
  }

  if (dead_baby == -1) {
    fprintf(stderr, "sman: wait returned -1\n");
    continue;
  }
  proc = find_child(dead_baby);
  if (proc == NULL) {
    /*      fprintf(stderr, "sman: %d is not known to be a child process\n",
    dead_baby);
    */
    continue;
  }
  switch(proc->death_action) {
  /* In order to restart hyperdoc from the axiom command prompt
     we no longer call clean_up_terminal. Instead we've added a
     case to just clean up the socket. */
  case Die:
    kill_all_children();
    clean_up_sockets();
    sleep(2);
    exit(0);
  case NadaDelShitsky:
    break;
  case DoItAgain:
    spawn_of_hell(proc->command, DoItAgain);
    break;
  case CleanHypertexSocket:
    clean_hypertex_socket();
    break;
  }
```

```
  }
}
```

———-

3.2.37 main sman

The main procedure should return an int. We change the return value here and in src/include/sman.h1.

— **sman.result** —

```
  return(0);
```

———-

— **sman.main** —

```
int
main(int argc, char *argv[],char *envp[])
{
  if (tpd == 1) fprintf(stderr,"sman:main entered\n");
  bsdSignal(SIGINT,  SIG_IGN,RestartSystemCalls);
  process_options(argc, argv);

  init_term_io();
  init_spad_process_list();
  start_the_Axiom(envp);
  if (open_server(SessionIOName) == -2) {
    fprintf(stderr, "Fatal error opening I/O socket\n");
    clean_up_sockets();
    exit(-1);
  }
  start_the_session_manager();
  if (start_spadclient)       start_the_spadclient();
  if (start_local_spadclient) start_the_local_spadclient();
  if (start_nagman)           start_the_nagman();
  if (start_ht)               start_the_hypertex();
  if (start_graphics)         start_the_graphics();
  sleep(1);

  if (fork_you(Die) != NULL) {
    sman_catch_signals();
    monitor_children();
    exit(0);
  }
  manage_spad_io(ptcNum);
  if (tpd == 1) fprintf(stderr,"sman:main exit\n");
```

```
\getchunk{sman.result}
}
```

———-

3.2.38 sman

— sman —

```
#define _SMAN_C

\getchunk{sman.includes}
\getchunk{sman.variables}
\getchunk{sman.processarguments}
\getchunk{sman.shouldIclef}
\getchunk{sman.inX}
\getchunk{sman.setupdefaults}
\getchunk{sman.processoptions}
\getchunk{sman.deathhandler}
\getchunk{sman.nagmanhandler}
\getchunk{sman.smancatchsignals}
\getchunk{sman.fixenv}
\getchunk{sman.inittermio}
\getchunk{sman.strPrefix}
\getchunk{sman.checkspadproc}
\getchunk{sman.cleanupoldsockets}
\getchunk{sman.forkyou}
\getchunk{sman.execcommandenv}
\getchunk{sman.spawnofhell}
\getchunk{sman.startthespadclient}
\getchunk{sman.startthelocalspadclient}
\getchunk{sman.startthenagman}
\getchunk{sman.startthesessionmanager}
\getchunk{sman.startthehypertex}
\getchunk{sman.startthegraphics}
\getchunk{sman.forkAxiom}
\getchunk{sman.starttheAxiom}
\getchunk{sman.cleanupsockets}
\getchunk{sman.readfromspadio}
\getchunk{sman.readfrommanager}
\getchunk{sman.managespadio}
\getchunk{sman.initspadprocesslist}
\getchunk{sman.printspadprocesslist}
\getchunk{sman.findchild}
\getchunk{sman.killallchildren}
\getchunk{sman.cleanupterminal}
\getchunk{sman.monitorchildren}
```

```
\getchunk{sman.main}
```

———-

Chapter 4

Support Routines

4.1 Command Completion

Hyperdoc has the ability to do command completion. The known commands are listed, one entry per line, in a file called command.list.

Chapter 5

The viewman program

— the viewman command line —

```
char *GraphicsProgram = "$AXIOM/lib/viewman";
```

———-

Chapter 6

The nagman program

— the nagman command line —

```
char *NagManagerProgram = "$AXIOM/lib/nagman";
```

————-

6.1 nag.x

— nag.nag.x —

```
/*
 * msg.x: Remote message printing protocol
 */
const MAXASP = 10;

/*
 * the nago structure is essentially a variable length string
 */

struct nago {
  opaque z <>;
  };
struct nagerr {
nago p;
nago q;
};

struct host{
```

```
nago h <>;
};

struct nagst {

/* Okay, if you understand this bit you know the essentials of how the link
 * works. h <> is an array of nago, which is an array of fortran source
 * code, the length of the array being the no. of asps (0 for most routines).
 * y is the actual (XDR) input data for the routine. nm is the name of the
 * routine. id is a tag identifying the host/axiom session. Finally per is a
 * number telling whether or not to erase old fortran files on the remote
 * machine (persistence - the number per distinct fortran files will be
 * stored, any more than this and earlier ones will be deleted.
 */

  nago h <>;
  nago y;
  nago nm;
  nago id;
  int per;
  };
program NAGPROG {
   version NAGVERS {
       nagerr CALLNAG(nagst) = 1;
       nago NAGMON(int)=2;
       void AXEND(nago)=3;
   } = 1;
/*
 * the following number is very important. It tells the
 * portmapper what number to register the nag daemon under.
 * There are rules about which number to pick - check SUN
 * technical info for more details
 */
} = 100088;
```

———-

6.2 nagman

6.2.1 includes

— **nag.includes** —

```
#include <unistd.h>
#include <stdlib.h>
```

```
#include <stdio.h>
#include <string.h>
#include <errno.h>
#include <termios.h>
#include <signal.h>
#include <sys/time.h>
#include <sys/stat.h>
#include <sys/socket.h>
#include <sys/un.h>
#include <rpc/rpc.h>     /* always needed */
#include <fcntl.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <netdb.h>
#include "nag.h"  /* generated by rpcgen */
#include "com.h"
#include "bsdsignal.h"
#include "sockio-c.h1"
#include "bsdsignal.h1"
#include "nagman.h1"
```

———-

6.2.2 variables

— **nag.variables** —

```
#ifdef ALPHAplatform
extern int getdomainname( char *, int );
#endif
#ifdef SUN4OS5platform
extern int getdomainname( char *, int );
extern int gethostname( char *, int );
#endif

nagerr * callnag_1(nagst *,CLIENT *);
nago * nagmon_1(int *,CLIENT *);
void * axend_1(nago *,CLIENT *);

#define DO 1
#define DONT 0

int hnum, vmax;
char *datafile, *resultsfile;

struct hostnode {
  char * name;
```

```
  struct hostnode *next;
} *hlist=NULL;

nagst nag4;
Sock *sock1;
```

6.2.3 term

this code runs when the user quits axiom. before nagman dies, it does an rpc call to nagd to tell it to get rid of files etc. The rpc call in question is axend_1 we also send a USR1 to sman to beget another nagman

— **nag.term** —

```
static void
term(int sig)
{
  CLIENT *cld;
  void *res;
  struct hostnode *pnode;

#ifndef HP9platform /* can't figure out a way to do this on HP/UX 9 */
  kill(atoi(getenv("SPADNUM")) , SIGUSR1);
#endif


  if(hnum!=0)
    {
      unlink(datafile);
      unlink(resultsfile);
    }

  for(pnode=hlist;pnode!=NULL;pnode=pnode->next)
    {
      cld=clnt_create(pnode->name,NAGPROG, NAGVERS, "tcp");
      if (cld == NULL)
goto NOHOST;

      res=axend_1(&(nag4.id),cld);
    NOHOST:
      clnt_destroy(cld);
    }
  exit(0);
}
```

6.2.4 size_of_file

— **nag.sizeoffile** —

```
static long
size_of_file(char *filename)
{
  struct stat buf_stat;

  stat(filename,&buf_stat);
  return (buf_stat.st_size);

}
```

———-

6.2.5 rpcloop

— **nag.rpcloop** —

```
static void
rpcloop(void)
{
  CLIENT *cl;
  int res,j,v=0,u,showMessage;
  long i;
  register struct hostent *alias1, *alias2;
  struct in_addr *addrnum;
  u_long junk;
  struct timeval tv;
  nagerr *result;
  char *Buf , *buf1;
  char *ffile[MAXASP];
  char routine[12], naghost[256];
  FILE *nfp1, *nfp2, *nfp3;
  struct hostnode *phost;
  int fd;

  for (;;)
    {

      if((Buf=get_string(sock1))==NULL) term(1); /* one string carries all */

      if(hnum!=0)
{
```

```
  /* call parameters */
  free(nag4.nm.z.z_val); /* the routine name */
  free(nag4.y.z.z_val);  /* the XDR data */
  for(i=0;i<v;i++)
    {
      unlink(ffile[i]);
      free(ffile[i]); /* the asp filenames */
      free(nag4.h.h_val[i].z.z_val); /* the asps themselves*/
    }
  free(nag4.h.h_val); /* the asps array */
  unlink(datafile);
  unlink(resultsfile);
  free(resultsfile);
  free(datafile);
  vmax= (v>vmax)? v : vmax;
}



      buf1=strtok(Buf," ");
      if (buf1) strcpy(naghost,buf1);
      else printf("can't parse the naghost\n");
      /* INFO         printf("%s\n",naghost);*/

      buf1=strtok(NULL," ");
      if (buf1) strcpy(routine,buf1);
      else printf("can't parse the routine\n");
      /* INFO         printf("%s\n",routine);*/

      /* make copy of filenames because we will reuse Buf before deleting the files*/
      buf1=strtok(NULL," ");
      if (buf1) resultsfile=strdup(buf1);
      else printf("can't parse the resultsfile file\n");
      /* INFO         printf("%s\n",resultsfile);*/

      buf1=strtok(NULL," ");
      if (buf1) datafile=strdup(buf1);
      else printf("can't parse the datafile file\n");
      /* INFO         printf("%s\n",datafile);*/

      buf1=strtok(NULL," ");
      if (buf1) nag4.per=atoi(buf1);
      else printf("can't parse the persistence\n");
      /* INFO         printf("%d\n",nag4.per);*/

      buf1=strtok(NULL," ");
      if (buf1) {
if (!strcmp(buf1,"on")) showMessage=DO;
else showMessage=DONT;
```

```
      }
      else printf("can't parse the messages flag\n");
      /* INFO         printf("%s\n",buf1);*/

      v=0; /* asp counter */
      while( (buf1=strtok(NULL," ")) )
{
  ffile[v++]=strdup(buf1);
  /* INFO         printf("%s\n",ffile[v-1]);*/
}

      /* INFO   printf("number of asps seen %d\n",v);*/

      if(showMessage==DO) printf("nagman:acknowledging request for %s\n",routine);

      res=0;  /* prepare result integer to be sent to Lisp */

      if((nfp3=fopen(resultsfile,"w"))==NULL)
{
  printf("can't open output file\n");
  goto END;
}

      /* nag4.h is the XDR array of asp text */
      nag4.h.h_len=v;
      nag4.h.h_val=(nago *)malloc((v)*sizeof(nago));


      /* get asp text in call argument */
      for(u=0;u<v;u++)
{
  /* this should be done by mmap */
  if((nfp1=fopen(ffile[u],"r"))==NULL)
    {
      fprintf(stderr,"can't open asp file %s\n",ffile[u]);
      fclose(nfp1);
      goto END;
    }
  fclose(nfp1);
  i=size_of_file(ffile[u]);

  /* allocs memory for the file */
  nag4.h.h_val[u].z.z_val= (char *)malloc((i+1)*sizeof(char));

  fd=open(ffile[u],O_RDONLY);
  read(fd,nag4.h.h_val[u].z.z_val,i);
  close(fd);
  /* make null-term. string */
  nag4.h.h_val[u].z.z_val[i]='\0';
  /* set the length */
```

```
  nag4.h.h_val[u].z.z_len=strlen(nag4.h.h_val[u].z.z_val);
}


      nag4.nm.z.z_val=strdup(routine);
      nag4.nm.z.z_len=strlen(routine);

      /* get XDR data in call argument */
      /* should be done by mmap */
      if((nfp2=fopen(datafile,"r"))==NULL)
{
  fprintf(stderr,"can't open data file\n");
  fclose(nfp2);
  goto END;
}

      fclose(nfp2);
      i=size_of_file(datafile);
      nag4.y.z.z_val=(char *)malloc(i*sizeof(char));

      fd=open(datafile,O_RDONLY);
      read(fd,nag4.y.z.z_val,i);
      close(fd);
      nag4.y.z.z_len=i;


      /*
       * Create client "handle" used for calling MESSAGEPROG on
       * the server designated on the command line.  We tell
       * the RPC package to use the "tcp" protocol when
       * contacting the server.
       */

      /* update naghost by lookup */

      if ((junk = inet_addr(naghost))!=-1)
{
  addrnum=(struct in_addr *)junk;
  if((alias2=gethostbyaddr((char *)&addrnum,
   sizeof(addrnum),
   AF_INET))!=NULL)
    strcpy(naghost,alias2->h_name);
  else
    if((alias1=gethostbyname(naghost))!=NULL)
      strcpy(naghost,alias1->h_name);
}
      else
if((alias1=gethostbyname(naghost))!=NULL)
  strcpy(naghost,alias1->h_name);
```

```
      cl = clnt_create(naghost, NAGPROG, NAGVERS, "tcp");
      if (cl == NULL)
{
  /*
   * Couldn't establish connection with server.
   * Print error message and die.
   */
  clnt_pcreateerror(naghost);
  goto END;
}
      else
if (showMessage==DO)
  printf("nagman:connection successful to %s\n",naghost);

      /*
       * this number here sets the "timeout" for the rpc call. after this number
       * of seconds, the call will quit if no response is received
       *
       */

      tv.tv_sec=1000000;
      tv.tv_usec=0;
      clnt_control(cl,CLSET_TIMEOUT,(char *)&tv);


      result = callnag_1(&nag4, cl);

      for(phost=hlist;phost!=NULL;phost=phost->next)
{
  /*
   * hlist is the "hostlist" of sites that have been contacted by nagman.
   * here we check if this call is contacting a new site, and if so add it
   * to the hostlist
   *
   */

  if(!strcmp(phost->name,naghost))
    goto SKIP;
}

      if(hnum==0) {
hlist=(struct hostnode *)malloc(sizeof(struct hostnode));
hlist->name=strdup(naghost);
hlist->next=NULL;
      }

      else {
```

```
phost=(struct hostnode *)malloc(sizeof(struct hostnode));
phost->name=strdup(naghost);
phost->next=hlist;
hlist=phost;
      }
      hnum++;


    SKIP:
      if (result == NULL)
{
  /*
   * An error occurred while calling the server.
   * Print error message and die.
   */
  if (showMessage==DO)
    printf("nagman:no results (error) from %s\n",naghost);
  clnt_perror(cl,naghost);
  clnt_destroy(cl);
  goto END;
}

      /*
       * (*result).p is the part of the result with the XDRed results in it
       * (numbers). (*result).q is the part with (text) error messages that
       * have come from the NAG library. If there is neither an XDR result,
       * nor a text error message from the library, then something is wrong
       * so we just print out the "no result or error returned" message.
       *
       */

      else if ((*result).p.z.z_len==0)
{
  if((*result).q.z.z_len==0)
    {
      if (showMessage==DO)
printf("nagman:empty result (error) from %s\n",naghost);
      clnt_destroy(cl);
      goto END;
    }
  else
    {
      if (showMessage==DO)
printf("nagman:receiving results from %s\n\n",naghost);
      for(j=0;j<(*result).q.z.z_len;j++)
printf("%c",(*result).q.z.z_val[j]);
      clnt_destroy(cl);
      goto END;
    }
}
```

```
      else
if (showMessage==DO)
  printf("nagman:receiving results from %s\n\n",naghost);

      if (showMessage==DO)
fwrite(result->q.z.z_val,sizeof(char),result->q.z.z_len,stdout);

      /*INFO printf("\nRESULTS of length %d\n",(*result).p.z.z_len);*/

      fwrite(result->p.z.z_val,sizeof(char),result->p.z.z_len, nfp3);
      res=1;
      clnt_destroy(cl);

      /*
       * in case of any type of error, a goto END in the above code causes
       * nagman to skip here and return to AXIOM
       *
       */


    END:
      fclose(nfp3);
      /*
       * if everything has gone alright, send_int returns the integer res=1. If
       * not it returns res=0. This is detected by the boot code which acts
       * accordingly.
       */
      send_int(sock1,res);
      free(Buf);
    }


}
```

———-

6.2.6 catchSignals

catchSignals sets up signal handling. If nagman gets a sigterm it does not die but goes back to rpcloop

— **nag.catchSignals** —

```
static void
catchSignals(void)
{
  bsdSignal(SIGTERM,term,RestartSystemCalls);
  bsdSignal(SIGSEGV,term,RestartSystemCalls);
}
```

———-

6.2.7 main nagman

— **nag.main** —

```
void
main(int argc,char **argv)
{
  char this[256],*hname,*dname,*spadnum;
  int stat;

  catchSignals();
  stat=gethostname(this,256);
  if (stat!=0) perror("gethostname");
  hname=strdup(this);

  stat=getdomainname(this,256);
  if (stat!=0) perror("getdomainname");
  dname=strdup(this);
  spadnum=getenv("SPADNUM");
  if (spadnum==0) {
    fprintf(stderr,"nagman error: SPADNUM is not in the environment\n");
    exit(0);
  }

  /* some machines return a full name from hostname
     need to check hname has a . in it */

  if  (strchr(hname,'.'))
    /* '.' found */
    sprintf(this,"%s_%i",hname,atoi(spadnum));
  else
    /* substring not found */
    sprintf(this,"%s.%s_%i",hname,dname,atoi(spadnum));

  /* this must contain the Internet address of the current host */
  nag4.id.z.z_val=strdup(this);
  nag4.id.z.z_len=strlen(nag4.id.z.z_val);
  hnum=0;
  vmax=0;
  /*
   * this line sets up a socket for communication with the lisp
   */

  sock1 = connect_to_local_server(SpadServer, DebugWindow, 120 /*seconds*/);
```

```
  if (!sock1) exit(0);

  rpcloop();
}
```

———-

6.2.8 nagman

— **nagman** —

```
#define _NAGMAN_C
\getchunk{nag.includes}
\getchunk{nag.variables}
\getchunk{nag.term}
\getchunk{nag.sizeoffile}
\getchunk{nag.rpcloop}
\getchunk{nag.catchSignals}
\getchunk{nag.main}
```

———-

Chapter 7

The hypertex program

— the hypertex command line —

```
char *HypertexProgram = "$AXIOM/bin/hypertex -s";
```

———-

Chapter 8

The `clef` program

— the clef command line —

```
char *ClefProgram = "$AXIOM/bin/clef -f $AXIOM/lib/command.list -e ";
```

———-

Chapter 9

The session program

— the session manager command line —

```
char *SessionManagerProgram = "$AXIOM/lib/session";
```

———-

9.1 session

9.1.1 includes

— ses.includes —

```
#include <stdlib.h>
#include <sys/time.h>
#include <stdio.h>
#include <string.h>
#include <signal.h>
#ifdef SGIplatform
#include <bstring.h>
#endif
#include "com.h"
#include "bsdsignal.h"
#include "sockio-c.h1"
#include "bsdsignal.h1"
#include "session.h1"
```

———-

9.1.2 variables

— **ses.variables** —

```
#define BufSize 4096 /* size of communication buffer */

typedef struct sock_list {      /* linked list of Sock */
  Sock Socket;
  struct sock_list *next;
} Sock_List;

Sock *spad_io = (Sock *) 0;    /* to_server socket for SessionIO */
Sock *spad_server = (Sock *) 0;    /* to_server socket for SpadServer    */
Sock *menu_client = (Sock *) 0;    /* to_client socket for MenuServerName  */
Sock *active_session = (Sock *) 0; /* pointer to currently active session */

Sock_List *plSock = (Sock_List *) 0;

char big_bad_buf[BufSize]; /* big I/O buffer */
int num_active_clients = 0; /* number of InterpWindows attached */
int reading_output = 0;
fd_set session_socket_mask;
```

———-

9.1.3 usr1_handler

— **ses.usr1handler** —

```
static void
usr1_handler(int sig)
{
  return;
}
```

———-

9.1.4 usr2_handler

SIGUSR2 is generated by spadclients. We interpret it as an interrupt for the Lisp.

— **ses.usr2handler** —

```
static void
```

```
usr2_handler(int sig)
{
  send_signal(spad_server, SIGINT);
  return;
}
```

———-

9.1.5 term_handler

— **ses.termhandler** —

```
static void
term_handler(int sig)
{
  exit(1);
}
```

———-

9.1.6 pr

— **ses.pr** —

```
static void
pr()
{
  Sock_List *pSock;

  fprintf(stderr,"The socket list:\n");
  for(pSock=plSock;pSock!=(Sock_List *)0;pSock=pSock->next){
    fprintf(stderr,"(%d,%d,%d)\t",
      pSock->Socket.pid, 2<<(pSock->Socket.socket), pSock->Socket.frame);
  }
  fprintf(stderr,"\n");
}
```

———-

9.1.7 close_client

— **ses.closeclient** —

```
static void
close_client(int frame)
{
  Sock_List *pSock,*locSock;
  int socket_fd;

  /* we will check for frame equality,
     kill with send_signal,
     notify HyperTex so that it updates its list (if it's a spadbuf),
     repair the list,
     unset the active_session,
     update num_active_clients
     */


  /* first check head */
#ifdef DEBUG
fprintf(stderr,"close_client(%d)\n",frame);
#endif

  if ( (plSock) && (plSock->Socket.frame == frame) ){
    socket_fd = plSock->Socket.socket;
    send_signal((Sock *)plSock, SIGTERM);
    if ( menu_client != (Sock *) 0){
      send_int(menu_client,CloseClient);
      send_int(menu_client,(*plSock).Socket.pid);
    }
#ifdef DEBUG
fprintf(stderr,"trying to clear %u\n",socket_fd);
#endif
    FD_CLR(socket_fd,&session_socket_mask);
    locSock = plSock;
    if ((*plSock).next == (Sock_List *) 0)
      {plSock = (Sock_List *) 0;}
    else
      {plSock = plSock->next;}
    active_session = (Sock *) 0;
    num_active_clients--;
    free(locSock);
  }

  /* now check the rest */

  else {
    for (pSock=plSock; pSock->next != (Sock_List *) 0 ; pSock=pSock->next)
```

```
      if (pSock->next->Socket.frame == frame){
socket_fd = pSock->next->Socket.socket;
send_signal((Sock *)pSock->next, SIGTERM);
if ( menu_client != (Sock *) 0){
  send_int(menu_client,CloseClient);
  send_int(menu_client,(*plSock).Socket.pid);
}
#ifdef DEBUG
fprintf(stderr,"trying to clear %u\n",socket_fd);
#endif
FD_CLR(socket_fd,&session_socket_mask);
locSock = pSock->next;
if (  pSock->next->next == (Sock_List *) 0  )
  { pSock->next= (Sock_List *) 0;}
else
  { pSock->next = pSock->next->next;}
num_active_clients--;
active_session = (Sock *) 0;
free(locSock);
break;
      }
  }
#ifdef DEBUG
pr();
#endif
}
```

———-

9.1.8 read_SpadServer_command

— **ses.readSpadServercommand** —

```
static void
read_SpadServer_command(void)
{
  int cmd, frame, num;
  cmd  = get_int(spad_server);
  switch (cmd) {
  case EndOfOutput:
    if (menu_client != (Sock *) 0) send_signal(menu_client, SIGUSR2);
    if (reading_output != 0) reading_output = 0;
    break;
  case QueryClients:
    /*  don't count MenuServer */
    num =  num_active_clients ;
    send_int(spad_server, num);
```

```
    break;
  case CloseClient:
    frame = get_int(spad_server);
    if (frame != -1) close_client(frame);
    break;
  case SendXEventToHyperTeX:
    break;
  default:
    fprintf(stderr, "session : unknown command from SpadServer %d\n", cmd);
    break;
  }
}
```

———-

9.1.9 test_sock_for_process

— **ses.testsockforprocess** —

```
static int
test_sock_for_process(Sock *sock)
{
  if (sock == (Sock *)0 ) return -1;
  return kill(sock->pid, 0);
}
```

———-

9.1.10 read_menu_client_command

— **ses.readmenuclientcommand** —

```
static void
read_menu_client_command(void)
{
  int cmd,frame, i,socket_fd;
  Sock_List *pSock;

  /* save it for possible clearing */
  socket_fd =  menu_client->socket;

  if (test_sock_for_process(menu_client) == -1) {
    FD_CLR(socket_fd,&session_socket_mask);
```

```
    menu_client = (Sock *) 0;
    reading_output = 0;
    return;
  }
  cmd = get_int(menu_client);
  switch(cmd) {
  case -1: /* socket closed */
    FD_CLR(socket_fd,&session_socket_mask);
    menu_client = (Sock *) 0;
    reading_output = 0;
    break;
  case SwitchFrames:
#ifdef DEBUG
fprintf(stderr,"menu:SwitchFrames\n");
#endif
    frame = get_int(menu_client);
    send_int(spad_server, SwitchFrames);
    send_int(spad_server, frame);
    for(i=0,pSock=plSock; pSock != (Sock_List *) 0 ; i++,pSock=pSock->next)
      if ((pSock->Socket.frame == frame)) {
active_session = (Sock *)pSock;
reading_output = 1;
break;
      }
    if (i == num_active_clients) {
      /* fprintf(stderr, "Couldn't find socket for frame %d\n", frame); */
    }
    break;
  case QuerySpad:
#ifdef DEBUG
fprintf(stderr,"menu:QuerySpad\n");
#endif
    send_int(menu_client, reading_output);
    break;
  default:
    fprintf(stderr, "session : unknown command from MenuServer: %d\n", cmd);
    menu_client = (Sock *) 0;
    break;
  }
}
```

———-

9.1.11 read_from_spad_io

— **ses.readfromspadio** —

```
static void
read_from_spad_io(void)
{
  int ret_code;
  ret_code = sread(spad_io, big_bad_buf, BufSize, "session: stdout socket");
  if (ret_code == -1) return;
  if(active_session != (Sock *) 0) {
    ret_code = swrite(active_session, big_bad_buf, ret_code,
      NULL);
  }
}
```

———-

9.1.12 kill_spad

— **ses.killspad** —

```
static void
kill_spad(void)
{
  int i;
  Sock_List *pSock;

  send_signal(spad_server, SIGTERM);
  for  (pSock=plSock,i=0;
(i<num_active_clients) && (pSock != (Sock_List *) 0);
i++,pSock=pSock->next) {
    if ((pSock->Socket).socket != 0)
      send_signal((Sock *)pSock, SIGTERM);
  }
  if (menu_client != (Sock *) 0) send_signal(menu_client, SIGTERM);
  exit(0);
}
```

———-

9.1.13 accept_session_connection

— **ses.acceptsessionconnection** —

```
static int
accept_session_connection(Sock *server_sock)
```

```
{
  int sock_fd, ret_code;
  Sock_List *pls;

  /* Could be three things : KillSpad MenuServer InterpWindow  */

  pls = (Sock_List *) malloc(sizeof (Sock_List));
  sock_fd = accept(server_sock->socket, 0, 0);
  if (sock_fd == -1) {
    perror("session : accepting connection");
    return -1;
  }
  (pls->Socket).socket = sock_fd;
    get_socket_type((Sock *)pls);

    switch((pls->Socket).purpose) {
    case KillSpad:
      kill_spad();
      return KillSpad;
      free(pls);
    case MenuServer:
#ifdef DEBUG
      fprintf(stderr,"session: accepted MenuServer , fd = %d\n",sock_fd);
#endif
      menu_client = &(pls->Socket);
      FD_SET(menu_client->socket, &session_socket_mask);
      return MenuServer;
    case InterpWindow:
#ifdef DEBUG
      fprintf(stderr,"session: accepted InterpWindow , fd = %d\n",sock_fd);
#endif

      /* new Sock is put at the head of the list */
      if (plSock == (Sock_List *)0 ) {
plSock = pls;
plSock->next = (Sock_List *)0 ;
      }
      else{
pls->next = plSock;
plSock = pls;
      }

      /* we need to maintain session_socket_mask here
         since we roll our own accept */

      FD_SET(plSock->Socket.socket, &session_socket_mask);
      send_int(spad_server, CreateFrame);
      {
          int command = get_int(spad_server);
          /* XXX hack -- the whole protocol looks broken, we just
```

```
          try to detect losage */
          if (command != CreateFrameAnswer) {
              fprintf(stderr, "session: non-fatal, got out of sync "
                              "with Spad server\n  (lost race)\n");
          /*    exit(1); */
          }
      }
      plSock->Socket.frame = get_int(spad_server);
      active_session = (Sock *)plSock;
      get_string_buf(spad_server, big_bad_buf, BufSize);
      ret_code = swrite((Sock *)plSock, big_bad_buf, strlen(big_bad_buf)+1,
"session: writing to InterpWindow");
      if (ret_code == -1)
return -1;
      num_active_clients++;
#ifdef DEBUG
pr();
#endif
      return plSock->Socket.purpose;
    }
    return (-1);
}
```

———-

9.1.14 read_from_session

— **ses.readfromsession** —

```
static void
read_from_session(Sock *sock)
{
  int ret_code;
  if (sock != active_session) {
    send_int(spad_server, SwitchFrames);
    send_int(spad_server, sock->frame);
  }
  active_session = sock;
  ret_code = sread(sock, big_bad_buf, BufSize,
   "session: reading InterpWindow");
  if (ret_code == -1) {
    active_session = (Sock *) 0;
    reading_output = 0;
    return;
  }
  ret_code = swrite(spad_io, big_bad_buf, ret_code,
    "session: writing SessionIO");
```

```
  if (ret_code == -1) {
    active_session = (Sock *)0 ;
    reading_output = 0;
    return;
  }
  reading_output = 1;
}
```

———-

9.1.15 manage_sessions

— ses.managesessions —

```
static void
manage_sessions(void)
{
  int ret_code;
  fd_set rd, wr, ex;
  Sock_List  *pSock;

  reading_output = 0;
  while (1) {
    FD_ZERO(&rd);
    FD_ZERO(&wr);
    FD_ZERO(&ex);

    /* Allow server socket and all connections if not waiting for output
       socket_mask is maintained by libspad.a  */
#ifdef DEBUG
fprintf(stderr,"session_socket_mask=%u ",*((long *)session_socket_mask.fds_bits));
#endif
    rd = session_socket_mask;
    if (!reading_output) {
      rd = session_socket_mask;
    }

    /* Allow the active_session if set */
    if (active_session) FD_SET(active_session->socket, &rd);
#ifdef DEBUG
fprintf(stderr,"[rd=%u ",*((long *)rd.fds_bits));
#endif

    ret_code = sselect(FD_SETSIZE, &rd, &wr, &ex, NULL);
    if (ret_code == -1) {
break;
    }
```

```
#ifdef DEBUG
fprintf(stderr,"rd=%u]\n",*((long *)rd.fds_bits));
#endif

    if ((menu_client != (Sock *) 0)  && FD_ISSET(menu_client->socket, &rd)) {
      /* MenuServer wants to talk */
      read_menu_client_command(); }


    if (FD_ISSET(spad_io->socket, &rd)) {
      /* Lisp has output */
      read_from_spad_io(); }


    if (FD_ISSET(server[1].socket, &rd)) {
      /* Someone wants to connect to our server socket */
      accept_session_connection(server+1); }


    for(pSock=plSock; pSock != (Sock_List *) 0 ; pSock=pSock->next) {
      if ((active_session == (Sock *)pSock || !reading_output) &&
  (pSock->Socket).socket>0 && FD_ISSET(pSock->Socket.socket, &rd)) {
/* An InterpWindow */
read_from_session((Sock *)pSock); }
    }


    if (FD_ISSET(spad_server->socket, &rd)) {
      /* The Lisp socket */
      read_SpadServer_command(); }
  }
}
```

———-

9.1.16 main sessionmanager

— **ses.main** —

```
int
main(void)
{

#ifdef DEBUG2
  /* delay for attaching with debugger before interesting things happen */
  sleep(30);
#endif
```

```
 /* spad_server connects to Lisp server socket
    read_SpadServer_command handles requests */
  spad_server = connect_to_local_server(SpadServer, SessionManager, Forever);
  if (spad_server == (Sock *) 0) {
    fprintf(stderr, "session: Cannot connect to AXIOM server!\n");
    exit(0);
  }
  else {
#ifdef DEBUG
    fprintf(stderr, "session: connected SpadServer , fd = %d\n",
    spad_server->socket);
#endif
    FD_SET(spad_server->socket, &session_socket_mask);
  }


  /* spad_io connects to SessionIOName server socket
    this is Lisp std IO read_from_spad_io handles requests */
  spad_io = connect_to_local_server(SessionIOName, SessionIO, Forever);
  if (spad_io == (Sock *) 0) {
    fprintf(stderr, "session: Cannot connect to AXIOM IO!\n");
    exit(0);
  }
  else {
#ifdef DEBUG
    fprintf(stderr,"session: connected SessionIOName , fd = %d\n",
    spad_io->socket);
#endif
    FD_SET(spad_io->socket, &session_socket_mask);
  }
  bsdSignal(SIGUSR2, usr2_handler,DontRestartSystemCalls);
  bsdSignal(SIGUSR1, usr1_handler,RestartSystemCalls);
  bsdSignal(SIGINT,  SIG_IGN,RestartSystemCalls);
  bsdSignal(SIGTERM, term_handler,RestartSystemCalls);

  /* open_server opens the server socket so that we can accept connections
    we expect connections from spadbuf/spadclient(purpose:InterpWindow)
    and hypertex (MenuServer) */

  if (open_server(SessionServer) == -2) {
    fprintf(stderr, "session: Cannot make server socket!\n");
    exit(-1);
  }
  else {
#ifdef DEBUG
    fprintf(stderr, "session: opened SessionServer , fd = %d\n",
    server[1].socket);
#endif
    FD_SET(server[1].socket,&session_socket_mask);
```

```
  }
  manage_sessions();
  return(0);
}
```

———-

9.1.17 session

— **session** —

```
/* #define DEBUG */
#define _SESSION_C

\getchunk{ses.includes}
\getchunk{ses.variables}
\getchunk{ses.usr1handler}
\getchunk{ses.usr2handler}
\getchunk{ses.termhandler}
\getchunk{ses.pr}
\getchunk{ses.closeclient}
\getchunk{ses.readSpadServercommand}
\getchunk{ses.testsockforprocess}
\getchunk{ses.readmenuclientcommand}
\getchunk{ses.readfromspadio}
\getchunk{ses.killspad}
\getchunk{ses.acceptsessionconnection}
\getchunk{ses.readfromsession}
\getchunk{ses.managesessions}
\getchunk{ses.main}
```

———-

Chapter 10

The spadclient program

— the spadclient command line —

```
char *SpadClientProgram = "$AXIOM/lib/spadclient";
```

————-

10.1 spadclient

— spadclient —

```
#define _SPADCLIENT_C

#include <stdio.h>
#include <signal.h>
#include "com.h"
#include "bsdsignal.h"

#include "bsdsignal.h1"
#include "sockio-c.h1"
#include "spadclient.h1"

Sock *sock;

static void
inter_handler(int sig)
{
  send_signal(sock, SIGUSR2);
  fflush(stderr);
```

```
}


int
main(void)
{
  sock = connect_to_local_server(SessionServer, InterpWindow, Forever);
  bsdSignal(SIGINT, inter_handler,RestartSystemCalls);
  remote_stdio(sock);
  return(0);
}
```

———-

Chapter 11

The Command Completion List

— command.list —

```
-
/
/\
^
^=
~
~=
*
**
\/
#
+
<
<=
=
>
>=
0
1
abelianGroup
abs
absolutelyIrreducible?
accuracyIF
acos
acosh
acoshIfCan
acosIfCan
acot
acoth
```

```
acothIfCan
acotIfCan
acsc
acsch
acschIfCan
acscIfCan
aCubic
adaptive
adaptive?
adaptive3D?
addBadValue
addChild!
addData!
addField!
addiag
addMatch
addMatchRestricted
addmod
addPoint
addPoint2
addPointLast
adjoint
airyAi
airyBi
Aleph
algDsolve
algebraic?
algebraicCoefficients?
algebraicDecompose
algebraicOf
algebraicSort
algebraicVariables
algint
algintegrate
algSplitSimple
aLinear
allRootsOf
alphabetic
alphabetic?
alphanumeric
alphanumeric?
alternating
alternatingGroup
alternative?
An
AND
And
and
anfactor
antiAssociative?
```

antiCommutative?
antiCommutator
anticoord
antisymmetric?
antisymmetricTensors
any
any?
append
appendPoint
apply
applyQuote
applyRules
approximants
approximate
approxNthRoot
approxSqrt
aQuadratic
aQuartic
areEquivalent?
arg1
arg2
argscript
argument
argumentList!
argumentListOf
arity
aromberg
arrayStack
asec
asech
asechIfCan
asecIfCan
asimpson
asin
asinh
asinhIfCan
asinIfCan
aspFilename
assert
assign
assoc
associatedEquations
associatedSystem
associates?
associative?
associator
associatorDependence
atan
atanh
atanhIfCan

```
atanIfCan
atom?
atoms
atrapezoidal
att2Result
augment
autoReduced?
axes
axesColorDefault
B1solve
back
backOldPos
badNum
badValues
bag
balancedBinaryTree
balancedFactorisation
bandedHessian
bandedJacobian
base
baseRDE
baseRDEsys
BasicMethod
basicSet
basis
basisOfCenter
basisOfCentroid
basisOfCommutingElements
basisOfLeftAnnihilator
basisOfLeftNucleus
basisOfLeftNucloid
basisOfMiddleNucleus
basisOfNucleus
basisOfRightAnnihilator
basisOfRightNucleus
basisOfRightNucloid
bat
bat1
beauzamyBound
belong?
bernoulli
bernoulliB
besselI
besselJ
besselK
besselY
Beta
bezoutDiscriminant
bezoutMatrix
bezoutResultant
```

bfEntry
bfKeys
binary
binaryFunction
binarySearchTree
binaryTournament
binaryTree
binomial
binomThmExpt
bipolar
bipolarCylindrical
biRank
birth
bit?
bitCoef
bitLength
bits
bitTruth
bivariate?
bivariatePolynomials
bivariateSLPEBR
blankSeparate
block
blue
bombieriNorm
bool
bool?
bottom!
boundOfCauchy
box
brace
bracket
branchIfCan
branchPoint?
branchPointAtInfinity?
bright
brillhartIrreducible?
brillhartTrials
bringDown
bsolve
btwFact
bubbleSort!
build
BumInSepFFE
bumprow
bumptab
bumptab1
BY
c02aff
c02agf

```
c05adf
c05nbf
c05pbf
c06eaf
c06ebf
c06ecf
c06ekf
c06fpf
c06fqf
c06frf
c06fuf
c06gbf
c06gcf
c06gqf
c06gsf
cache
cAcos
cAcosh
cAcot
cAcoth
cAcsc
cAcsch
calcRanges
call
cap
car
cardinality
cartesian
cAsec
cAsech
cAsin
cAsinh
cAtan
cAtanh
cCos
cCosh
cCot
cCoth
cCsc
cCsch
cdr
ceiling
center
central?
certainlySubVariety?
cExp
cfirst
chainSubResultants
changeBase
changeMeasure
```

changeName
changeNameToObjf
changeThreshhold
changeVar
changeWeightLevel
char
character?
characteristic
characteristicPolynomial
characteristicSerie
characteristicSet
charClass
charpol
charthRoot
chebyshevT
chebyshevU
check
checkCxResult
checkForZero
checkMxCDF
checkMxDF
checkPrecision
checkResult
checkRur
child
child?
children
chineseRemainder
chiSquare
chiSquare1
choosemon
chvar
Ci
className
clearCache
clearDenominator
clearFortranOutputStack
clearTable!
clearTheFTable
clearTheIFTable
clearTheSymbolTable
clikeUniv
clip
clipBoolean
clipParametric
clipPointsDefault
clipSurface
clipWithRanges
cLog
close

```
close!
closeComponent
closed?
closedCurve
closedCurve?
cn
code
coef
coefChoose
coefficient
coefficients
coerce
coerceImages
coerceListOfPairs
coerceP
coercePreimagesImages
coHeight
coleman
collect
collectQuasiMonic
collectUnder
collectUpper
color
colorDef
colorFunction
column
combineFeatureCompatibility
commaSeparate
comment
common
commonDenominator
commutative?
commutativeEquality
commutator
comp
compactFraction
companionBlocks
comparison
compBound
compdegd
compile
compiledFunction
complement
complementaryBasis
complete
completeEchelonBasis
completeEval
completeHensel
completeHermite
completeSmith
```

complex
complex?
complexEigenvalues
complexEigenvectors
complexElementary
complexExpand
complexForm
complexIntegrate
complexLimit
complexNormalize
complexNumeric
complexNumericIfCan
complexRoots
complexSolve
complexZeros
component
components
compose
composite
composites
computeBasis
computeCycleEntry
computeCycleLength
computeInt
computePowers
concat
concat!
cond
condition
conditionP
conditions
conditionsForIdempotents
conical
conjHerm
conjug
conjugate
conjugates
connect
connect?
cons
consnewpol
const
constant
constant?
constantCoefficientRicDE
constantIfCan
constantKernel
constantLeft
constantOperator
constantOpIfCan

```
constantRight
constantToUnaryFunction
constDsolve
construct
contains?
content
continue
continuedFraction
contract
contractSolve
controlPanel
convergents
convert
coord
coordinate
coordinates
copies
copy
copy!
copyInto!
corrPoly
cos
cos2sec
cosh
cosh2sech
coshIfCan
cosIfCan
cosSinInfo
cot
cot2tan
cot2trig
coth
coth2tanh
coth2trigh
cothIfCan
cotIfCan
count
countable?
countRealRoots
countRealRootsMultiple
cPower
cRationalPower
create
create3Space
createGenericMatrix
createIrreduciblePoly
createLowComplexityNormalBasis
createLowComplexityTable
createMultiplicationMatrix
createMultiplicationTable
```

createNormalElement
createNormalPoly
createNormalPrimitivePoly
createPrimitiveElement
createPrimitiveNormalPoly
createPrimitivePoly
createRandomElement
createThreeSpace
createZechTable
credPol
crest
critB
critBonD
critM
critMonD1
critMTonD1
critpOrder
critT
cross
crushedSet
csc
csc2sin
csch
csch2sinh
cschIfCan
cscIfCan
cSec
cSech
cSin
cSinh
csubst
cTan
cTanh
cubic
cup
currentSubProgram
curry
curryLeft
curryRight
curve
curve?
curveColor
curveColorPalette
cycle
cycleElt
cycleEntry
cycleLength
cyclePartition
cycleRagits
cycles

```
cycleSplit!
cycleTail
cyclic
cyclic?
cyclicCopy
cyclicEntries
cyclicEqual?
cyclicGroup
cyclicParents
cyclicSubmodule
cyclotomic
cyclotomicDecomposition
cyclotomicFactorization
cylindrical
D
d01ajf
d01akf
d01alf
d01amf
d01anf
d01apf
d01aqf
d01asf
d01bbf
d01fcf
d01gaf
d01gbf
d02bbf
d02bhf
d02cjf
d02ejf
d02gaf
d02gbf
d02kef
d02raf
d03edf
d03eef
d03faf
dAndcExp
dark
datalist
ddFact
debug
debug3D
dec
decimal
declare
declare!
decompose
decomposeFunc
```

decrease
decreasePrecision
deepCopy
deepestInitial
deepestTail
deepExpand
defineProperty
definingEquations
definingInequation
definingPolynomial
degree
degreePartition
degreeSubResultant
degreeSubResultantEuclidean
delay
delete
delete!
deleteProperty!
deleteRoutine!
delta
denom
denominator
denominators
denomLODE
denomRicDE
depth
dequeue
dequeue!
deref
deriv
derivationCoordinates
derivative
destruct
determinant
df2ef
df2fi
df2mf
df2st
dflist
dfRange
diag
diagonal
diagonal?
diagonalMatrix
diagonalProduct
diagonals
dictionary
diff
difference
differentialVariables

```
differentiate
digamma
digit
digit?
digits
dihedral
dihedralGroup
dilog
dim
dimension
dimensionOfIrreducibleRepresentation
dimensions
dimensionsOf
diophantineSystem
dioSolve
direction
directory
directProduct
directSum
discreteLog
discriminant
discriminantEuclidean
display
dispose!
distance
distdfact
distFact
distribute
div
divergence
divide
divideExponents
divideIfCan
divideIfCan!
divisor
divisorCascade
divisors
dmp2rfi
dmpToHdmp
dmpToP
dn
dom
domainOf
dominantTerm
dot
double
double?
doubleComplex?
doubleDisc
doubleRank
```

doubleResultant
doublyTransitive?
draw
drawComplex
drawComplexVectorField
drawCurves
drawStyle
drawToScale
droot
duplicates
duplicates?
e
e01baf
e01bef
e01bff
e01bgf
e01bhf
e01daf
e01saf
e01sbf
e01sef
e01sff
e02adf
e02aef
e02agf
e02ahf
e02ajf
e02akf
e02baf
e02bbf
e02bcf
e02bdf
e02bef
e02daf
e02dcf
e02ddf
e02def
e02dff
e02gaf
e02zaf
e04dgf
e04fdf
e04gcf
e04jaf
e04mbf
e04naf
e04ucf
e04ycf
edf2df
edf2ef

```
edf2efi
edf2fi
ef2edf
Ei
eigenMatrix
eigenvalues
eigenvector
eigenvectors
eisensteinIrreducible?
elColumn2!
elem?
element?
elementary
elements
elliptic
elliptic?
ellipticCylindrical
elRow1!
elRow2!
elt
empty
empty?
endOfFile?
endSubProgram
enqueue!
enterInCache
enterPointData
entries
entry
entry?
enumerate
epilogue
EQ
eq
eq?
equality
equation
erf
error
errorInfo
errorKind
escape
euclideanGroebner
euclideanNormalForm
euclideanSize
euler
eulerE
eulerPhi
eval
evaluate
```

evaluateInverse
even?
evenInfiniteProduct
evenlambert
every?
exactQuotient
exactQuotient!
exists?
exp
exp1
expand
expandLog
expandPower
expandTrigProducts
expenseOfEvaluation
expenseOfEvaluationIF
expextendedint
expIfCan
expint
expintegrate
expintfldpoly
explicitEntries?
explicitlyEmpty?
explicitlyFinite?
explimitedint
explogs2trigs
exponent
exponential
exponential1
exponentialOrder
exponents
expPot
expr
expressIdealMember
exprHasAlgebraicWeight
exprHasLogarithmicWeights
exprHasWeightCosWXorSinWX
exprToGenUPS
exprToUPS
exprToXXP
expt
exptMod
exQuo
exquo
extend
extendedEuclidean
extendedint
extendedIntegrate
extendedResultant
extendedSubResultantGcd

```
extendIfCan
extension
extensionDegree
exteriorDifferential
external?
externalList
extract!
extractBottom!
extractClosed
extractIfCan
extractIndex
extractPoint
extractProperty
extractSplittingLeaf
extractTop!
eyeDistance
F
f01brf
f01bsf
f01maf
f01mcf
f01qcf
f01qdf
f01qef
f01rcf
f01rdf
f01ref
f02aaf
f02abf
f02adf
f02aef
f02aff
f02agf
f02ajf
f02akf
f02awf
f02axf
f02bbf
f02bjf
f02fjf
f02wef
f02xef
f04adf
f04arf
f04asf
f04atf
f04axf
f04faf
f04jgf
f04maf
```

f04mbf
f04mcf
f04qaf
f07adf
f07aef
f07fdf
f07fef
f2df
F2FG
f2st
factor
factor1
factorAndSplit
factorByRecursion
factorFraction
factorGroebnerBasis
factorial
factorials
factorList
factorOfDegree
factorPolynomial
factors
factorset
factorSFBRlcUnit
factorsOfCyclicGroupSize
factorsOfDegree
factorSquareFree
factorSquareFreeByRecursion
factorSquareFreePolynomial
failed
failed?
false
ffactor
FG2F
fglmIfCan
fi2df
fibonacci
field
fields
figureUnits
filename
fill!
fillPascalTriangle
filterUntil
filterWhile
find
findCycle
finite?
finiteBasis
finiteBound

```
fintegrate
first
firstDenom
firstNumer
firstSubsetGray
firstUncouplingMatrix
fixedDivisor
fixedPoint
fixedPointExquo
fixedPoints
fixPredicate
flagFactor
flatten
flexible?
flexibleArray
float
float?
floatlist
floatlist?
floor
fmecg
forLoop
FormatArabic
FormatRoman
formula
fortran
fortranCarriageReturn
fortranCharacter
fortranCompilerName
fortranComplex
fortranDouble
fortranDoubleComplex
fortranInteger
fortranLinkerArgs
fortranLiteral
fortranLiteralLine
fortranLogical
fortranReal
fortranTypeOf
fprindINFO
fracPart
fractionFreeGauss!
fractionPart
fractRadix
fractRagits
freeOf?
Frobenius
frobenius
front
froot
```

frst
fTable
fullDisplay
fullPartialFraction
function
functionIsContinuousAtEndPoints
functionIsFracPolynomial?
functionIsOscillatory
Gamma
gbasis
gcd
gcdcofact
gcdcofactprim
gcdPolynomial
gcdprim
gcdPrimitive
gderiv
GE
generalInfiniteProduct
generalizedContinuumHypothesisAssumed
generalizedContinuumHypothesisAssumed?
generalizedEigenvector
generalizedEigenvectors
generalizedInverse
generalLambert
generalPosition
generalSqFr
generalTwoFactor
generate
generateIrredPoly
generator
generators
generic
generic?
genericLeftDiscriminant
genericLeftMinimalPolynomial
genericLeftNorm
genericLeftTrace
genericLeftTraceForm
genericPosition
genericRightDiscriminant
genericRightMinimalPolynomial
genericRightNorm
genericRightTrace
genericRightTraceForm
genus
geometric
getBadValues
getButtonValue
getCode

```
getCurve
getDatabase
getExplanations
getGoodPrime
getGraph
gethi
getlo
getMatch
getMeasure
getMultiplicationMatrix
getMultiplicationTable
getOrder
getPickedPoints
getRef
getStream
getVariableOrder
getZechTable
GF2FG
goodnessOfFit
goodPoint
GospersMethod
goto
gradient
graeffe
gramschmidt
graphCurves
graphImage
graphs
graphState
graphStates
green
groebgen
groebner
groebner?
groebnerFactorize
groebnerIdeal
groebSolve
ground
ground?
GT
halfExtendedResultant1
halfExtendedResultant2
halfExtendedSubResultantGcd1
halfExtendedSubResultantGcd2
harmonic
has?
hash
hasHi
hasoln
hasPredicate?
```

hasSolution?
hasTopPredicate?
Hausdorff
hclf
hconcat
hcrf
hdmpToDmp
hdmpToP
head
headReduce
headReduced?
headRemainder
heap
heapSort
height
henselFact
HenselLift
hermite
hermiteH
HermiteIntegrate
hessian
hex
hexDigit
hexDigit?
hi
high
highCommonTerms
hitherPlane
hMonic
HMS
homogeneous?
horizConcat
hspace
htrigs
hue
hyperelliptic
hypergeometric0F1
iCompose
id
ideal
idealiser
idealiserMatrix
idealSimplify
identification
identity
identityMatrix
identitySquareMatrix
iExquo
iflist2Result
iFTable

```
ignore?
iiabs
iiacos
iiacosh
iiacot
iiacoth
iiacsc
iiacsch
iiasec
iiasech
iiasin
iiasinh
iiatan
iiatanh
iibinom
iicos
iicosh
iicot
iicoth
iicsc
iicsch
iidprod
iidsum
iiexp
iifact
iiGamma
iilog
iiperm
iipow
iisec
iisech
iisin
iisinh
iisqrt2
iisqrt3
iitan
iitanh
imag
imagE
imagI
imagi
imaginary
imagJ
imagj
imagK
imagk
implies
in?
inc
incr
```

```
int?
intChoose
intcompBasis
integer
integer?
integerBound
integerIfCan
integers
integral
integral?
integralAtInfinity?
integralBasis
integralBasisAtInfinity
integralCoordinates
integralDerivationMatrix
integralLastSubResultant
integralMatrix
integralMatrixAtInfinity
integralRepresents
integrate
intensity
intermediateResultsIF
internal?
internalAugment
internalDecompose
internalInfRittWu?
internalIntegrate
internalIntegrate0
internalLastSubResultant
internalSubPolSet?
internalSubQuasiComponent?
internalZeroSetSplit
interpolate
interpret
interpretString
interReduce
intersect
interval
intlist
intlist?
intPatternMatch
inv
inverse
inverseColeman
inverseIntegralMatrix
inverseIntegralMatrixAtInfinity
inverseLaplace
invertible?
invertibleElseSplit?
invertibleSet
```

invertIfCan
invmod
invmultisect
invWrite
iomode
ipow
iprint
iroot
irreducible?
irreducibleFactor
irreducibleFactors
irreducibleRepresentation
Is
is?
isAbsolutelyIrreducible?
isExpt
isList
isMult
isobaric?
isOp
isPlus
isPower
isQuotient
isTimes
iter
iteratedInitials
jacobi
jacobian
jacobiIdentity?
janko2
jordanAdmissible?
jordanAlgebra?
karatsuba
karatsubaDivide
karatsubaOnce
kernel
kernels
key
key?
keys
kmax
knownInfBasis
kovacic
kroneckerDelta
KrullNumber
ksec
label
lagrange
LagrangeInterpolation
laguerre

```
laguerreL
lambda
lambert
laplace
laplacian
largest
last
lastSubResultant
lastSubResultantElseSplit
lastSubResultantEuclidean
latex
laurent
laurentIfCan
laurentRep
Lazard
Lazard2
LazardQuotient
LazardQuotient2
lazy?
lazyEvaluate
lazyGintegrate
lazyIntegrate
lazyIrreducibleFactors
lazyPquo
lazyPrem
lazyPremWithDefault
lazyPseudoDivide
lazyPseudoQuotient
lazyPseudoRemainder
lazyResidueClass
lazyVariations
lcm
ldf2lst
ldf2vmf
LE
leader
leadingBasisTerm
leadingCoefficient
leadingCoefficientRicDE
leadingExponent
leadingIdeal
leadingIndex
leadingMonomial
leadingSupport
leadingTerm
leaf?
leastAffineMultiple
leastMonomial
leastPower
leaves
```

left
leftAlternative?
leftCharacteristicPolynomial
leftDiscriminant
leftDivide
leftExactQuotient
leftExtendedGcd
leftFactor
leftFactorIfCan
leftGcd
leftLcm
leftMinimalPolynomial
leftMult
leftNorm
leftOne
leftPower
leftQuotient
leftRank
leftRankPolynomial
leftRecip
leftRegularRepresentation
leftRemainder
leftScalarTimes!
leftTrace
leftTraceMatrix
leftTrim
leftUnit
leftUnits
leftZero
legendre
legendreP
lend!
length
lepol
less?
level
leviCivitaSymbol
lex
lexGroebner
lexico
lexTriangular
lfextendedint
lfextlimint
lfinfieldint
lfintegrate
lflimitedint
lfunc
lhs
li
library

```
lieAdmissible?
lieAlgebra?
LiePoly
LiePolyIfCan
lift
lifting
lifting1
light
lighting
limit
limitedint
limitedIntegrate
limitPlus
linear
linear?
linearAssociatedExp
linearAssociatedLog
linearAssociatedOrder
linearDependence
linearDependenceOverZ
linearlyDependent?
linearlyDependentOverZ?
linearMatrix
linearPart
linearPolynomials
linears
lineColorDefault
linGenPos
linkToFortran
linSolve
lintgcd
list
list?
listBranches
listConjugateBases
listexp
listLoops
listOfLists
listOfMonoms
listOfTerms
listRepresentation
lists
listYoungTableaus
lllip
lllp
llprop
lo
localAbs
localIntegralBasis
localReal?
```

localUnquote
LODO2FUN
log
log10
log2
logGamma
logical?
logIfCan
logpart
lookup
loopPoints
low
lowerCase
lowerCase!
lowerCase?
lowerPolynomial
LowTriBddDenomInv
lp
lprop
lquo
lSpaceBasis
lstart!
LT
lyndon
lyndon?
LyndonBasis
LyndonCoordinates
lyndonIfCan
LyndonWordsList
LyndonWordsList1
magnitude
mainCharacterization
mainCoefficients
mainContent
mainDefiningPolynomial
mainForm
mainKernel
mainMonomial
mainMonomials
mainPrimitivePart
mainSquareFreePart
mainValue
mainVariable
mainVariable?
mainVariables
make
makeCos
makeCrit
makeEq
makeFloatFunction

```
makeFR
makeGraphImage
makeMulti
makeObject
makeop
makeprod
makeRecord
makeResult
makeSceneGraph
makeSeries
makeSin
makeSketch
makeSUP
makeTerm
makeUnit
makeVariable
makeViewport2D
makeViewport3D
makeYoungTableau
makingStats?
mantissa
map
map!
mapBivariate
mapCoef
mapdiv
mapDown!
mapExpon
mapExponents
mapGen
mapMatrixIfCan
mapmult
mapSolve
mapUnivariate
mapUnivariateIfCan
mapUp!
mask
mat
match
match?
mathieu11
mathieu12
mathieu22
mathieu23
mathieu24
matrix
matrixConcat3D
matrixDimensions
matrixGcd
max
```

maxColIndex
maxdeg
maximumExponent
maxIndex
maxint
maxPoints
maxPoints3D
maxrank
maxrow
maxRowIndex
mdeg
measure
measure2Result
meatAxe
medialSet
member?
members
merge
merge!
mergeDifference
mergeFactors
mesh
mesh?
meshFun2Var
meshPar1Var
meshPar2Var
message
messagePrint
middle
midpoint
midpoints
mightHaveRoots
min
minColIndex
mindeg
mindegTerm
minGbasis
minimalPolynomial
minimize
minimumDegree
minimumExponent
minIndex
minordet
minPoints
minPoints3D
minPol
minPoly
minrank
minRowIndex
minset

```
minus!
minusInfinity
mirror
mix
mkAnswer
mkcomm
mkIntegral
mkPrim
modifyPoint
modifyPointData
modTree
modularFactor
modularGcd
modularGcdPrimitive
module
moduleSum
moduloP
modulus
moebius
moebiusMu
monic?
monicCompleteDecompose
monicDecomposeIfCan
monicDivide
monicLeftDivide
monicModulo
monicRightDivide
monicRightFactorIfCan
monom
monomial
monomial?
monomialIntegrate
monomialIntPoly
monomials
monomRDE
monomRDEsys
more?
moreAlgebraic?
morphism
move
movedPoints
mpsode
mr
mulmod
multiEuclidean
multiEuclideanTree
multinomial
multiple
multiple?
multiplyCoefficients
```

multiplyExponents
multisect
multiset
multivariate
multMonom
musserTrials
mvar
myDegree
nagCosInt
nagDAiryAi
nagDAiryBi
nagDFT
nagEigenvalues
nagEigenvectors
nagEllipticIntegralRC
nagEllipticIntegralRD
nagEllipticIntegralRF
nagEllipticIntegralRJ
nagErf
nagErfC
nagExpInt
nagFresnelC
nagFresnelS
nagHankelH1
nagHankelH2
nagHermitianDFT
nagHermitianInverseDFT
nagIncompleteGammaP
nagIncompleteGammaQ
nagInverseDFT
nagKelvinBei
nagKelvinBer
nagKelvinKei
nagKelvinKer
nagMin
nagPolygonIntegrate
nagScaledDAiryAi
nagScaledDAiryBi
nagScaledHankelH1
nagScaledHankelH2
nagSinInt
name
nand
nary?
ncols
negative?
neglist
new
newLine
newReduc

```
newSubProgram
newTypeLists
next
nextColeman
nextIrreduciblePoly
nextItem
nextLatticePermutation
nextNormalPoly
nextNormalPrimitivePoly
nextPartition
nextPrime
nextPrimitiveNormalPoly
nextPrimitivePoly
nextsousResultant2
nextSublist
nextsubResultant2
nextSubsetGray
nil
nilFactor
nlde
node
node?
nodeOf?
nodes
noKaratsuba
noLinearFactor?
noncommutativeJordanAlgebra?
nonLinearPart
nonQsign
nonSingularModel
nor
norm
normal
normal?
normal01
normalDenom
normalDeriv
normalElement
normalForm
normalise
normalize
normalizeAtInfinity
normalized?
normalizedAssociate
normalizedDivide
normalizeIfCan
normDeriv2
normFactors
normInvertible?
NOT
```

```
Not
not
notelem
npcoef
nrows
nsqfree
nthCoef
nthExpon
nthExponent
nthFactor
nthFlag
nthFractionalTerm
nthr
nthRoot
nthRootIfCan
Nul
null
null?
nullary
nullary?
nullity
nullSpace
number?
numberOfChildren
numberOfComponents
numberOfComposites
numberOfComputedEntries
numberOfCycles
numberOfDivisors
numberOfFactors
numberOfFractionalTerms
numberOfHues
numberOfImproperPartitions
numberOfIrreduciblePoly
numberOfMonomials
numberOfNormalPoly
numberOfOperations
numberOfPrimitivePoly
numberOfVariables
numer
numerator
numerators
numeric
numericalIntegration
numericalOptimization
numericIfCan
numFunEvals
numFunEvals3D
obj
objectOf
```

```
objects
oblateSpheroidal
ocf2ocdf
octon
odd?
oddInfiniteProduct
oddintegers
oddlambert
ode
ode1
ode2
ODESolve
OMbindTCP
OMclose
OMcloseConn
OMconnectTCP
OMconnInDevice
OMconnOutDevice
OMencodingBinary
OMencodingSGML
OMencodingUnknown
OMencodingXML
omError
OMgetApp
OMgetAtp
OMgetAttr
OMgetBind
OMgetBVar
OMgetEndApp
OMgetEndAtp
OMgetEndAttr
OMgetEndBind
OMgetEndBVar
OMgetEndError
OMgetEndObject
OMgetError
OMgetFloat
OMgetInteger
OMgetObject
OMgetString
OMgetSymbol
OMgetType
OMgetVariable
OMlistCDs
OMlistSymbols
OMmakeConn
OMopenFile
OMopenString
OMParseError?
OMputApp
```

OMputAtp
OMputAttr
OMputBind
OMputBVar
OMputEndApp
OMputEndAtp
OMputEndAttr
OMputEndBind
OMputEndBVar
OMputEndError
OMputEndObject
OMputError
OMputFloat
OMputInteger
OMputObject
OMputString
OMputSymbol
OMputVariable
OMread
OMReadError?
OMreadFile
OMreadStr
OMreceive
OMsend
OMserve
OMsetEncoding
OMsupportsCD?
OMsupportsSymbol?
OMunhandledSymbol
OMUnknownCD?
OMUnknownSymbol?
OMwrite
one?
oneDimensionalArray
op
open
open?
operation
operator
operators
opeval
optAttributes
optimize
option
option?
optional
optional?
options
optpair
OR

```
Or
or
orbit
orbits
ord
order
orthonormalBasis
outerProduct
outlineRender
output
outputArgs
outputAsFortran
outputAsScript
outputAsTex
outputFixed
outputFloating
outputForm
outputGeneral
outputList
outputMeasure
outputSpacing
over
overbar
overlabel
overlap
overset?
pack!
packageCall
packHS
pade
padecf
padicallyExpand
padicFraction
pair?
palgextint
palgextint0
palginfieldint
palgint
palgint0
palgintegrate
palglimint
palglimint0
palgLODE
palgLODE0
palgRDE
palgRDE0
parabolic
parabolicCylindrical
paraboloidal
parametersOf
```

parametric?
ParCond
ParCondList
paren
parent
partialDenominators
partialFraction
partialNumerators
partialQuotients
particularSolution
partition
partitions
parts
pascalTriangle
pastel
pattern
patternMatch
patternMatchTimes
patternVariable
pdct
PDESolve
pdf2df
pdf2ef
perfectNthPower?
perfectNthRoot
perfectSqrt
perfectSquare?
permanent
permutation
permutationGroup
permutationRepresentation
permutations
perspective
phiCoord
pHS
physicalLength
physicalLength!
pi
pile
plenaryPower
pleskenSplit
plot
plotPolar
plus
plus!
plusInfinity
pmComplexintegrate
pmintegrate
po
point

```
point?
pointColor
pointColorDefault
pointColorPalette
pointData
pointlist
pointlist?
pointLists
pointPlot
points
pointSizeDefault
poisson
pol
polar
polarCoordinates
polCase
pole?
PollardSmallFactor
polygamma
polygon
polygon?
polynomial
polynomialZeros
polyPart
polyRDE
polyred
polyRicDE
pomopo!
pop!
popFortranOutputStack
position
position!
positive?
positiveRemainder
positiveSolve
possiblyInfinite?
possiblyNewVariety?
postfix
pow
power
power!
powerAssociative?
powern
powers
powerSum
powmod
pquo
pr2dmp
precision
predicate
```

predicates
prefix
prefix?
prefixRagits
prem
prepareDecompose
prepareSubResAlgo
preprocess
presub
presuper
previous
prevPrime
primaryDecomp
prime
prime?
primeFactor
primeFrobenius
primes
primextendedint
primextintfrac
primintegrate
primintfldpoly
primitive?
primitiveElement
primitiveMonomials
primitivePart
primitivePart!
primlimintfrac
primlimitedint
primPartElseUnitCanonical
primPartElseUnitCanonical!
prinb
principal?
principalIdeal
prindINFO
prinpolINFO
prinshINFO
print
printCode
printHeader
printInfo
printInfo!
printingInfo?
printStatement
printStats!
printTypes
probablyZeroDim?
problemPoints
processTemplate
prod

```
product
prolateSpheroidal
prologue
properties
property
pseudoDivide
pseudoQuotient
pseudoRemainder
psolve
ptFunc
pToDmp
pToHdmp
ptree
puiseux
pureLex
purelyAlgebraic?
purelyAlgebraicLeadingMonomial?
purelyTranscendental?
push!
pushdown
pushdterm
pushFortranOutputStack
pushucoef
pushuconst
pushup
put!
putColorInfo
putGraph
qelt
qfactor
qinterval
qPot
qqq
qroot
qsetelt!
quadratic
quadratic?
quadraticForm
quadraticNorm
quartic
quasiAlgebraicSet
quasiComponent
quasiMonic?
quasiMonicPolynomials
quasiRegular
quasiRegular?
quatern
queue
quickSort
quickWrite
```

quo
quoByVar
quote
quoted?
quotedOperators
quotient
quotientByP
radical
radicalEigenvalues
radicalEigenvector
radicalEigenvectors
radicalOfLeftTraceForm
radicalRoots
radicalSimplify
radicalSolve
radix
radPoly
raisePolynomial
ramified?
ramifiedAtInfinity?
ran
randnum
random
randomLC
randomR
range
rangeIsFinite
rangePascalTriangle
ranges
rank
rarrow
ratDenom
ratDsolve
rational
rational?
rationalApproximation
rationalFunction
rationalIfCan
rationalPoint?
rationalPoints
rationalPower
ratpart
ratPoly
ravel
rCoord
rdHack1
rdregime
read
read!
readable?

```
readIfCan!
readLine!
readLineIfCan!
real
real?
realEigenvalues
realEigenvectors
realElementary
realRoots
realSolve
realZeros
recip
reciprocalPolynomial
recolor
recoverAfterFail
rectangularMatrix
recur
red
redmat
redPo
redPol
redpps
reduce
reduceBasisAtInfinity
reduceByQuasiMonic
reduced?
reducedContinuedFraction
reducedDiscriminant
reducedForm
reducedQPowers
reducedSystem
reduceLODE
ReduceOrder
reduction
reductum
ref
refine
regime
region
regularRepresentation
reindex
relationsIdeal
relativeApprox
relerror
rem
remainder
RemainderList
remove
remove!
removeConstantTerm
```

removeCoshSq
removeCosSq
removeDuplicates
removeDuplicates!
removeIrreducibleRedundantFactors
removeRedundantFactors
removeRedundantFactorsInContents
removeRedundantFactorsInPols
removeRoughlyRedundantFactorsInContents
removeRoughlyRedundantFactorsInPol
removeRoughlyRedundantFactorsInPols
removeSinhSq
removeSinSq
removeSquaresIfCan
removeSuperfluousCases
removeSuperfluousQuasiComponents
removeZero
removeZeroes
rename
rename!
render
renderToFile!
reopen!
reorder
repeating
repeating?
repeatUntilLoop
replace
replaceKthElement
representationType
represents
repSq
reseed
reset
reset!
resetAttributeButtons
resetBadValues
resetNew
resetVariableOrder
reshape
resize
rest
restorePrecision
result
resultant
resultantEuclidean
resultantEuclideannaif
resultantnaif
resultantReduit
resultantReduitEuclidean

```
retract
retractable?
retractIfCan
returns
returnType!
returnTypeOf
reverse
reverse!
reverseLex
revert
rewriteIdealWithHeadRemainder
rewriteIdealWithQuasiMonicGenerators
rewriteIdealWithRemainder
rewriteSetByReducingWithParticularGenerators
rewriteSetWithReduction
RF2UTS
rhs
ricDsolve
ridHack1
right
rightAlternative?
rightCharacteristicPolynomial
rightDiscriminant
rightDivide
rightExactQuotient
rightExtendedGcd
rightFactorCandidate
rightFactorIfCan
rightGcd
rightLcm
rightMinimalPolynomial
rightMult
rightNorm
rightOne
rightPower
rightQuotient
rightRank
rightRankPolynomial
rightRecip
rightRegularRepresentation
rightRemainder
rightScalarTimes!
rightTrace
rightTraceMatrix
rightTrim
rightUnit
rightUnits
rightZero
rischDE
rischDEsys
```

```
s13aaf
s13acf
s13adf
s14aaf
s14abf
s14baf
s15adf
s15aef
s17acf
s17adf
s17aef
s17aff
s17agf
s17ahf
s17ajf
s17akf
s17dcf
s17def
s17dgf
s17dhf
s17dlf
s18acf
s18adf
s18aef
s18aff
s18dcf
s18def
s19aaf
s19abf
s19acf
s19adf
s20acf
s20adf
s21baf
s21bbf
s21bcf
s21bdf
safeCeiling
safeFloor
safetyMargin
sample
satisfy?
saturate
save
say
sayLength
scalarMatrix
scalarTypeOf
scale
scaleRoots
```

scan
ScanArabic
ScanFloatIgnoreSpaces
ScanFloatIgnoreSpacesIfCan
scanOneDimSubspaces
ScanRoman
schema
schwerpunkt
screenResolution
screenResolution3D
script
scripted?
scripts
sdf2lst
se2rfi
search
sec
sec2cos
sech
sech2cosh
sechIfCan
secIfCan
second
seed
SEGMENT
segment
select
select!
selectAndPolynomials
selectFiniteRoutines
selectfirst
selectIntegrationRoutines
selectMultiDimensionalRoutines
selectNonFiniteRoutines
selectODEIVPRoutines
selectOptimizationRoutines
selectOrPolynomials
selectPDERoutines
selectPolynomials
selectsecond
selectSumOfSquaresRoutines
semicolonSeparate
semiDegreeSubResultantEuclidean
semiDiscriminantEuclidean
semiIndiceSubResultantEuclidean
semiLastSubResultantEuclidean
semiResultantEuclidean1
semiResultantEuclidean2
semiResultantEuclideannaif
semiResultantReduitEuclidean

```
semiSubResultantGcdEuclidean1
semiSubResultantGcdEuclidean2
separant
separate
separateDegrees
separateFactors
sequences
series
seriesSolve
seriesToOutputForm
set
setAdaptive
setAdaptive3D
setAttributeButtonStep
setButtonValue
setchildren!
setClipValue
setClosed
setColumn!
setCondition!
setDifference
setelt
setelt!
setEmpty!
setEpilogue!
setErrorBound
setFieldInfo
setfirst!
setFormula!
setImagSteps
setIntersection
setLabelValue
setlast!
setleaves!
setleft!
setLegalFortranSourceExtensions
setMaxPoints
setMaxPoints3D
setMinPoints
setMinPoints3D
setnext!
setOfMinN
setOrder
setPoly
setPosition
setPredicates
setprevious!
setPrologue!
setProperties
setProperty
```

setRealSteps
setref
setrest!
setright!
setRow!
setScreenResolution
setScreenResolution3D
setStatus
setStatus!
setsubMatrix!
setTex!
setTopPredicate
setUnion
setValue!
setvalue!
setVariableOrder
SFunction
sh
shade
shallowCopy
shallowExpand
shanksDiscLogAlgorithm
shellSort
shift
shiftLeft
shiftRight
shiftRoots
show
showAll?
showAllElements
showArrayValues
showAttributes
showClipRegion
showFortranOutputStack
showIntensityFunctions
showRegion
showScalarValues
showTheFTable
showTheIFTable
showTheRoutinesTable
showTheSymbolTable
showTypeInOutput
shrinkable
shuffle
shufflein
Si
sign
signAround
simpleBounds?
simplify

```
simplifyExp
simplifyLog
simplifyPower
simpson
simpsono
sin
sin?
sin2csc
sincos
singleFactorBound
singRicDE
singular?
singularAtInfinity?
singularitiesOf
sinh
sinh2csch
sinhcosh
sinhIfCan
sinIfCan
size
size?
sizeLess?
sizeMultiplication
sizePascalTriangle
skewSFunction
slash
slex
smith
sn
sncndn
socf2socdf
solid
solid?
solve
solve1
solveid
solveInField
solveLinear
solveLinearlyOverQ
solveLinearPolynomialEquation
solveLinearPolynomialEquationByFractions
solveLinearPolynomialEquationByRecursion
solveRetract
someBasis
sort
sort!
sortConstraints
sorted?
space
sparsityIF
```

specialTrigs
spherical
split
split!
splitConstant
splitDenominator
splitLinear
splitNodeOf!
splitSquarefree
sPol
sqfree
sqfrFactor
sqrt
square?
squareFree
squareFreeFactors
squareFreeLexTriangular
squareFreePart
squareFreePolynomial
squareFreePrim
squareMatrix
squareTop
stack
standardBasisOfCyclicSubmodule
start!
startPolynomial
startStats!
startTable!
startTableGcd!
startTableInvSet!
status
stFunc1
stFunc2
stFuncN
stiffnessAndStabilityFactor
stiffnessAndStabilityOfODEIF
stirling1
stirling2
stop
stop!
stopMusserTrials
stopTable!
stopTableGcd!
stopTableInvSet!
stoseIntegralLastSubResultant
stoseInternalLastSubResultant
stoseInvertible?
stoseInvertible?reg
stoseInvertibleSet
stoseInvertibleSetreg

```
stoseInvertibleSetsqfreg
stoseInvertible?sqfreg
stoseLastSubResultant
stosePrepareSubResAlgo
stoseSquareFreePart
string
string?
stripCommentsAndBlanks
strongGenerators
stronglyReduce
stronglyReduced?
structuralConstants
sts2stst
SturmHabicht
SturmHabichtCoefficients
SturmHabichtMultiple
SturmHabichtSequence
sturmSequence
sturmVariationsOf
style
sub
subCase?
subHeight
subMatrix
submod
subNode?
subNodeOf?
subPolSet?
subQuasiComponent?
subResultantChain
subResultantGcd
subResultantGcdEuclidean
subResultantsChain
subresultantSequence
subresultantVector
subscript
subscriptedVariables
subSet
subset?
subspace
subst
substitute
substring?
subtractIfCan
subTriSet?
suchThat
suffix?
sum
summation
sumOfDivisors
```

sumOfKthPowerDivisors
sumOfSquares
sumSquares
sup
supDimElseRittWu?
super
superHeight
superscript
supersub
supRittWu?
surface
swap
swap!
swapColumns!
swapRows!
sylvesterMatrix
sylvesterSequence
symbol
symbol?
symbolIfCan
symbolTable
symbolTableOf
symFunc
symmetric?
symmetricDifference
symmetricGroup
symmetricPower
symmetricProduct
symmetricRemainder
symmetricSquare
symmetricTensors
systemCommand
systemSizeIF
t
tab
tab1
table
tableau
tableForDiscreteLogarithm
tablePow
tail
tan
tan2cot
tan2trig
tanAn
tanh
tanh2coth
tanh2trigh
tanhIfCan
tanIfCan

```
tanintegrate
tanNa
tanQ
tanSum
taylor
taylorIfCan
taylorQuoByVar
taylorRep
tensorProduct
terms
test
testDim
testModulus
tex
thetaCoord
third
timer
times
times!
title
top
top!
topFortranOutputStack
topPredicate
toroidal
torsion?
torsionIfCan
toScale
toseInvertible?
toseInvertibleSet
toseLastSubResultant
toseSquareFreePart
totalDegree
totalDifferential
totalfract
totalGroebner
totalLex
totolex
tower
trace
trace2PowMod
traceMatrix
tracePowMod
trailingCoefficient
tRange
transcendenceDegree
transcendent?
transcendentalDecompose
transform
translate
```

transpose
trapezoidal
trapezoidalo
traverse
tree
triangSolve
triangular?
triangularSystems
triangulate
trigs
trigs2explogs
trim
trivialIdeal?
true
trueEqual
trunc
truncate
tryFunctionalDecomposition
tryFunctionalDecomposition?
tube
tubePlot
tubePoints
tubePointsDefault
tubeRadius
tubeRadiusDefault
tValues
twist
twoFactor
typeList
typeLists
unary?
unaryFunction
uncouplingMatrices
unexpand
uniform
uniform01
union
uniqueID
unit
unit?
unitCanonical
unitNormal
unitNormalize
units
unitsColorDefault
unitVector
univariate
univariate?
univariatePolynomial
univariatePolynomials

```
univariatePolynomialsGcds
univariateSolve
univcase
universe
unmakeSUP
unparse
unprotectedRemoveRedundantFactors
unrankImproperPartitions0
unrankImproperPartitions1
unravel
untab
UnVectorise
unvectorise
UP2ifCan
UP2UTS
updatD
update
upDateBranches
updateStatus!
updatF
upperCase
upperCase!
upperCase?
UpTriBddDenomInv
useEisensteinCriterion
useEisensteinCriterion?
useNagFunctions
userOrdered?
useSingleFactorBound
useSingleFactorBound?
usingTable?
UTS2UP
validExponential
value
var1Steps
var1StepsDefault
var2Steps
var2StepsDefault
variable
variables
variationOfParameters
vark
varList
varselect
vconcat
vector
Vectorise
vectorise
vedf2vef
vertConcat
```

viewDefaults
viewDeltaXDefault
viewDeltaYDefault
viewPhiDefault
viewpoint
viewport2D
viewport3D
viewPosDefault
viewSizeDefault
viewThetaDefault
viewWriteAvailable
viewWriteDefault
viewZoomDefault
virtualDegree
void
vput!
vspace
vstart!
walkTree
weakBiRank
weierstrass
weight
weighted
weights
whatInfinity
whileLoop
wholePart
wholeRadix
wholeRagits
width
withPredicates
wordInGenerators
wordInStrongGenerators
wordsForStrongGenerators
wreath
writable?
write
write!
writeLine!
wronskianMatrix
wrregime
xCoord
xn
xor
xRange
Y
yCoord
yCoordinates
yellow
youngGroup

```
yRange
zag
zCoord
zero
zero?
zeroDim?
zeroDimensional?
zeroDimPrimary?
zeroDimPrime?
zeroMatrix
zeroOf
zeroSetSplit
zeroSetSplitIntoTriangularSystems
zerosOf
zeroSquareMatrix
zeroVector
zoom
zRange
AbelianGroup
AbelianMonoid
AbelianMonoidRing
AbelianSemiGroup
Aggregate
Algebra
AlgebraicallyClosedField
AlgebraicallyClosedFunctionSpace
ArcHyperbolicFunctionCategory
ArcTrigonometricFunctionCategory
AssociationListAggregate
AttributeRegistry
BagAggregate
BasicType
BiModule
BinaryRecursiveAggregate
BinaryTreeCategory
BitAggregate
CachableSet
CancellationAbelianMonoid
CharacteristicNonZero
CharacteristicZero
CoercibleTo
Collection
CombinatorialFunctionCategory
CombinatorialOpsCategory
CommutativeRing
ComplexCategory
ConvertibleTo
DequeueAggregate
Dictionary
DictionaryOperations
```

DifferentialExtension
DifferentialPolynomialCategory
DifferentialRing
DifferentialVariableCategory
DirectProductCategory
DivisionRing
DoublyLinkedAggregate
ElementaryFunctionCategory
Eltable
EltableAggregate
EntireRing
EuclideanDomain
Evalable
ExpressionSpace
ExtensibleLinearAggregate
ExtensionField
Field
FieldOfPrimeCharacteristic
FileCategory
FileNameCategory
Finite
FiniteAbelianMonoidRing
FiniteAlgebraicExtensionField
FiniteDivisorCategory
FiniteFieldCategory
FiniteLinearAggregate
FiniteRankAlgebra
FiniteRankNonAssociativeAlgebra
FiniteSetAggregate
FloatingPointSystem
FortranFunctionCategory
FortranMachineTypeCategory
FortranMatrixCategory
FortranMatrixFunctionCategory
FortranProgramCategory
FortranVectorCategory
FortranVectorFunctionCategory
FramedAlgebra
FramedNonAssociativeAlgebra
FreeAbelianMonoidCategory
FreeLieAlgebra
FreeModuleCat
FullyEvalableOver
FullyLinearlyExplicitRingOver
FullyPatternMatchable
FullyRetractableTo
FunctionFieldCategory
FunctionSpace
GcdDomain
GradedAlgebra

GradedModule
Group
HomogeneousAggregate
HyperbolicFunctionCategory
IndexedAggregate
IndexedDirectProductCategory
InnerEvalable
IntegerNumberSystem
IntegralDomain
IntervalCategory
IVLeafNodeCategory
IVNodeCategory
KeyedDictionary
LazyStreamAggregate
LeftAlgebra
LeftModule
LieAlgebra
LinearAggregate
LinearlyExplicitRingOver
LinearOrdinaryDifferentialOperatorCategory
LiouvillianFunctionCategory
ListAggregate
Logic
MatrixCategory
Module
Monad
MonadWithUnit
MonogenicAlgebra
MonogenicLinearOperator
Monoid
MultiDictionary
MultisetAggregate
MultivariateTaylorSeriesCategory
NonAssociativeAlgebra
NonAssociativeRing
NonAssociativeRng
NormalizedTriangularSetCategory
NumericalIntegrationCategory
NumericalOptimizationCategory
OctonionCategory
OneDimensionalArrayAggregate
OpenMath
OrderedAbelianGroup
OrderedAbelianMonoid
OrderedAbelianMonoidSup
OrderedAbelianSemiGroup
OrderedCancellationAbelianMonoid
OrderedFinite
OrderedIntegralDomain
OrderedMonoid

OrderedMultisetAggregate
OrderedRing
OrderedSet
OrdinaryDifferentialEquationsSolverCategory
PAdicIntegerCategory
PartialDifferentialEquationsSolverCategory
PartialDifferentialRing
PartialTranscendentalFunctions
Patternable
PatternMatchable
PermutationCategory
PlottablePlaneCurveCategory
PlottableSpaceCurveCategory
PointCategory
PolynomialCategory
PolynomialFactorizationExplicit
PolynomialSetCategory
PowerSeriesCategory
PrimitiveFunctionCategory
PrincipalIdealDomain
PriorityQueueAggregate
QuaternionCategory
QueueAggregate
QuotientFieldCategory
RadicalCategory
RealClosedField
RealConstant
RealNumberSystem
RealRootCharacterizationCategory
RectangularMatrixCategory
RecursiveAggregate
RecursivePolynomialCategory
RegularTriangularSetCategory
RetractableTo
RightModule
Ring
Rng
SegmentCategory
SegmentExpansionCategory
SemiGroup
SetAggregate
SetCategory
SExpressionCategory
SpecialFunctionCategory
SquareFreeNormalizedTriangularSetCategory
SquareFreeRegularTriangularSetCategory
SquareMatrixCategory
StackAggregate
StepThrough
StreamAggregate

```
StringAggregate
StringCategory
TableAggregate
ThreeSpaceCategory
TranscendentalFunctionCategory
TriangularSetCategory
TrigonometricFunctionCategory
TwoDimensionalArrayCategory
Type
UnaryRecursiveAggregate
UniqueFactorizationDomain
UnivariateLaurentSeriesCategory
UnivariateLaurentSeriesConstructorCategory
UnivariatePolynomialCategory
UnivariatePowerSeriesCategory
UnivariatePuiseuxSeriesCategory
UnivariatePuiseuxSeriesConstructorCategory
UnivariateSkewPolynomialCategory
UnivariateTaylorSeriesCategory
VectorCategory
VectorSpace
XAlgebra
XFreeAlgebra
XPolynomialsCat
AlgebraGivenByStructuralConstants
AlgebraicFunctionField
AlgebraicNumber
AnonymousFunction
AntiSymm
Any
ArrayStack
Asp1
Asp10
Asp12
Asp19
Asp20
Asp24
Asp27
Asp28
Asp29
Asp30
Asp31
Asp33
Asp34
Asp35
Asp4
Asp41
Asp42
Asp49
Asp50
```

Asp55
Asp6
Asp7
Asp73
Asp74
Asp77
Asp78
Asp8
Asp80
Asp9
AssociatedJordanAlgebra
AssociatedLieAlgebra
AssociationList
AttributeButtons
Automorphism
BalancedBinaryTree
BalancedPAdicInteger
BalancedPAdicRational
BasicFunctions
BasicOperator
BinaryExpansion
BinaryFile
BinarySearchTree
BinaryTournament
BinaryTree
Bits
Boolean
CardinalNumber
CartesianTensor
Character
CharacterClass
CliffordAlgebra
Color
Commutator
Complex
ContinuedFraction
d01ajfAnnaType
d01akfAnnaType
d01alfAnnaType
d01amfAnnaType
d01anfAnnaType
d01apfAnnaType
d01aqfAnnaType
d01asfAnnaType
d01fcfAnnaType
d01gbfAnnaType
d01TransformFunctionType
d02bbfAnnaType
d02bhfAnnaType
d02cjfAnnaType

```
d02ejfAnnaType
d03eefAnnaType
d03fafAnnaType
Database
DataList
DecimalExpansion
DenavitHartenbergMatrix
Dequeue
DeRhamComplex
DifferentialSparseMultivariatePolynomial
DirectProduct
DirectProductMatrixModule
DirectProductModule
DistributedMultivariatePolynomial
DoubleFloat
DrawOption
e04dgfAnnaType
e04fdfAnnaType
e04gcfAnnaType
e04jafAnnaType
e04mbfAnnaType
e04nafAnnaType
e04ucfAnnaType
ElementaryFunctionsUnivariateLaurentSeries
ElementaryFunctionsUnivariatePuiseuxSeries
Enumeration
EqTable
Equation
EuclideanModularRing
Exit
ExponentialExpansion
ExponentialOfUnivariatePuiseuxSeries
Expression
ExtAlgBasis
Factored
File
FileName
FiniteDivisor
FiniteField
FiniteFieldCyclicGroup
FiniteFieldCyclicGroupExtension
FiniteFieldCyclicGroupExtensionByPolynomial
FiniteFieldExtension
FiniteFieldExtensionByPolynomial
FiniteFieldNormalBasis
FiniteFieldNormalBasisExtension
FiniteFieldNormalBasisExtensionByPolynomial
FlexibleArray
Float
FormalFraction
```

FortranCode
FortranExpression
FortranProgram
FortranScalarType
FortranTemplate
FortranType
FourierComponent
FourierSeries
Fraction
FractionalIdeal
FramedModule
FreeAbelianGroup
FreeAbelianMonoid
FreeGroup
FreeModule
FreeModule1
FreeMonoid
FreeNilpotentLie
FullPartialFractionExpansion
FunctionCalled
GeneralDistributedMultivariatePolynomial
GeneralModulePolynomial
GeneralPolynomialSet
GeneralSparseTable
GeneralTriangularSet
GeneralUnivariatePowerSeries
GenericNonAssociativeAlgebra
GraphImage
HashTable
Heap
HexadecimalExpansion
HomogeneousDirectProduct
HomogeneousDistributedMultivariatePolynomial
HyperellipticFiniteDivisor
IndexCard
IndexedBits
IndexedDirectProductAbelianGroup
IndexedDirectProductAbelianMonoid
IndexedDirectProductObject
IndexedDirectProductOrderedAbelianMonoid
IndexedDirectProductOrderedAbelianMonoidSup
IndexedExponents
IndexedFlexibleArray
IndexedList
IndexedMatrix
IndexedOneDimensionalArray
IndexedString
IndexedTwoDimensionalArray
IndexedVector
InfiniteTuple

```
InnerAlgebraicNumber
InnerFiniteField
InnerFreeAbelianMonoid
InnerIndexedTwoDimensionalArray
InnerPAdicInteger
InnerPrimeField
InnerSparseUnivariatePowerSeries
InnerTable
InnerTaylorSeries
InputForm
Integer
IntegerMod
IntegrationFunctionsTable
IntegrationResult
Interval
InventorDataSink
InventorRenderPackage
InventorViewPort
IVBaseColor
IVBasicNode
IVCoordinate3
IVCoordinate4
IVFaceSet
IVField
IVGroup
IVIndexedLineSet
IVNodeConnection
IVNodeObject
IVPointSet
IVQuadMesh
IVSeparator
IVSimpleInnerNode
IVUtilities
IVValue
Kernel
KeyedAccessFile
LaurentPolynomial
Library
LieExponentials
LiePolynomial
LieSquareMatrix
LinearOrdinaryDifferentialOperator
LinearOrdinaryDifferentialOperator1
LinearOrdinaryDifferentialOperator2
List
ListMonoidOps
ListMultiDictionary
LocalAlgebra
Localize
LyndonWord
```

MachineComplex
MachineFloat
MachineInteger
Magma
MakeCachableSet
Mapping
Matrix
ModMonic
ModularField
ModularRing
ModuleMonomial
ModuleOperator
MoebiusTransform
MonoidRing
Multiset
MultivariatePolynomial
NagDiscreteFourierTransformInterfacePackage
NagEigenInterfacePackage
NagOptimisationInterfacePackage
NagQuadratureInterfacePackage
NagResultChecks
NagSpecialFunctionsInterfacePackage
NewSparseMultivariatePolynomial
NewSparseUnivariatePolynomial
None
NonNegativeInteger
NumericalIntegrationProblem
NumericalODEProblem
NumericalOptimizationProblem
NumericalPDEProblem
Octonion
ODEIntensityFunctionsTable
OneDimensionalArray
OnePointCompletion
OpenMathConnection
OpenMathDevice
OpenMathEncoding
OpenMathError
OpenMathErrorKind
Operator
OppositeMonogenicLinearOperator
OrderedCompletion
OrderedDirectProduct
OrderedFreeMonoid
OrderedVariableList
OrderlyDifferentialPolynomial
OrderlyDifferentialVariable
OrdinaryDifferentialRing
OrdinaryWeightedPolynomials
OrdSetInts

```
OutputForm
PackedHermitianSequence
PAdicInteger
PAdicRational
PAdicRationalConstructor
Palette
ParametricPlaneCurve
ParametricSpaceCurve
ParametricSurface
PartialFraction
Partition
Pattern
PatternMatchListResult
PatternMatchResult
PendantTree
Permutation
PermutationGroup
Pi
PlaneAlgebraicCurvePlot
Plot
Plot3D
PoincareBirkhoffWittLyndonBasis
Point
Polynomial
PolynomialIdeals
PolynomialRing
PositiveInteger
PrimeField
PrimitiveArray
Product
QuadraticForm
QuasiAlgebraicSet
Quaternion
QueryEquation
Queue
RadicalFunctionField
RadixExpansion
RealClosure
Record
RectangularMatrix
Reference
RegularChain
RegularTriangularSet
RenderTools
ResidueRing
Result
RewriteRule
RightOpenIntervalRootCharacterization
RomanNumeral
RoutinesTable
```

RuleCalled
Ruleset
ScriptFormulaFormat
Segment
SegmentBinding
SequentialDifferentialPolynomial
SequentialDifferentialVariable
Set
SetOfMIntegersInOneToN
SExpression
SExpressionOf
SimpleAlgebraicExtension
SimpleFortranProgram
SingleInteger
SingletonAsOrderedSet
SparseMultivariatePolynomial
SparseMultivariateTaylorSeries
SparseTable
SparseUnivariateLaurentSeries
SparseUnivariatePolynomial
SparseUnivariatePuiseuxSeries
SparseUnivariateSkewPolynomial
SparseUnivariateTaylorSeries
SplitHomogeneousDirectProduct
SplittingNode
SplittingTree
SquareFreeRegularTriangularSet
SquareMatrix
Stack
Stream
String
StringTable
SubSpace
SubSpaceComponentProperty
SuchThat
Switch
Symbol
SymbolTable
SymmetricPolynomial
Table
Tableau
TaylorSeries
TexFormat
TextFile
TheSymbolTable
ThreeDimensionalMatrix
ThreeDimensionalViewport
ThreeSpace
Timer
Tree

```
TubePlot
Tuple
TwoDimensionalArray
TwoDimensionalViewport
Union
UnivariateLaurentSeries
UnivariateLaurentSeriesConstructor
UnivariatePolynomial
UnivariatePuiseuxSeries
UnivariatePuiseuxSeriesConstructor
UnivariatePuiseuxSeriesWithExponentialSingularity
UnivariateSkewPolynomial
UnivariateTaylorSeries
UniversalSegment
Variable
Vector
Void
WeightedPolynomials
WuWenTsunTriangularSet
XDistributedPolynomial
XPBWPolynomial
XPolynomial
XPolynomialRing
XRecursivePolynomial
AlgebraicFunction
AlgebraicHermiteIntegration
AlgebraicIntegrate
AlgebraicIntegration
AlgebraicManipulations
AlgebraicMultFact
AlgebraPackage
AlgFactor
AnnaNumericalIntegrationPackage
AnnaNumericalOptimizationPackage
AnnaOrdinaryDifferentialEquationPackage
AnnaPartialDifferentialEquationPackage
AnyFunctions1
ApplyRules
ApplyUnivariateSkewPolynomial
AssociatedEquations
AttachPredicates
BalancedFactorisation
BasicOperatorFunctions1
BezoutMatrix
BoundIntegerRoots
BrillhartTests
CartesianTensorFunctions2
ChangeOfVariable
CharacteristicPolynomialInMonogenicalAlgebra
CharacteristicPolynomialPackage
```

ChineseRemainderToolsForIntegralBases
CoerceVectorMatrixPackage
CombinatorialFunction
CommonDenominator
CommonOperators
CommuteUnivariatePolynomialCategory
ComplexFactorization
ComplexFunctions2
ComplexIntegerSolveLinearPolynomialEquation
ComplexPattern
ComplexPatternMatch
ComplexRootFindingPackage
ComplexRootPackage
ComplexTrigonometricManipulations
ConstantLODE
CoordinateSystems
CRApackage
CycleIndicators
CyclicStreamTools
CyclotomicPolynomialPackage
d01AgentsPackage
d01WeightsPackage
d02AgentsPackage
d03AgentsPackage
DefiniteIntegrationTools
DegreeReductionPackage
DiophantineSolutionPackage
DirectProductFunctions2
DiscreteLogarithmPackage
DisplayPackage
DistinctDegreeFactorize
DoubleFloatSpecialFunctions
DoubleResultantPackage
DrawComplex
DrawNumericHack
DrawOptionFunctions0
DrawOptionFunctions1
e04AgentsPackage
EigenPackage
ElementaryFunction
ElementaryFunctionDefiniteIntegration
ElementaryFunctionLODESolver
ElementaryFunctionODESolver
ElementaryFunctionSign
ElementaryFunctionStructurePackage
ElementaryIntegration
ElementaryRischDE
ElementaryRischDESystem
EllipticFunctionsUnivariateTaylorSeries
EquationFunctions2

```
ErrorFunctions
EuclideanGroebnerBasisPackage
EvaluateCycleIndicators
ExpertSystemContinuityPackage
ExpertSystemContinuityPackage1
ExpertSystemToolsPackage
ExpertSystemToolsPackage1
ExpertSystemToolsPackage2
ExpressionFunctions2
ExpressionSpaceFunctions1
ExpressionSpaceFunctions2
ExpressionSpaceODESolver
ExpressionToOpenMath
ExpressionToUnivariatePowerSeries
ExpressionTubePlot
FactoredFunctions
FactoredFunctions2
FactoredFunctionUtilities
FactoringUtilities
FGLMIfCanPackage
FindOrderFinite
FiniteDivisorFunctions2
FiniteFieldFunctions
FiniteFieldHomomorphisms
FiniteFieldPolynomialPackage
FiniteFieldPolynomialPackage2
FiniteFieldSolveLinearPolynomialEquation
FiniteLinearAggregateFunctions2
FiniteLinearAggregateSort
FiniteSetAggregateFunctions2
FloatingComplexPackage
FloatingRealPackage
FortranCodePackage1
FortranOutputStackPackage
FortranPackage
FractionalIdealFunctions2
FractionFunctions2
FramedNonAssociativeAlgebraFunctions2
FunctionalSpecialFunction
FunctionFieldCategoryFunctions2
FunctionFieldIntegralBasis
FunctionSpaceAssertions
FunctionSpaceAttachPredicates
FunctionSpaceComplexIntegration
FunctionSpaceFunctions2
FunctionSpaceIntegration
FunctionSpacePrimitiveElement
FunctionSpaceReduce
FunctionSpaceSum
FunctionSpaceToExponentialExpansion
```

FunctionSpaceToUnivariatePowerSeries
FunctionSpaceUnivariatePolynomialFactor
GaloisGroupFactorizationUtilities
GaloisGroupFactorizer
GaloisGroupPolynomialUtilities
GaloisGroupUtilities
GaussianFactorizationPackage
GeneralHenselPackage
GeneralizedMultivariateFactorize
GeneralPolynomialGcdPackage
GenerateUnivariatePowerSeries
GenExEuclid
GenUFactorize
GenusZeroIntegration
GosperSummationMethod
GraphicsDefaults
GrayCode
GroebnerFactorizationPackage
GroebnerInternalPackage
GroebnerPackage
GroebnerSolve
HallBasis
HeuGcd
IdealDecompositionPackage
IncrementingMaps
InfiniteProductCharacteristicZero
InfiniteProductFiniteField
InfiniteProductPrimeField
InfiniteTupleFunctions2
InfiniteTupleFunctions3
Infinity
InnerAlgFactor
InnerCommonDenominator
InnerMatrixLinearAlgebraFunctions
InnerMatrixQuotientFieldFunctions
InnerModularGcd
InnerMultFact
InnerNormalBasisFieldFunctions
InnerNumericEigenPackage
InnerNumericFloatSolvePackage
InnerPolySign
InnerPolySum
InnerTrigonometricManipulations
InputFormFunctions1
IntegerBits
IntegerCombinatoricFunctions
IntegerFactorizationPackage
IntegerLinearDependence
IntegerNumberTheoryFunctions
IntegerPrimesPackage

IntegerRetractions
IntegerRoots
IntegerSolveLinearPolynomialEquation
IntegralBasisPolynomialTools
IntegralBasisTools
IntegrationResultFunctions2
IntegrationResultRFToFunction
IntegrationResultToFunction
IntegrationTools
InternalPrintPackage
InternalRationalUnivariateRepresentationPackage
InverseLaplaceTransform
IrredPolyOverFiniteField
IrrRepSymNatPackage
KernelFunctions2
Kovacic
LaplaceTransform
LazardSetSolvingPackage
LeadingCoefDetermination
LexTriangularPackage
LinearDependence
LinearOrdinaryDifferentialOperatorFactorizer
LinearOrdinaryDifferentialOperatorsOps
LinearPolynomialEquationByFractions
LinearSystemMatrixPackage
LinearSystemMatrixPackage1
LinearSystemPolynomialPackage
LinGroebnerPackage
LiouvillianFunction
ListFunctions2
ListFunctions3
ListToMap
MakeBinaryCompiledFunction
MakeFloatCompiledFunction
MakeFunction
MakeRecord
MakeUnaryCompiledFunction
MappingPackage1
MappingPackage2
MappingPackage3
MappingPackageInternalHacks1
MappingPackageInternalHacks2
MappingPackageInternalHacks3
MatrixCategoryFunctions2
MatrixCommonDenominator
MatrixLinearAlgebraFunctions
MergeThing
MeshCreationRoutinesForThreeDimensions
ModularDistinctDegreeFactorizer
ModularHermitianRowReduction

MonoidRingFunctions2
MonomialExtensionTools
MoreSystemCommands
MPolyCatFunctions2
MPolyCatFunctions3
MPolyCatPolyFactorizer
MPolyCatRationalFunctionFactorizer
MRationalFactorize
MultFiniteFactorize
MultipleMap
MultiVariableCalculusFunctions
MultivariateFactorize
MultivariateLifting
MultivariateSquareFree
NagEigenPackage
NagFittingPackage
NagIntegrationPackage
NagInterpolationPackage
NagLapack
NagLinearEquationSolvingPackage
NAGLinkSupportPackage
NagMatrixOperationsPackage
NagOptimisationPackage
NagOrdinaryDifferentialEquationsPackage
NagPartialDifferentialEquationsPackage
NagPolynomialRootsPackage
NagRootFindingPackage
NagSeriesSummationPackage
NagSpecialFunctionsPackage
NewSparseUnivariatePolynomialFunctions2
NonCommutativeOperatorDivision
NoneFunctions1
NonLinearFirstOrderODESolver
NonLinearSolvePackage
NormalizationPackage
NormInMonogenicAlgebra
NormRetractPackage
NPCoef
NumberFieldIntegralBasis
NumberFormats
NumberTheoreticPolynomialFunctions
Numeric
NumericalOrdinaryDifferentialEquations
NumericalQuadrature
NumericComplexEigenPackage
NumericContinuedFraction
NumericRealEigenPackage
NumericTubePlot
OctonionCategoryFunctions2
ODEIntegration

```
ODETools
OneDimensionalArrayFunctions2
OnePointCompletionFunctions2
OpenMathPackage
OpenMathServerPackage
OperationsQuery
OrderedCompletionFunctions2
OrderingFunctions
OrthogonalPolynomialFunctions
OutputPackage
PadeApproximantPackage
PadeApproximants
PAdicWildFunctionFieldIntegralBasis
ParadoxicalCombinatorsForStreams
ParametricLinearEquations
ParametricPlaneCurveFunctions2
ParametricSpaceCurveFunctions2
ParametricSurfaceFunctions2
PartialFractionPackage
PartitionsAndPermutations
PatternFunctions1
PatternFunctions2
PatternMatch
PatternMatchAssertions
PatternMatchFunctionSpace
PatternMatchIntegerNumberSystem
PatternMatchIntegration
PatternMatchKernel
PatternMatchListAggregate
PatternMatchPolynomialCategory
PatternMatchPushDown
PatternMatchQuotientFieldCategory
PatternMatchResultFunctions2
PatternMatchSymbol
PatternMatchTools
Permanent
PermutationGroupExamples
PiCoercions
PlotFunctions1
PlotTools
PointFunctions2
PointPackage
PointsOfFiniteOrder
PointsOfFiniteOrderRational
PointsOfFiniteOrderTools
PolToPol
PolyGroebner
PolynomialAN2Expression
PolynomialCategoryLifting
PolynomialCategoryQuotientFunctions
```

PolynomialComposition
PolynomialDecomposition
PolynomialFactorizationByRecursion
PolynomialFactorizationByRecursionUnivariate
PolynomialFunctions2
PolynomialGcdPackage
PolynomialInterpolation
PolynomialInterpolationAlgorithms
PolynomialNumberTheoryFunctions
PolynomialRoots
PolynomialSetUtilitiesPackage
PolynomialSolveByFormulas
PolynomialSquareFree
PolynomialToUnivariatePolynomial
PowerSeriesLimitPackage
PrecomputedAssociatedEquations
PrimitiveArrayFunctions2
PrimitiveElement
PrimitiveRatDE
PrimitiveRatRicDE
PrintPackage
PseudoLinearNormalForm
PseudoRemainderSequence
PureAlgebraicIntegration
PureAlgebraicLODE
PushVariables
QuasiAlgebraicSet2
QuasiComponentPackage
QuaternionCategoryFunctions2
QuotientFieldCategoryFunctions2
RadicalEigenPackage
RadicalSolvePackage
RadixUtilities
RandomDistributions
RandomFloatDistributions
RandomIntegerDistributions
RandomNumberSource
RationalFactorize
RationalFunction
RationalFunctionDefiniteIntegration
RationalFunctionFactor
RationalFunctionFactorizer
RationalFunctionIntegration
RationalFunctionLimitPackage
RationalFunctionSign
RationalFunctionSum
RationalIntegration
RationalLODE
RationalRetractions
RationalRicDE

RationalUnivariateRepresentationPackage
RealPolynomialUtilitiesPackage
RealSolvePackage
RealZeroPackage
RealZeroPackageQ
RectangularMatrixCategoryFunctions2
ReducedDivisor
ReduceLODE
ReductionOfOrder
RegularSetDecompositionPackage
RegularTriangularSetGcdPackage
RepeatedDoubling
RepeatedSquaring
RepresentationPackage1
RepresentationPackage2
ResolveLatticeCompletion
RetractSolvePackage
SAERationalFunctionAlgFactor
ScriptFormulaFormat1
SegmentBindingFunctions2
SegmentFunctions2
SimpleAlgebraicExtensionAlgFactor
SimplifyAlgebraicNumberConvertPackage
SmithNormalForm
SortedCache
SortPackage
SparseUnivariatePolynomialFunctions2
SpecialOutputPackage
SquareFreeQuasiComponentPackage
SquareFreeRegularSetDecompositionPackage
SquareFreeRegularTriangularSetGcdPackage
StorageEfficientMatrixOperations
StreamFunctions1
StreamFunctions2
StreamFunctions3
StreamInfiniteProduct
StreamTaylorSeriesOperations
StreamTranscendentalFunctions
StreamTranscendentalFunctionsNonCommutative
StructuralConstantsPackage
SturmHabichtPackage
SubResultantPackage
SupFractionFactorizer
SymmetricFunctions
SymmetricGroupCombinatoricFunctions
SystemODESolver
SystemSolvePackage
TableauxBumpers
TabulatedComputationPackage
TangentExpansions

TemplateUtilities
TexFormat1
ToolsForSign
TopLevelDrawFunctions
TopLevelDrawFunctionsForAlgebraicCurves
TopLevelDrawFunctionsForCompiledFunctions
TopLevelDrawFunctionsForPoints
TopLevelThreeSpace
TranscendentalHermiteIntegration
TranscendentalIntegration
TranscendentalManipulations
TranscendentalRischDE
TranscendentalRischDESystem
TransSolvePackage
TransSolvePackageService
TriangularMatrixOperations
TrigonometricManipulations
TubePlotTools
TwoDimensionalPlotClipping
TwoFactorize
UnivariateFactorize
UnivariateLaurentSeriesFunctions2
UnivariatePolynomialCategoryFunctions2
UnivariatePolynomialCommonDenominator
UnivariatePolynomialDecompositionPackage
UnivariatePolynomialDivisionPackage
UnivariatePolynomialFunctions2
UnivariatePolynomialMultiplicationPackage
UnivariatePolynomialSquareFree
UnivariatePuiseuxSeriesFunctions2
UnivariateSkewPolynomialCategoryOps
UnivariateTaylorSeriesFunctions2
UnivariateTaylorSeriesODESolver
UniversalSegmentFunctions2
UserDefinedPartialOrdering
UserDefinedVariableOrdering
UTSodetools
VectorFunctions2
ViewDefaultsPackage
ViewportPackage
WeierstrassPreparation
WildFunctionFieldIntegralBasis
XExponentialPackage
ZeroDimensionalSolvePackage

Chapter 12

Research Topics

These are included here as ideas that may get expanded in more detail later.

12.1 Proofs

The goal would be to prove that Axiom's algorithms are correct.

For instance, show that the GCD algorithm is correct. This involves several levels of proof. At one level we need to prove that the GCD algorithm is mathematically correct and that it terminates. This can be picked up from the literature.

A second level of correctness involves proving that the implementation of the algorithm is correct. This involves using something like ACL2 [KMJ00] and proof of the common lisp implementation.

A third level is to show that the binary implementation conforms to the semantics of the common lisp implementation. This involves using something like Function Extraction (FX) [LMW79] to extract the machine-level behavior of the program and comparing it to the specification.

12.2 Indefinites

There are times when it would be convenient to write algorithms in terms of indefinite values. For instance, we would like to be able to declare that X and Y are matrices and compute X*Y symbolically. We would like to be able to do the same with arbitrary integers, I and J. In general, for a given domain we would like to create domain elements that are not fully specified but have the computation proceed with these "indefinite" values.

12.3 Provisos

We would like to create "provisos" on statements such as:

$$\frac{1}{x} \text{ provided } x \neq 0$$

We would then like to rewrite this in terms of intervals to create three "continuations" where each continuation is a separate domain of computation (and could thus be computed in parallel). So for the above example we would generate:

$$\frac{1}{x} \text{ such that } x \in [-\infty, 0)$$
$$\frac{1}{x} \text{ such that } x \in (0, 0)$$
$$\frac{1}{x} \text{ such that } x \in (0, \infty]$$

When a new proviso is added, for instance, when we divide by y then there would be further subdivision of the computation, forming a tree:

$$\frac{1}{xy} \text{ such that } x \in [-\infty, 0) \text{ and } y \in [-\infty, 0)$$
$$\frac{1}{xy} \text{ such that } x \in (0, 0) \text{ and } y \in [-\infty, 0)$$
$$\frac{1}{xy} \text{ such that } x \in (0, \infty] \text{ and } y \in [-\infty, 0)$$
$$\frac{1}{xy} \text{ such that } x \in [-\infty, 0) \text{ and } y \in (0, 0)$$
$$\frac{1}{xy} \text{ such that } x \in (0, 0) \text{ and } y \in (0, 0)$$
$$\frac{1}{xy} \text{ such that } x \in (0, \infty] \text{ and } y \in (0, 0)$$
$$\frac{1}{xy} \text{ such that } x \in [-\infty, 0) \text{ and } y \in (0, \infty]$$
$$\frac{1}{xy} \text{ such that } x \in (0, 0) \text{ and } y \in (0, \infty]$$
$$\frac{1}{xy} \text{ such that } x \in (0, \infty] \text{ and } y \in (0, \infty]$$

Interesting questions arise, such has how to recover the function over the real line. Of course, the domain and range are not restricted to the real line in general but could, for instance, range over the complex plane.

Note that the provisos need not be an interval. They could be anything such as a polynomial or a property like "$f(x)$ is entire".

Chapter 13

Makefile

13.1 Environment variables

— **make.environment** —

```
BOOK=${SPD}/books/bookvol6.pamphlet

# this is where we are compiling from
IN=     ${SRC}/sman

# this is the intermediate place
MID=    ${INT}/sman

# this is the intermediate place
MIDOBJ=    ${OBJ}/${SYS}/sman

# this is where to put the various commands
OUT= ${MNT}/${SYS}/bin
OUTLIB= ${MNT}/${SYS}/lib

# this is where the include files live
INC=    ${SRC}/include

# this is where we hid the libspad library
LIB= ${OBJ}/${SYS}/lib

# this is where the documentation ends up
DOC=    ${MNT}/${SYS}/doc
CFLAGS= ${CCF}
LDFLAGS= -L${LIB} -lspad ${LDF}

SMANOBJS= ${LIB}/libspad.a
```

———-

13.2 The axiom command

— **make.axiomcmd** —

```
${OUT}/axiom: ${BOOK}
@echo 1 making ${OUT}/axiom from ${BOOK}
@ (cd ${OUT} ; \
        echo '(tangle "${BOOK}" "axiomcmd" "axiom")' | ${OBJ}/${SYS}/bin/lisp )
@chmod +x ${OUT}/axiom
@ cp ${OUT}/axiom ${MID}
```

———-

13.3 session

— **make.session** —

```
${OUTLIB}/session: ${SMANOBJS} ${MIDOBJ}/session.o
@ echo 1 linking session
@ ${CC} -o ${OUTLIB}/session ${MIDOBJ}/session.o ${SMANOBJS}

${MID}/session.c: ${BOOK}
@ echo 2 making ${MID}/session.c from ${BOOK}
@ (cd ${MID} ; \
        echo '(tangle "${BOOK}" "session" "session.c")' | \
          ${OBJ}/${SYS}/bin/lisp )

${MIDOBJ}/session.o: ${MID}/session.c ${INC}/session.h1
@ echo 3 making ${MIDOBJ}/session.o from ${MID}/session.c
@ ( cd ${MIDOBJ} ; ${CC} -c ${CFLAGS} ${MID}/session.c -I${INC} )
```

———-

13.4 nagman

Note that we do not build the nagman component as we do not have the necessary code (for instance, callnag).

— **make.nagman** —

```
${OUT}/nagman: ${SMANOBJS} ${MIDOBJ}/nagman.o
@ echo 5 linking nagman
@ ${CC} -o ${OUT}/nagman ${MIDOBJ}/nagman.o ${SMANOBJS}

${MID}/nagman.c: ${BOOK}
@ echo 6 making ${MID}/nagman.c from ${IN}/bookvol6.pamphlet
@ (cd ${MID} ; \
        echo '(tangle "${BOOK}" "nagman" "nagman.c")' | \
           ${OBJ}/${SYS}/bin/lisp )

${MIDOBJ}/nagman.o: ${MID}/nagman.c ${INC}/nagman.h1
@ echo 7 making ${MIDOBJ}/nagman.o from ${MID}/nagman.c
@ ( cd ${MIDOBJ} ; ${CC} -c ${CFLAGS} ${MID}/nagman.c -I${INC} )
```

————-

13.5 spadclient

— **make.spadclient** —

```
${OUTLIB}/spadclient: ${SMANOBJS} ${MIDOBJ}/spadclient.o
@ echo 9 linking spadclient
@ ${CC} -o ${OUTLIB}/spadclient ${MIDOBJ}/spadclient.o  ${SMANOBJS}

${MID}/spadclient.c: ${BOOK}
@ echo 10 making ${MID}/spadclient.c from ${IN}/bookvol6.pamphlet
@ (cd ${MID} ; \
        echo '(tangle "${BOOK}" "spadclient" "spadclient.c")' | \
           ${OBJ}/${SYS}/bin/lisp )

${MIDOBJ}/spadclient.o: ${MID}/spadclient.c ${INC}/spadclient.h1
@ echo 11 making ${MIDOBJ}/spadclient.o from ${MID}/spadclient.c
@ ( cd ${MIDOBJ} ; ${CC} -c ${CFLAGS} ${MID}/spadclient.c -I${INC} )
```

————-

13.6 sman

— **make.sman** —

```
${OUT}/sman: ${SMANOBJS} ${MIDOBJ}/sman.o
@ echo 13 linking sman
@ ${CC} -o ${OUT}/sman ${MIDOBJ}/sman.o ${SMANOBJS}

${MID}/sman.h: ${BOOK}
@ echo 00 making ${MID}/sman.h from ${IN}/bookvol6.pamphlet
@ (cd ${MID} ; \
        echo '(tangle "${BOOK}" "sman.h" "sman.h")' | \
           ${OBJ}/${SYS}/bin/lisp )

${MID}/sman.c: ${MID}/sman.h ${BOOK}
@ echo 14 making ${MID}/sman.c from ${IN}/bookvol6.pamphlet
@ (cd ${MID} ; \
        echo '(tangle "${BOOK}" "sman" "sman.c")' | \
           ${OBJ}/${SYS}/bin/lisp )

${MIDOBJ}/sman.o: ${MID}/sman.c ${INC}/sman.h1
@ echo 15 making ${MIDOBJ}/sman.o from ${MID}/sman.c
@ ( cd ${MIDOBJ} ; ${CC} -I${INC} -I${MID} -c ${CFLAGS} ${MID}/sman.c )
```

—— * ——

```
\getchunk{make.environment}
all: ${OUTLIB}/session ${OUTLIB}/spadclient ${OUT}/sman ${OUT}/axiom
@ echo 18 finished ${IN}

clean:
@echo 19 cleaning ${SRC}/sman

\getchunk{make.axiomcmd}
\getchunk{make.sman}
\getchunk{make.session}
\getchunk{make.spadclient}
\getchunk{make.nagman}
```

Bibliography

[1] Jenks, R.J. and Sutor, R.S.
"Axiom – The Scientific Computation System"
Springer-Verlag New York (1992) ISBN 0-387-97855-0

[2] Knuth, Donald E., "Literate Programming"
Center for the Study of Language and Information ISBN 0-937073-81-4 Stanford CA (1992)

[3] Page, William, "The Axiom Wiki Website"
http://wiki.axiom-developer.org

[4] Watt, Stephen, "Aldor",
http://www.aldor.org

[5] Lamport, Leslie,
"Latex – A Document Preparation System", Addison-Wesley, New York ISBN 0-201-52983-1

[6] Ramsey, Norman
"Noweb – A Simple, Extensible Tool for Literate Programming"
http://www.eecs.harvard.edu/ ~nr/noweb

[7] Axiom Book Volume 7 – Hyperdoc
file://usr/local/axiom/src/hyper/bookvol7.pamphlet

[8] Axiom Book Volume 8 – Graphics
file://usr/local/axiom/src/graph/bookvol8.pamphlet

[9] AIX Version 3.2 and 4 Performance Tuning Guide
http://www.rs6000.ibm.com/doc_link/en_US/
a_doc_lib/aixbman/prftungd/toc.htm