

1

Rosetta Translations

1.1 Introduction

The following is a collection of synonyms for various operations in the computer algebra systems Aldor, Axiom, CoCoA, Derive, DoCon, GAP, Giac, GiNaC, Gmp, Macsyma, Magnus, Maple, Mathcad, Mathematica, Maxima, MuPAD, Octave, Pari, Reduce, Sum^it, Singular and Yacas. This collection does not attempt to be comprehensive, but hopefully it will be useful in giving an indication of how to translate between the syntaxes used by the different systems in many common situations. Note that a blank entry means either (a) that there may be an exact translation of a particular operation for the indicated system, but we don't know what it is or (b) there is no exact translation but it may still be possible to work around this lack with a related functionality.

While commercial systems are not provided on this CD the intent of the Rosetta effort is to make it possible for experienced Computer Algebra users to experiment with other systems. Thus the commands for commercial systems are included to allow users of those systems to translate.

The tables do not cover the full range of functionality and are slanted in favor of general purpose, interactive systems. Some of these systems are special purpose and do not support a lot of the functionality of the more general purpose systems. Where they do support an interpreter the commands are provided. Some systems are primarily development tools rather than interpreters and are not represented in some tables.

Originally written by Michael Wester.

Modified for Rosetta by Timothy Daly, Alexander Hulpke (GAP), Hans Schoenemann (Singular) and Serge Mechveliani (DoCon).

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

1.2 System availability

System	License	Status	Web Location
Aldor	BSD	available	http://www.aldor.org
Axiom	open source	pending	http://home.earthlink.net/~jgg964/axiom.html
CLN	GPL	Rosetta	ftp://ftp.ilog.fr/pub/Users/haible/gnu
CoCoA	unknown	Rosetta	http://cocoa.dima.unige.it
Derive	commercial	available	http://www.mathware.com
DoCon	open source	available	http://www.haskell.org/docon
GAP	GPL	Rosetta	http://www.gap-system.org/~gap
Giac	GPL	Rosetta	http://www-fourier.ujf-grenoble.fr/~parisse/giac_us.html
GiNaC	GPL	Rosetta	http://www.ginac.de/Download.html
Gmp	GPL	Rosetta	http://www.swox.com/gmp
Macsyma	commercial	dead	unavailable
Magnus	GPL	Rosetta	http://zebra.sci.cny.cuny.edu/web
Maxima	GPL	Rosetta	http://www.ma.utexas.edu/maxima.html
Maple	commercial	available	http://www.maplesoft.com
Mathcad	commercial	available	http://www.mathcad.com
Mathematica	commercial	available	http://www.wolfram.com
Mpfr	GPL	Rosetta	http://www.loria.fr/equipes/polka
MuPAD	commercial	available	http://www.mupad.de
Ntl	GPL	Rosetta	http://shoup.net/ntl
Octave	GPL	Rosetta	http://www.octave.org
Pari	GPL	Rosetta	http://www.parigp-home.de
Reduce	commercial	available	http://www.zib.de/Symbolik/reduce
Singular	Singular	Rosetta	http://www.singular.uni-kl.de
Sum [^] it		pending	http://www-sop.inria.fr/cafe/Manuel.Bronstein
Yacas	GPL	Rosetta	http://yacas.sourceforge.net

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

System	Type	Interpreted, Compiled, Library
Aldor	library development	compiler and library
Axiom	General Purpose	interp and compiled
CLN	arb. prec. arithmetic	library
CoCoA	Commutative Algebra	
Derive	General Purpose	
DoCon	General Purpose	interpreted
GAP	Group Theory	interpreted
Giac	C++ algebra library	library
GiNaC	C++ algebra library	library
Gmp	arb. prec. arithmetic	library
Macsyma	General Purpose	
Magnus	Infinite Group Theory	
Maxima	General Purpose	
Maple	General Purpose	
Mathcad	General Purpose	
Mathematica	General Purpose	
Mpfr	arb. prec. arithmetic	library
MuPAD	General Purpose	
Ntl	Number Theory library	library
Octave	Numerical Computing	
Pari	Number Theory	
Reduce	General Purpose	
Singular	Polynomial Computations	
Sum^it	ODE and Diff. eqns	library
Yacas	General Purpose	

1.3 Programming and Miscellaneous

System	License	Status	Web Location
GCL	GPL	Rosetta	http://www.gnu.org/software/gcl
GnuPlot	GPL	Rosetta	http://www.gnuplot.org
Haskell	GPL	Rosetta	http://www.haskell.org
Readline	GPL	Rosetta	http://prep.ai.mit.edu/pub/gnu/readline
TeXmacs	GPL	Rosetta	http://www.texmacs.org

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Unix/Microsoft user initialization file	
Axiom	~/axiom.input	
CoCoA	~/init.coc ~/userinit.coc	
Derive		derive.ini
DoCon	~/ghci	(interactive system init.)
GAP	~/gaprc	GAP.RC
Giac	~/xcasrc	
GiNaC		
Macsyma	~/macsyma-init.macsyma	mac-init.mac
Magnus		
Maple	~/mapleinit	maplev5.ini
Mathcad		
Mathematica	~/init.m	init.m
Maxima	~/macsyma-init.macsyma	mac-init.mac
MuPAD	~/mupadinit	\mupad\bin\userinit.mu
Octave		
Pari	~/gprc	
Reduce	~/reducerc	reduce.rc
Singular	~/singularrc	~/singularrc
Yacas		

	Describe <i>keyword</i>	Find keywords containing <i>pattern</i>
Axiom)what operations pattern
CoCoA	H.Command("");	Man("keyword");
Derive		
DoCon		
GAP	?keyword	??keyword
Giac	?keyword	
GiNaC		
Macsyma	describe("keyword")\$	apropos("pattern");
Magnus		
Maple	?keyword	?pattern ¹
Mathcad	[F1]	[Shift] [F1]
Mathematica	?keyword	?*pattern*
Maxima	describe("keyword")\$	apropos("pattern");
MuPAD	?keyword	?*pattern*
Octave	help -i keyword	
Pari		
Reduce		
Singular	?keyword;	?*pattern*;
Yacas		

¹Only if the pattern is not a keyword and then the matches are simplistic.

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Comment	Line continuation	Prev. expr.	Case sensitive	Variables assumed
Axiom	-- comment	input _<CR>input	%	Yes	real
CoCoA	-- comment	input<CR>input;	It	Yes	
Derive	"comment"	input ~<CR>input		No	real
DoCon					
Giac	/* comment */	input<CR>input;	ans()	Yes	real
GAP	# comment	input\<CR>input	last	Yes	no assumption
Macsyma	/* comment */	input<CR>input;	%	No	real
Magnus					
Maple	# comment	input<CR>input;	%	Yes	complex
Mathcad	Text Region				
Mathematica	(* comment *)	input<CR>input	%	Yes	complex
Maxima	/* comment */	input<CR>input;	%	No	real
MuPAD	# comment #	input<CR>input;	%	Yes	complex
Octave	##			Yes	
Pari					
Reduce	% comment	input<CR>input;	ws	No	complex
Singular	// comment	input<CR>input;	-	Yes	none
Yacas					
	Load a file	Time a command		Quit	
Axiom)read "file")quiet)set messages time on)quit	
CoCoA	<< 'MySession'	Time T := F();		Quit; or Ciao;	
Derive	[Transfer Load Derive]			[Quit]	
DoCon					
Giac	read("file"):	always on		Ctrl-D/[Quit]	
GAP	Read("file");	time; (also see Runtime());		quit;	
Macsyma	load("file")\$	showtime: all\$		quit();	
Magnus					
Maple	read("file"):	readlib(showtime): on;		quit	
Mathcad	File: Open Document			File: Exit	
Mathematica	<< file	Timing[command]		Quit[]	
Maxima	load("file")\$	showtime: all\$		quit();	
MuPAD	read("file"):	time(command);		quit	
Octave	load file	tic(); cmd ; toc()		quit or exit	
Pari					
Reduce	in "file"\$	on time;		quit;	
Singular	< "file";	timer=1;		quit;	
Yacas					

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Display output	Suppress output	Substitution: $f(x, y) \rightarrow f(z, w)$
Axiom	input	input;	subst(f(x, y), [x = z, y = w])
CoCoA	input;		Subst(F, [z,w]);
Derive	input	var:= input	[Manage Substitute]
DoCon			
Giac	input	nodisp(input)	subst(f(x, y), [x, y], [z, w]);
GAP	input;	input;;	Value(f, [x,y], [z,w]); ²
Macsyma	input;	input\$	subst([x = z, y = w], f(x, y));
Magnus			
Maple	input;	input:	subs({x = z, y = w}, f(x, y));
Mathcad			
Mathematica	input	input;	f[x, y] /. {x -> z, y -> w}
Maxima	input;	input\$	subst([x = z, y = w], f(x, y));
MuPAD	input;	input:	subs(f(x, y), [x = z, y = w]);
Octave	input	input;	
Pari			
Reduce	input;	input\$	sub({x = z, y = w}, f(x, y));
Singular	input;	default	substitute(f,x,z,y,w)
Yacas			
	Set	List	Matrix
Axiom	set [1, 2]	[1, 2]	matrix([[1, 2], [3, 4]])
CoCoA	Set([1,2])	[1, 2]	Mat[[1,2], [3,4]];
Derive	{1, 2}	[1, 2]	[[1,2], [3,4]]
DoCon			
Giac	set[1,2]	[1, 2]	[[1,2], [3,4]]
GAP	Set([1,2])	[1, 2]	[[1,2], [3,4]] ³
Macsyma	[1, 2]	[1, 2]	matrix([1, 2], [3, 4])
Magnus			
Maple	{1, 2}	[1, 2]	matrix([[1, 2], [3, 4]])
Mathcad			Math: Matrices
Mathematica	{1, 2}	{1, 2}	{{1, 2}, {3, 4}}
Maxima	[1, 2]	[1, 2]	matrix([1, 2], [3, 4])
MuPAD	{1, 2}	[1, 2]	export(Dom): export(linalg): matrix:= ExpressionField(normal): matrix([[1, 2], [3, 4]])
Octave			
Pari			
Reduce	{1, 2}	{1, 2}	mat((1, 2), (3, 4))
Singular			
Yacas			

²Only if f is a rational function involving the variables x and y .

³There are special compressed matrices over finite fields

	Equation	List element	Matrix element	Length of a list
Axiom	$x = 0$	1 . 2	m(2, 3)	#1
CoCoA		L[2]	L[2,3]	Len(L)
Derive	$x = 0$	1 SUB 2	m SUB 2 SUB 3	DIMENSION(1)
DoCon				
Giac	$x = 0$	1[2]	m[2, 3]	size(1)
GAP	$x = 0$	1[2]	m[2][3]	Length(1)
Macsyma	$x = 0$	1[2]	m[2, 3]	length(1)
Magnus				
Maple	$x = 0$	1[2]	m[2, 3]	nops(1)
Mathcad		1[2]	M[2,3]	
Mathematica	$x == 0$	1[[2]]	m[[2, 3]]	Length[1]
Maxima	$x = 0$	1[2]	m[2, 3]	length(1)
MuPAD	$x = 0$	1[2]	m[2, 3]	nops(1)
Octave				
Pari				
Reduce	$x = 0$	part(1, 2)	m(2, 3)	length(1)
Singular	$x == 0$	1[2]	m[2, 3]	size(1)
Yacas				
	Prepend/append an element to a list			Append two lists
Axiom	cons(e, 1)		concat(1, e)	append(l1, l2)
CoCoA	Concat([E1], L1)		Append(L, E)	Concat(L1, L2)
Derive	APPEND([e], 1)		APPEND(1, [e])	APPEND(l1, l2)
DoCon				
Giac	prepend(1, e)		append(1, e)	concat(l1, l2)
GAP	Concatenation([e], 1)		Add(1, e)	Append(l1, l2)
Macsyma	cons(e, 1)		endcons(e, 1)	append(l1, l2)
Magnus				
Maple	[e, op(1)]		[op(1), e]	[op(l1), op(l2)]
Mathcad				
Mathematica	Prepend[1, e]		Append[1, e]	Join[l1, l2]
Maxima	cons(e, 1)		endcons(e, 1)	append(l1, l2)
MuPAD	[e, op(1)]		append(1, e)	l1 . l2
Octave				
Pari				
Reduce	e . 1		append(1, e)	append(l1, l2)
Singular	insert(1, e, 1)		insert(1, e, size(1)+1)	l1+l2
Yacas				

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Matrix column dimension	Convert a list into a column vector
Axiom	<code>ncols(m)</code>	<code>transpose(matrix([1]))</code>
CoCoA	<code>Len(ColumnVectors(M))</code>	<code>Transposed(Mat(Vector([1])))</code>
Derive	<code>DIMENSION(m SUB 1)</code>	<code>[1]`</code>
DoCon		
Giac	<code>size(mat[0])</code>	<code>tran([1])</code>
GAP	<code>Length(mat[1])</code>	objects are identical
Macsyma	<code>mat_ncols(m)</code>	<code>transpose(matrix(1))</code>
Magnus		
Maple	<code>linalg[coldim](m)</code>	<code>linalg[transpose](matrix([1]))</code>
Mathcad		
Mathematica	<code>Dimensions[m][[2]]</code>	<code>Transpose[{1}]</code>
Maxima	<code>mat_ncols(m)</code>	<code>transpose(matrix(1))</code>
MuPAD	<code>linalg::ncols(m)</code>	<code>transpose(matrix([1]))</code> ⁴
Octave		
Pari		
Reduce	<code>load_package(linalg)\$ column_dim(m)</code>	<code>matrix v(length(1), 1)\$ for i:=1:length(1) do v(i, 1):= part(1, i)</code>
Singular	<code>ncols(m)</code>	<code>vector v; for(int i=1;i<=size(1);i++) { v[i]=1[i]; }</code>
Yacas		

⁴See the definition of `matrix` above.

	Convert a column vector into a list
Axiom	<code>[v(i, 1) for i in 1..nrows(v)]</code>
CoCoA	<code>V:=ColumnVectors(M); List(V[1]);</code>
Derive	<code>v` SUB 1</code>
DoCon	
Giac	<code>tran(v)[0]</code>
GAP	objects are identical
Macsyma	<code>part(transpose(v), 1)</code>
Magnus	
Maple	<code>op(convert(linalg[transpose](v), listlist))</code>
Mathcad	
Mathematica	<code>Flatten[v]</code>
Maxima	<code>part(transpose(v), 1)</code>
MuPAD	<code>[op(v)]</code>
Octave	
Pari	
Reduce	<code>load_package(linalg)\$ for i:=1:row_dim(v) collect(v(i, 1))</code>
Singular	<code>list l; for(int i; i<=size(v);i++) { l[i]=v[i]; }</code>
Yacas	

	True	False	And	Or	Not	Equal	Not equal
Axiom	<code>true</code>	<code>false</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code>~=</code>
CoCoA	<code>TRUE</code>	<code>FALSE</code>	<code>AND</code>	<code>OR</code>	<code>NOT</code>	<code>=</code>	<code><></code>
Derive	<code>TRUE</code>	<code>FALSE</code>	<code>AND</code>	<code>OR</code>	<code>NOT</code>	<code>=</code>	<code>/=</code>
DoCon							
Giac	<code>1</code>	<code>0</code>	<code>&&</code>	<code> </code>	<code>!</code>	<code>==</code>	<code>!=</code>
GAP	<code>true</code>	<code>false</code> ⁵	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code><></code>
Macsyma	<code>true</code>	<code>false</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code>#</code>
Magnus							
Maple	<code>true</code>	<code>false</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code><></code>
Mathcad							
Mathematica	<code>True</code>	<code>False</code>	<code>&&</code>	<code> </code>	<code>!</code>	<code>==</code>	<code>!=</code>
Maxima	<code>true</code>	<code>false</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code>#</code>
MuPAD	<code>true</code>	<code>false</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code><></code>
Octave							
Pari							
Reduce	<code>t</code>	<code>nil</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>=</code>	<code>neq</code>
Singular	<code>1</code>	<code>0</code>	<code>and</code>	<code>or</code>	<code>not</code>	<code>==</code>	<code><></code>
Yacas							

⁵Some commands may also return `fail` to indicate failure.

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	If+then+else statements	Strings (concatenated)
Axiom	if _ then _ else if _ then _ else _	concat(["x", "y"])
CoCoA	If _ Then _ Elself _ Then _ Else _ End	Concat("x","y")
Derive	IF(_, _, IF(_, _, _))	"xy"
DoCon		
Giac	if (_) { ; } else { ; }	"x"+"y"
GAP	if _ then _ elif _ then _ else _ fi	Concatenation("x","y")
Macsyma	if _ then _ else if _ then _ else _	concat("x", "y")
Magnus		
Maple	if _ then _ elif _ then _ else _ fi	"x" . "y"
Mathcad		
Mathematica	If[, _, If[, _, _]]	"x" <> "y"
Maxima	if _ then _ else if _ then _ else _	concat("x", "y")
MuPAD	if _ then _ elif _ then _ else _ end_if	"x" . "y"
Octave		
Pari		
Reduce	if _ then _ else if _ then _ else _	"xy" or mkid(x, y)
Singular	if (.) { - } else { - }	"x"+"y"
Yacas		
	Simple loop and Block	Generate the list [1,2,...,n]
Axiom	for i in 1..n repeat (x; y)	[f(i) for i in 1..n]
CoCoA	For I:=1 To N Do _ End;	List(1..10)
Derive	VECTOR([x, y], i, 1, n)	VECTOR(f(i), i, 1, n)
DoCon		
Giac	for (;;){ }	makelist(f,1,10)
GAP	for i in [1..n] do _ od;	[1..n] or [1,2..n]
Macsyma	for i:1 thru n do (x, y);	makelist(f(i), i, 1, n);
Magnus		
Maple	for i from 1 to n do x; y od;	[f(i) \$ i = 1..n];
Mathcad		
Mathematica	Do[x; y, {i, 1, n}]	Table[f[i], {i, 1, n}]
Maxima	for i:1 thru n do (x, y);	makelist(f(i), i, 1, n);
MuPAD	for i from 1 to n do x; y end_for;	[f(i) \$ i = 1..n];
Octave		
Pari		
Reduce	for i:=1:n do <<x; y>>;	for i:=1:n collect f(i);
Singular	for(int i=n;i>0;i--) { x;y; }	list l; for(int i=n;i>0;i--)
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Complex loop iterating on a list		
Axiom	for x in [2, 3, 5] while x**2 < 10 repeat output(x)		
CoCoA	For X In [2, 3, 5] Do While X^2 < 10 Do PrintLn(X) End; End;		
Derive			
DoCon			
Giac	l:=[2,3,5]; for (j:=0;(j<size(l)) && (l[j]^2<10);j++){ print(l[j]); }		
GAP	for x in [2, 3, 5] do while x^2<10 do Print(x);od;od;		
Macysma	for x in [2, 3, 5] while x^2 < 10 do print(x)\$		
Magnus			
Maple	for x in [2, 3, 5] while x^2 < 10 do print(x) od:		
Mathcad			
Mathematica	For[l = {2, 3, 5}, l != {} && l[[1]]^2 < 10, l = Rest[l], Print[l[[1]]]]		
Maxima	for x in [2, 3, 5] while x^2 < 10 do print(x)\$		
MuPAD	for x in [2, 3, 5] do if x^2 < 10 then print(x) end_if end_for:		
Octave			
Pari			
Reduce	for each x in {2, 3, 5} do if x^2 < 10 then write(x)\$		
Singular	for(l=list(2,3,5);size(l)>0;l=delete(l,1)) { if(l[1]^2<10) { l[1];}}		
Yacas			
	Assignment	Function definition	Clear vars and funs
Axiom	y:= f(x)	f(x, y) == x*y)clear properties y f
CoCoA	Y:= F(x);	F(X,Y):=X*Y;	Delete Y;
Derive	y:= f(x)	f(x, y):= x*y	y:= f:=
DoCon			
Giac	y:= f(x)	f(x,y):= x*y	purge(y)
GAP	y:= f(x);	f:=function(x, y) return x*y; end;	no symbolic variables
Macysma	y: f(x);	f(x, y):= x*y;	remvalue(y)\$ remfunction(f)\$
Magnus			remfunction(f)\$
Maple	y:= f(x);	f:= proc(x, y) x*y end;	y:= 'y': f:= 'f':
Mathcad			
Mathematica	y = f[x]	f[x_, y_]:= x*y	Clear[y, f]
Maxima	y: f(x);	f(x, y):= x*y;	remvalue(y)\$
MuPAD	y:= f(x);	f:= proc(x, y) begin x*y end_proc;	y:= NIL: f:= NIL:
Octave			
Pari			
Reduce	y:= f(x);	procedure f(x, y); x*y;	clear y, f;
Singular	y=f(x);	proc f(x,y)	kill y,f;
Yacas			

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Function definition with a local variable	
Axiom	f(x) == (local n; n:= 2; n*x)	
CoCoA	Define F(X) N:=2; Return N*; End;	
Derive		
DoCon		
Giac	f(x):={ local n; n:=2; return(n*x); }	
GAP	f:=function(x) local n; n:=2;return n*x; end;	
Macysma	f(x):= block([n], n: 2, n*x);	
Magnus		
Maple	f:= proc(x) local n; n:= 2; n*x end;	
Mathcad		
Mathematica	f[x_]:= Module[{n}, n = 2; n*x]	
Maxima	f(x):= block([n], n: 2, n*x);	
MuPAD	f:= proc(x) local n; begin n:= 2; n*x end_proc;	
Octave		
Pari		
Reduce	procedure f(x); begin scalar n; n:= 2; return(n*x) end;	
Singular	proc f(int x) { int n=2; return(n*x); }	
Yacas		
	Return unevaluated symbol	Define a function from an expression
Axiom	e:= x*y; 'e	function(e, f, x, y)
CoCoA		
Derive	e:= x*y 'e	f(x, y):= e
DoCon		
Giac	e:=x*y quote(e)	f:=program([x,y],[0,0],e)
GAP	No unevaluated symbols ⁶	
Macysma	e: x*y\$ 'e;	define(f(x, y), e);
Magnus		
Maple	e:= x*y: 'e';	f:= unapply(e, x, y);
Mathcad		
Mathematica	e = x*y; HoldForm[e]	f[x_, y_] = e
Maxima	e: x*y\$ 'e;	define(f(x, y), e);
MuPAD	e:= x*y: hold(e);	f:= hold(func)(e, x, y);
Octave		
Pari		
Reduce	e:= x*y\$	for all x, y let f(x, y):= e;
Singular	string e="x*y";	proc f(x,y) { execute("return("+e+"");};}
Yacas		

⁶Variables can be assigned to generators of a suitable free object, for example `x:=X(Rationals,"x");` or `f:=FreeGroup(2);x:=f.1;`

	Fun. of an indefinite number of args	Apply “+” to sum a list
Axiom		<code>reduce(+, [1, 2])</code>
CoCoA		<code>Sum([1,2]);</code>
Derive	<code>LST 1:= 1</code>	
DoCon		
Giac	<code>lst():= { print(args); }</code>	<code>'+'([1,2])</code>
GAP	<code>lst:=function(args) - end;</code>	<code>Sum([1,2])</code>
Macsyma	<code>lst([1]):= 1;</code>	<code>apply("+", [1, 2])</code>
Magnus		
Maple	<code>lst:=proc() [args[1..nargs]] end;</code>	<code>convert([1, 2], `+`)</code>
Mathcad		
Mathematica	<code>lst[l_...]:= {1}</code>	<code>Apply[Plus, {1, 2}]</code>
Maxima	<code>lst([1]):= 1;</code>	<code>apply("+", [1, 2])</code>
MuPAD	<code>lst:= proc(l) begin [args()]</code> <code>end_proc;</code>	<code>_plus(op([1, 2]))</code>
Octave		
Pari		
Reduce		<code>xapply(+, {1, 2})⁷</code>
Singular	<code>proc f(list #) { - }</code>	
Yacas		
	Apply a fun. to a list of its args	Map an anonymous function onto a list
Axiom	<code>reduce(f, l)</code>	<code>map(x +-> x + y, [1, 2])</code>
CoCoA	<code>Call(Function("F"),l)</code>	<code>L:=[];</code> <code>Foreach I In [1,2] Do</code> <code>Append(L,Call(Function("F"),I); End;</code>
Derive		<code>x:= [1, 2]</code> <code>VECTOR(x SUB i + y, i, 1, DIMENSION(x))</code>
DoCon		
Giac	<code>map(f,l)</code>	NA (C++)
GAP	<code>List(l,f)</code>	<code>List([1,2],x->x+y)</code>
Macsyma	<code>apply(f, l)</code>	<code>map(lambda([x], x + y), [1, 2])</code>
Magnus		
Maple	<code>f(op(l))</code>	<code>map(x -> x + y, [1, 2])</code>
Mathcad		
Mathematica	<code>Apply[f, l]</code>	<code>Map[# + y &, {1, 2}]</code>
Maxima	<code>apply(f, l)</code>	<code>map(lambda([x], x + y), [1, 2])</code>
MuPAD	<code>f(op(l))</code>	<code>map([1, 2], func(x + y, x))</code>
Octave		
Pari		
Reduce	<code>xapply(f, l)</code>	<code>for each x in {1, 2} collect x + y</code>
Singular		
Yacas		

⁷procedure xapply(f, lst); lisp(f . cdr(lst))\$

	Pattern matching: $f(3y) + f(zy) \rightarrow 3f(y) + f(zy)$
Axiom	<pre>f := operator('f); (rule f((n integer?(n)) * x) == n*f(x))(- f(3*y) + f(z*y))</pre>
CoCoA	
Derive	
DoCon	
Giac	NA/C++
GAP	
Macsyma	<pre>matchdeclare(n, integerp, x, true)\$ defrule(fnx, f(n*x), n*f(x))\$ apply1(f(3*y) + f(z*y), fnx);</pre>
Magnus	
Maple	<pre>map(proc(q) local m; if match(q = f(n*y), y, 'm') and type(rhs(op(m)), integer) then subs(m, n * f(y)) else q fi end, f(3*y) + f(z*y));</pre>
Mathcad	
Mathematica	<pre>f[3*y] + f[z*y] /. f[n_Integer * x_] -> n*f[x]</pre>
Maxima	<pre>matchdeclare(n, integerp, x, true)\$ defrule(fnx, f(n*x), n*f(x))\$ apply1(f(3*y) + f(z*y), fnx);</pre>
MuPAD	<pre>d:= domain("match"): d::FREEVARIABLE:= TRUE: n:= new(d, "n", func(testtype(m, DOM_INT), m)): x:= new(d, "x", TRUE): map(f(3*y) + f(z*y), proc(q) local m; begin m:= match(q, f(n*x)); if m = FAIL then q else subs(hold("n" * f("x")), m) end_if end_proc);</pre>
Octave	
Pari	
Reduce	<pre>operator f; f(3*y) + f(z*y) where {f(~n * ~x) => n*f(x) when fixp(n)};</pre>
Singular	
Yacas	

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Define a new infix operator and then use it		
Axiom			
CoCoA			
Derive			
DoCon			
GAP	One can overload existing infix operators for ones own purposes		
Giac	C++ (loadable module)		
Macsyma	infix("~")\$ ~"(x, y):= sqrt(x^2 + y^2)\$ 3 ~ 4;		
Magnus			
Maple	`&~`:= (x, y) -> sqrt(x^2 + y^2): 3 &~ 4;		
Mathcad			
Mathematica	x_ \[Tilde] y_:= Sqrt[x^2 + y^2]; 3 \[Tilde] 4		
Maxima	infix("~")\$ ~"(x, y):= sqrt(x^2 + y^2)\$ 3 ~ 4;		
MuPAD	tilde:= proc(x, y) begin sqrt(x^2 + y^2) end_proc: 3 &tilde 4;		
Octave			
Pari			
Reduce	infix \$ procedure (x, y); sqrt(x^2 + y^2)\$ 3 4;		
Singular			
Yacas			
	Main expression operator	1 st operand	List of expression operands
Axiom ⁸		kernels(e) . 1	kernels(e)
CoCoA			
Derive			<i>various</i> ⁹
DoCon			
Giac	sommet(e)	feuille(e)[0]	feuille(e)
GAP	There are no formal unevaluated expressions		
Macsyma	part(e, 0)	part(e, 1)	args(e)
Magnus			
Maple	op(0, e)	op(1, e)	[op(e)]
Mathcad			
Mathematica	Head[e]	e[[1]]	ReplacePart[e, List, 0]
Maxima	part(e, 0)	part(e, 1)	args(e)
MuPAD	op(e, 0)	op(e, 1)	[op(e)]
Octave			
Pari			
Reduce	part(e, 0)	part(e, 1)	for i:=1:arglength(e) collect part(e, i)
Singular			
Yacas			

⁸The following commands work only on expressions that consist of a single level (e.g., $x + y + z$ but not $a/b + c/d$).

⁹TERMS, FACTORS, NUMERATOR, LHS, etc.

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Print text and results	
Axiom	output(concat(["sin(", string(0), ") = ", string(sin(0))]);	
CoCoA	PrintLn "Does Sin exist?";	
Derive	"sin(0)" = sin(0)	
DoCon		
Giac	print("sin",0,") =",sin(0))	
GAP	Print("There is no sin, but factors(10)= ",Factors(10), "\n")	
Macsyma	print("sin(", 0, ") =", sin(0))\$	
Magnus		
Maple	printf("sin(%a) = %a\n", 0, sin(0));	
Mathcad		
Mathematica	Print[StringForm["sin(``) = ``", 0, Sin[0]]];	
Maxima	print("sin(", 0, ") =", sin(0))\$	
MuPAD	print(Unquoted, "sin(".0.)" = sin(0));	
Octave		
Pari		
Reduce	write("sin(", 0, ") = ", sin(0))\$	
Singular	"sin(",0,") =", sin(0);	
Yacas		
	Generate FORTRAN	Generate T _E X/L ^A T _E X
Axiom	outputAsFortran(e)	outputAsTex(e)
CoCoA		Latex(e);
Derive	[Transfer Save Fortran]	
DoCon		
GAP		Print(LaTeX(e));
Giac		C++ gen2tex
Macsyma	fortran(e)\$ or gentran(eval(e))\$	tex(e);
Magnus		
Maple	fortran([e]);	latex(e);
Mathcad		
Mathematica	FortranForm[e]	TexForm[e]
Maxima	fortran(e)\$ or gentran(eval(e))\$	tex(e);
MuPAD	generate::fortran(e);	generate::TeX(e);
Octave		
Pari		
Reduce	on fort; e; off fort; or load_package(gentran)\$ gentran e;	load_package(tri)\$ on TeX; e; off TeX;
Singular		LIB "latex.lib"; texobj(e);
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Import two space separated columns of integers from file
Axiom	
CoCoA	
Derive	[Transfer Load daTa] (from file.dat)
DoCon	
Giac	NA (C++)
GAP	
Macsyma	xy: read_num_data_to_matrix("file", nrows, 2)\$
Magnus	
Maple	xy:= readdata("file", integer, 2):
Mathcad	
Mathematica	xy = ReadList["file", Number, RecordLists -> True]
Maxima	xy: read_num_data_to_matrix("file", nrows, 2)\$
MuPAD	
Octave	
Pari	
Reduce	
Singular	
Yacas	

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Export two space separated columns of integers to file ¹⁰
Axiom	<pre>)set output algebra "file" (creates file.spout) for i in 1..n repeat output(_ concat([string(xy(i, 1)), " ", string(xy(i, 2))])))set output algebra console</pre>
Derive	<pre>xy [Transfer Print Expressions File] (creates file.prt)</pre>
CoCoA	
DoCon	
Giac	<pre>NA (C++)</pre>
GAP	<pre>PrintTo("file");for i in [1..n] do AppendTo("file",xy[i][1]," ",xy[i][2],"\n");od;</pre>
Macsyma	<pre>writefile("file")\$ for i:1 thru n do print(xy[i, 1], xy[i, 2])\$ closefile()\$</pre>
Magnus	
Maple	<pre>writedata("file", xy);</pre>
Mathcad	
Mathematica	<pre>outfile = OpenWrite["file"]; Do[WriteString[outfile, xy[[i, 1]], " ", xy[[i, 2]], "\n"], {i, 1, n}] Close[outfile];</pre>
Maxima	<pre>writefile("file")\$ for i:1 thru n do print(xy[i, 1], xy[i, 2])\$ closefile()\$</pre>
MuPAD	<pre>fprint(Unquoted, Text, "file", ("\n", xy[i, 1], xy[i, 2]) \$ i = 1..n):</pre>
Octave	
Pari	
Reduce	<pre>out "file"; for i:=1:n do write(xy(i, 1), " ", xy(i, 2)); shut "file";</pre>
Singular	<pre>link l=":w file";(open(l); for(int i=1;i<=n;i++) { write(l,xy[i,1]," ",xy[i,2]); } close(l);</pre>
Yacas	

1.4 Mathematics and Graphics

Since GAP aims at discrete mathematics, it does not provide much of the calculus functionality listed in the following section.

¹⁰Some editing of file will be necessary for all systems but Maple and Mathematica.

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	e	π	i	$+\infty$	$\sqrt{2}$	$2^{1/3}$
Axiom	%e	%pi	%i	%plusInfinity	sqrt(2)	2**(1/3)
CoCoA					Isqrt(2)	
Derive	#e	pi	#i	inf	SQRT(2)	2^(1/3)
DoCon						
GAP			E(4)	infinity	ER(2) ¹¹	
Giac	e	pi	i	+infinity	sqrt(2)	2^(1/3)
Macsyma	%e	%pi	%i	inf	sqrt(2)	2^(1/3)
Magnus						
Maple	exp(1)	Pi	I	infinity	sqrt(2)	2^(1/3)
Mathcad						
Mathematica	E	Pi	I	Infinity	Sqrt[2]	2^(1/3)
Maxima	%e	%pi	%i	inf	sqrt(2)	2^(1/3)
MuPAD	E	PI	I	infinity	sqrt(2)	2^(1/3)
Octave						
Pari						
Reduce	e	pi	i	infinity	sqrt(2)	2^(1/3)
Singular						
Yacas						

	Euler's constant	Natural log	Arctangent	$n!$
Axiom		log(x)	atan(x)	factorial(n)
CoCoA				
Derive	euler_gamma	LOG(x)	ATAN(x)	n!
DoCon				
Giac	-psi(1)	log(x)	atan(x)	factorial(n)
GAP		LogInt(x,base)		Factorial(n)
Macsyma	%gamma	log(x)	atan(x)	n!
Magnus				
Maple	gamma	log(x)	arctan(x)	n!
Mathcad				
Mathematica	EulerGamma	Log[x]	ArcTan[x]	n!
Maxima	%gamma	log(x)	atan(x)	n!
MuPAD	EULER	ln(x)	atan(x)	n!
Octave				
Pari				
Reduce	Euler_Gamma	log(x)	atan(x)	factorial(n)
Singular				
Yacas				

¹¹ER represents special cyclotomic numbers and is not a root function.

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Legendre polynomial	Chebyshev poly. of the 1 st kind
Axiom	legendreP(n, x)	chebyshevT(n, x)
CoCoA		
Derive	LEGENDRE_P(n, x)	CHEBYCHEV_T(n, x)
DoCon		
GAP		
Giac	legendre(n)	tchebyshev1(n)
Macsyma	legendre_p(n, x)	chebyshev_t(n, x)
Magnus		
Maple	orthopoly[P](n, x)	orthopoly[T](n, x)
Mathcad		
Mathematica	LegendreP[n, x]	ChebyshevT[n, x]
Maxima	legendre_p(n, x)	chebyshev_t(n, x)
MuPAD	orthopoly::legendre(n, x)	orthopoly::chebyshev1(n, x)
Octave		
Pari		
Reduce	LegendreP(n, x)	ChebyshevT(n, x)
Singular		
Yacas		
	Fibonacci number	Elliptic integral of the 1 st kind
Axiom	fibonacci(n)	
CoCoA		
Derive	FIBONACCI(n)	ELLIPTIC_E(phi, k^2)
DoCon		
GAP	Fibonacci(n)	
Giac		
Macsyma	fib(n)	elliptic_e(phi, k^2)
Magnus		
Maple	combinat[fibonacci](n)	EllipticE(sin(phi), k)
Mathcad		
Mathematica	Fibonacci[n]	EllipticE[phi, k^2]
Maxima	fib(n)	elliptic_e(phi, k^2)
MuPAD	numlib::fibonacci(n)	
Octave		
Pari		
Reduce		EllipticE(phi, k^2)
Singular	LIB "general.lib"; fibonacci(n);	
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	$\Gamma(x)$	$\psi(x)$	Cosine integral	Bessel fun. (1 st)
Axiom	Gamma(x)	psi(x)	real(Ei(%i*x))	besselJ(n, x)
CoCoA				
Derive	GAMMA(x)	PSI(x)	CI(x)	BESSEL_J(n, x)
DoCon				
GAP				
Giac	gamma(x)	psi(x)		
MacSyma	gamma(x)	psi[0](x)	cos_int(x)	bessel_j[n](x)
Magnus				
Maple	GAMMA(x)	Psi(x)	Ci(x)	BesselJ(n, x)
Mathcad				
Mathematica	Gamma[x]	PolyGamma[x]	CosIntegral[x]	BesselJ[n, x]
Maxima	gamma(x)	psi[0](x)	cos_int(x)	bessel_j[n](x)
MuPAD	gamma(x)	psi(x)		besselJ(n, x)
Octave				
Pari				
Reduce	Gamma(x)	Psi(x)	Ci(x)	BesselJ(n, x)
Singular				
Yacas				
	Hypergeometric fun. ${}_2F_1(a, b; c; x)$		Dirac delta	Unit step fun.
Axiom				
CoCoA				
Derive	GAUSS(a, b, c, x)			STEP(x)
DoCon				
GAP				
Giac				
MacSyma	hgfred([a, b], [c], x)		delta(x)	unit_step(x)
Magnus				
Maple	hypergeom([a, b], [c], x)		Dirac(x)	Heaviside(x)
Mathcad				
Mathematica	HypergeometricPFQ[{a,b},{c},x]		<< Calculus`DiracDelta`	
Maxima	hgfred([a, b], [c], x)		delta(x)	unit_step(x)
MuPAD			dirac(x)	heaviside(x)
Octave				
Pari				
Reduce	hypergeometric({a, b}, {c}, x)			
Singular				
Yacas				

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Define $ x $ via a piecewise function	
Axiom		
CoCoA		
Derive	$a(x) := -x \cdot \text{CHI}(-\infty, x, 0) + x \cdot \text{CHI}(0, x, \infty)$	
DoCon		
GAP		
Giac		
Macsyma	$a(x) := -x \cdot \text{unit_step}(-x) + x \cdot \text{unit_step}(x)$	
Magnus		
Maple	$a := x \rightarrow \text{piecewise}(x < 0, -x, x):$	
Mathcad		
Mathematica	<pre><< Calculus`DiracDelta` a[x_]:= -x*UnitStep[-x] + x*UnitStep[x]</pre>	
Maxima	$a(x) := -x \cdot \text{unit_step}(-x) + x \cdot \text{unit_step}(x)$	
MuPAD	<pre>a:= proc(x) begin -x*heaviside(-x) + x*heaviside(x) end_proc:</pre>	
Octave		
Pari		
Reduce		
Singular		
Yacas		
	Assume x is real	Remove that assumption
Axiom		
CoCoA		
Derive	$x : \text{epsilon Real}$	$x :=$
DoCon		
GAP		
Giac	$\text{assume}(x, \text{real})$	$\text{purge}(x)$
Macsyma	$\text{declare}(x, \text{real})$	$\text{remove}(x, \text{real})$
Magnus		
Maple	$\text{assume}(x, \text{real});$	$x := 'x':$
Mathcad		
Mathematica	$x/: \text{Im}[x] = 0;$	$\text{Clear}[x]$
Maxima	$\text{declare}(x, \text{real})$	$\text{remove}(x, \text{real})$
MuPAD	$\text{assume}(x, \text{Type}::\text{RealNum}):$	$\text{unassume}(x, \text{Type}::\text{RealNum}):$
Octave		
Pari		
Reduce		
Singular		
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Assume $0 < x \leq 1$	Remove that assumption
Axiom		
CoCoA		
Derive	<code>x :epsilon (0, 1]</code>	<code>x:=</code>
DoCon		
GAP		
Giac	<code>assume(x>0 && x<=1)</code>	<code>purge(x)</code>
Macysma	<code>assume(x > 0, x <= 1)\$</code>	<code>forget(x > 0, x <= 1)\$</code>
Magnus		
Maple	<code>assume(x > 0);</code> <code>additionally(x <= 1);</code>	<code>x:= 'x':</code>
Mathcad		
Mathematica	<code>Assumptions -> 0 < x <= 1¹²</code>	
Maxima	<code>assume(x > 0, x <= 1)\$</code>	<code>forget(x > 0, x <= 1)\$</code>
MuPAD	<code>assume(x > 0): assume(x <= 1):</code>	<code>unassume(x):</code>
Octave		
Pari		
Reduce		
Singular		
Yacas		
	Basic simplification of an expression e	
Axiom	<code>simplify(e) or normalize(e) or complexNormalize(e)</code>	
CoCoA		
Derive	<code>e</code>	
DoCon		
GAP	<code>e</code>	
Giac	<code>simplify(e)</code>	
Macysma	<code>ratsimp(e) or radcan(e)</code>	
Magnus		
Maple	<code>simplify(e)</code>	
Mathcad		
Mathematica	<code>Simplify[e] or FullSimplify[e]</code>	
Maxima	<code>ratsimp(e) or radcan(e)</code>	
MuPAD	<code>simplify(e) or normal(e)</code>	
Octave		
Pari		
Reduce	<code>e</code>	
Singular		
Yacas		

¹²This is an option for `Integrate`.

	Use an unknown function	Numerically evaluate an expr.
Axiom	<code>f := operator('f); f(x)</code>	<code>exp(1) :: Complex Float</code>
CoCoA		
Derive	<code>f(x) := f(x)</code>	<code>Precision := Approximate APPROX(EXP(1)) Precision := Exact</code>
DoCon		
Giac	<code>f(x)</code>	<code>evalf(exp(1))</code>
GAP		<code>EvalF(123/456)</code>
Macsyma	<code>f(x)</code>	<code>sfloat(exp(1));</code>
Magnus		
Maple	<code>f(x)</code>	<code>evalf(exp(1));</code>
Mathcad		
Mathematica	<code>f[x]</code>	<code>N[Exp[1]]</code>
Maxima	<code>f(x)</code>	<code>sfloat(exp(1));</code>
MuPAD	<code>f(x)</code>	<code>float(exp(1));</code>
Octave		
Pari		
Reduce	<code>operator f; f(x)</code>	<code>on rounded; exp(1); off rounded;</code>
Singular		
Yacas		
	$n \bmod m$	Solve $e \equiv 0 \pmod m$ for x
Axiom	<code>rem(n, m)</code>	<code>solve(e = 0 :: PrimeField(m), x)</code>
CoCoA		
Derive	<code>MOD(n, m)</code>	<code>SOLVE_MOD(e = 0, x, m)</code>
DoCon		
GAP	<code>n mod m</code>	solve using finite fields
Giac	<code>n % m</code>	
Macsyma	<code>mod(n, m)</code>	<code>modulus: m\$ solve(e = 0, x)</code>
Magnus		
Maple	<code>n mod m</code>	<code>msolve(e = 0, m)</code>
Mathcad		
Mathematica	<code>Mod[n, m]</code>	<code>Solve[{e == 0, Modulus == m}, x]</code>
Maxima	<code>mod(n, m)</code>	<code>modulus: m\$ solve(e = 0, x)</code>
MuPAD	<code>n mod m</code>	<code>solve(poly(e = 0, [x], IntMod(m)), x)</code>
Octave		
Pari		
Reduce	<code>on modular; setmod m\$ n</code>	<code>load_package(modsr)\$ on modular; setmod m\$ m_solve(e = 0, x)</code>
Singular	<code>n mod m</code>	
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Put over common denominator	Expand into separate fractions
Axiom	$a/b + c/d$	$(a*d + b*c)/(b*d) :: _$ MPOLY([a], FRAC POLY INT)
CoCoA		
Derive	FACTOR(a/b + c/d, Trivial)	EXPAND((a*d + b*c)/(b*d))
DoCon		
GAP	$a/b+c/d$	
Giac	normal(a/b + c/d)	
Macysma	xthru(a/b + c/d)	expand((a*d + b*c)/(b*d))
Magnus		
Maple	normal(a/b + c/d)	expand((a*d + b*c)/(b*d))
Mathcad		
Mathematica	Together[a/b + c/d]	Apart[(a*d + b*c)/(b*d)]
Maxima	xthru(a/b + c/d)	expand((a*d + b*c)/(b*d))
MuPAD	normal(a/b + c/d)	expand((a*d + b*c)/(b*d))
Octave		
Pari		
Reduce	$a/b + c/d$	on div; (a*d + b*c)/(b*d)
Singular	$a/b + c/d$	(a*d + b*c)/(b*d)
Yacas		
	Manipulate the root of a polynomial	
Axiom	a:= rootOf(x**2 - 2); a**2	
CoCoA		
Derive		
DoCon		
GAP	x:=X(Rationals,"x"); a:=RootOfDefiningPolynomial(AlgebraicExtension(Rationals,x^2-2)); a^2	
Giac	x:=rootof([1,0],[1,0,-2])	
Macysma	algebraic:true\$ tellrat(a^2 - 2)\$ rat(a^2);	
Magnus		
Maple	a:= RootOf(x^2 - 2): simplify(a^2);	
Mathcad		
Mathematica	a = Root[#^2 - 2 &, 2] a^2	
Maxima	algebraic:true\$ tellrat(a^2 - 2)\$ rat(a^2);	
MuPAD		
Octave		
Pari		
Reduce	load_package(arnum)\$ defpoly(a^2 - 2); a^2;	
Singular		
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Noncommutative multiplication	Solve a pair of equations
Axiom		<code>solve([eqn1, eqn2], [x, y])</code>
CoCoA		
Derive	<code>x :epsilon Nonscalar</code> <code>y :epsilon Nonscalar</code> <code>x . y</code>	<code>SOLVE([eqn1, eqn2], [x, y])</code>
DoCon		
Giac	NA	<code>solve([eqn1, eqn2], [x, y])</code>
GAP	*	
Macsyma	<code>x . y</code>	<code>solve([eqn1, eqn2], [x, y])</code>
Magnus		
Maple	<code>x &* y</code>	<code>solve({eqn1, eqn2}, {x, y})</code>
Mathcad		
Mathematica	<code>x ** y</code>	<code>Solve[{eqn1, eqn2}, {x, y}]</code>
Maxima	<code>x . y</code>	<code>solve([eqn1, eqn2], [x, y])</code>
MuPAD		<code>solve({eqn1, eqn2}, {x, y})</code>
Octave		
Pari		
Reduce	<code>operator x, y;</code> <code>noncom x, y;</code> <code>x() * y()</code>	<code>solve({eqn1, eqn2}, {x, y})</code>
Singular		<code>LIB "solve.lib";</code> <code>solve(ideal(eqn1,eqn2));</code>
Yacas		
	Decrease/increase angles in trigonometric functions	
Axiom	<code>simplify(normalize(sin(2*x)))</code>	
CoCoA		
Derive	<code>Trigonometry:= Expand</code> <code>sin(2*x)</code>	<code>Trigonometry:= Collect</code> <code>2*sin(x)*cos(x)</code>
DoCon		
GAP		
Giac	<code>texpand(sin(2*x))</code>	<code>tlin(2*sin(x)*cos(x))</code>
Macsyma	<code>trigexpand(sin(2*x))</code>	<code>trigreduce(2*sin(x)*cos(x))</code>
Magnus		
Maple	<code>expand(sin(2*x))</code>	<code>combine(2*sin(x)*cos(x))</code>
Mathcad		
Mathematica	<code>TrigExpand[Sin[2*x]]</code>	<code>TrigReduce[2*Sin[x]*Cos[x]]</code>
Maxima	<code>trigexpand(sin(2*x))</code>	<code>trigreduce(2*sin(x)*cos(x))</code>
MuPAD	<code>expand(sin(2*x))</code>	<code>combine(2*sin(x)*cos(x), sincos)</code>
Octave		
Pari		
Reduce	<code>load_package(assist)\$</code> <code>trigexpand(sin(2*x))</code>	<code>trigreduce(2*sin(x)*cos(x))</code>
Singular		
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Gröbner basis
Axiom	groebner([p1, p2, ...])
CoCoA	GBasis(Ideal(p1, p2, ...));
Derive	
DoCon	
GAP	
Giac	gbasis([p1,p2,...],[x,y,...])
Macsyma	grobner([p1, p2, ...])
Magnus	
Maple	Groebner[gbasis]([p1, p2, ...], plex(x1, x2, ...))
Mathcad	
Mathematica	GroebnerBasis[{p1, p2, ...}, {x1, x2, ...}]
Maxima	grobner([p1, p2, ...])
MuPAD	groebner::gbasis([p1, p2, ...])
Octave	
Pari	
Reduce	load_package(groebner)\$ groebner({p1, p2, ...})
Singular	groebner(ideal(p1,p2 ...))
Yacas	
	Factorization of e over $i = \sqrt{-1}$
Axiom	factor(e, [rootOf(i**2 + 1)])
CoCoA	
Derive	FACTOR(e, Complex)
DoCon	
GAP	Factors(GaussianIntegers,e)
Giac	factor(e) in complex mode
Macsyma	gfactor(e); or factor(e, i^2 + 1);
Magnus	
Maple	factor(e, I);
Mathcad	
Mathematica	Factor[e, Extension -> I]
Maxima	gfactor(e); or factor(e, i^2 + 1);
MuPAD	QI:= Dom::AlgebraicExtension(Dom::Rational, i^2 + 1); QI::name:= "QI": Factor(poly(e, QI));
Octave	
Pari	
Reduce	on complex, factor; e; off complex, factor;
Singular	ring C=(0,i),x,dp;minpoly=i2+1;factorize(e);
Yacas	

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Real part	Convert a complex expr. to rectangular form	
Axiom	<code>real(f(z))</code>	<code>complexForm(f(z))</code>	
CoCoA			
Derive	<code>RE(f(z))</code>	<code>f(z)</code>	
DoCon			
GAP	<code>(f(z)+GaloisCyc(f(z),-1))/2</code>		
Giac	<code>re(f(z))</code>	<code>re(f(z))+i*im(f(z))</code>	
Macsyma	<code>realpart(f(z))</code>	<code>rectform(f(z))</code>	
Magnus			
Maple	<code>Re(f(z))</code>	<code>evalc(f(z))</code>	
Mathcad			
Mathematica	<code>Re[f[z]]</code>	<code>ComplexExpand[f[z]]</code>	
Maxima	<code>realpart(f(z))</code>	<code>rectform(f(z))</code>	
MuPAD	<code>Re(f(z))</code>	<code>rectform(f(z))</code>	
Octave			
Pari			
Reduce	<code>repart(f(z))</code>	<code>repart(f(z)) + i*impart(f(z))</code>	
Singular	<code>repart(f(z)) * repart(f(z)) + i*impart(f(z))</code>		
Yacas			

	Matrix addition	Matrix multiplication	Matrix transpose
Axiom	<code>A + B</code>	<code>A * B</code>	<code>transpose(A)</code>
CoCoA	<code>A + B;</code>	<code>A * B;</code>	<code>Transposed(A);</code>
Derive	<code>A + B</code>	<code>A . B</code>	<code>A`</code>
DoCon			
GAP	<code>A + B</code>	<code>A * B</code>	<code>TransposedMat(A)</code>
Giac	<code>A + B</code>	<code>A * B</code>	<code>tran(A)</code>
Macsyma	<code>A + B</code>	<code>A . B</code>	<code>transpose(A)</code>
Magnus			
Maple	<code>evalm(A + B)</code>	<code>evalm(A &* B)</code>	<code>linalg[transpose](A)</code>
Mathcad			
Mathematica	<code>A + B</code>	<code>A . B</code>	<code>Transpose[A]</code>
Maxima	<code>A + B</code>	<code>A . B</code>	<code>transpose(A)</code>
MuPAD	<code>A + B</code>	<code>A * B</code>	<code>transpose(A)</code>
Octave			
Pari			
Reduce	<code>A + B</code>	<code>A * B</code>	<code>tp(A)</code>
Singular	<code>A + B</code>	<code>A * B</code>	<code>transpose(A)</code>
Yacas			

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Solve the matrix equation $Ax = b$	
Axiom	solve(A, transpose(b)) . 1 . particular :: Matrix ___	
CoCoA		
Derive		
DoCon		
GAP	SolutionMat(TransposedMat(A),b)	
Giac	NA rref(tran(concat(tran(A),[b]))) or inv(A)*b	
Macysma	xx: genvector('x, mat_nrows(b))\$ x: part(matlinsolve(A . xx = b, xx), 1, 2)	
Magnus		
Maple	x:= linalg[linsolve](A, b)	
Mathcad		
Mathematica	x = LinearSolve[A, b]	
Maxima	xx: genvector('x, mat_nrows(b))\$ x: part(matlinsolve(A . xx = b, xx), 1, 2)	
MuPAD		
Octave		
Pari		
Reduce		
Singular		
Yacas		
	Sum: $\sum_{i=1}^n f(i)$	Product: $\prod_{i=1}^n f(i)$
Axiom	sum(f(i), i = 1..n)	product(f(i), i = 1..n)
CoCoA	L:=[]; Foreach I In 1..N Do Append(L,F(I)); End; Sum(L);	L:=[]; Foreach I in 1..N Do Append(L,F(I)); End; Product(L);
Derive	SUM(f(i), i, 1, n)	PRODUCT(f(i), i, 1, n)
DoCon		
GAP	Sum([1..n],f)	Product([1..n],f)
Giac	sum(1/j,j,1..n)	NA
Macysma	closedform(sum(f(i), i, 1, n))	closedform(product(f(i), i, 1, n))
Magnus		
Maple	sum(f(i), i = 1..n)	product(f(i), i = 1..n)
Mathcad		
Mathematica	Sum[f[i], {i, 1, n}]	Product[f[i], {i, 1, n}]
Maxima	closedform(sum(f(i), i, 1, n))	closedform(product(f(i), i, 1, n))
MuPAD	sum(f(i), i = 1..n)	product(f(i), i = 1..n)
Octave		
Pari		
Reduce	sum(f(i), i, 1, n)	prod(f(i), i, 1, n)
Singular		
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Limit: $\lim_{x \rightarrow 0^-} f(x)$	Taylor/Laurent/etc. series
Axiom	limit(f(x), x = 0, "left")	series(f(x), x = 0, 3)
CoCoA		
Derive	LIM(f(x), x, 0, -1)	TAYLOR(f(x), x, 0, 3)
DoCon		
GAP		
Giac	limit(f(x), x=0, -1)	series(f(x), x=0, 3)
Macysma	limit(f(x), x, 0, minus)	taylor(f(x), x, 0, 3)
Magnus		
Maple	limit(f(x), x = 0, left)	series(f(x), x = 0, 4)
Mathcad		
Mathematica	Limit[f[x], x->0, Direction->1]	Series[f[x], {x, 0, 3}]
Maxima	limit(f(x), x, 0, minus)	taylor(f(x), x, 0, 3)
MuPAD	limit(f(x), x = 0, Left)	series(f(x), x = 0, 4)
Octave		
Pari		
Reduce	limit!-(f(x), x, 0)	taylor(f(x), x, 0, 3)
Singular		
Yacas		
	Differentiate: $\frac{d^3 f(x,y)}{dx dy^2}$	Integrate: $\int_0^1 f(x) dx$
Axiom	D(f(x, y), [x, y], [1, 2])	integrate(f(x), x = 0..1)
CoCoA		
Derive	DIF(DIF(f(x, y), x), y, 2)	INT(f(x), x, 0, 1)
DoCon		
GAP		
Giac	diff(f(x,y), [x,y], [1,2])	integrate(f(x), x, 0, 1)
Macysma	diff(f(x, y), x, 1, y, 2)	integrate(f(x), x, 0, 1)
Magnus		
Maple	diff(f(x, y), x, y\$2)	int(f(x), x = 0..1)
Mathcad		
Mathematica	D[f[x, y], x, {y, 2}]	Integrate[f[x], {x, 0, 1}]
Maxima	diff(f(x, y), x, 1, y, 2)	integrate(f(x), x, 0, 1)
MuPAD	diff(f(x, y), x, y\$2)	int(f(x), x = 0..1)
Octave		
Pari		
Reduce	df(f(x, y), x, y, 2)	int(f(x), x, 0, 1)
Singular	diff(diff(diff(f,x),y),y)	
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Laplace transform	Inverse Laplace transform
Axiom	laplace(e, t, s)	inverseLaplace(e, s, t)
CoCoA		
Derive	LAPLACE(e, t, s)	
DoCon		
GAP		
Giac	laplace(e,t,s)	ilaplace(e,s,t)
Macsyma	laplace(e, t, s)	ilt(e, s, t)
Magnus		
Maple	inttrans[laplace](e,t,s)	inttrans[invlaplace](e,s,t)
Mathcad		
Mathematica	<< Calculus`LaplaceTransform` LaplaceTransform[e, t, s]	InverseLaplaceTransform[e,s,t]
Maxima	laplace(e, t, s)	ilt(e, s, t)
MuPAD	transform::laplace(e,t,s)	transform::ilaplace(e, s, t)
Octave		
Pari		
Reduce	load_package(laplace)\$ laplace(e, t, s)	load_package(defint)\$ invlap(e, t, s)
Singular		
Yacas		
	Solve an ODE (with the initial condition $y'(0) = 1$)	
Axiom	solve(eqn, y, x)	
CoCoA		
Derive	APPLY_IC(RHS(ODE(eqn, x, y, y-)), [x, 0], [y, 1])	
DoCon		
GAP		
Giac	desolve([y'+y=sin(x),y'(0)=1],x,y)	
Macsyma	ode_ibc(ode(eqn, y(x), x), x = 0, diff(y(x), x) = 1)	
Magnus		
Maple	dsolve({eqn, D(y)(0) = 1}, y(x))	
Mathcad		
Mathematica	DSolve[{eqn, y'[0] == 1}, y[x], x]	
Maxima	ode_ibc(ode(eqn, y(x), x), x = 0, diff(y(x), x) = 1)	
MuPAD	solve(ode({eqn, D(y)(0) = 1}, y(x)))	
Octave		
Pari		
Reduce	odesolve(eqn, y(x), x)	
Singular		
Yacas		

Based on material originally published in *Computer Algebra Systems: A Practical Guide* edited by Michael J. Wester, John Wiley & Sons, Chichester, United Kingdom, ISBN 0-471-98353-5, xvi+436 pages, 1999.

	Define the differential operator $L = D_x + I$ and apply it to $\sin x$
Axiom	<code>DD : LOD0(Expression Integer, e --> D(e, x)) := D();</code> <code>L:= DD + 1; L(sin(x))</code>
CoCoA	
Derive	
DoCon	
GAP	
Giac	NA
Macsyma	<code>load(opalg)\$ L: (diffop(x) - 1)\$ L(sin(x));</code>
Magnus	
Maple	<code>id:= x -> x: L:= (D + id): L(sin)(x);</code>
Mathcad	
Mathematica	<code>L = D[#, x]& + Identity; Through[L[Sin[x]]]</code>
Maxima	<code>load(opalg)\$ L: (diffop(x) - 1)\$ L(sin(x));</code>
MuPAD	<code>L:= (D + id): L(sin)(x);</code>
Octave	
Pari	
Reduce	
Singular	
Yacas	
	2D plot of two separate curves overlaid
Axiom	<code>draw(x, x = 0..1); draw(acsch(x), x = 0..1);</code>
CoCoA	
Derive	<code>[Plot Overlay]</code>
DoCon	
GAP	
Giac	<code>plotfunc(x,x); plotfunc(sin(x),x);</code>
Macsyma	<code>plot(x, x, 0, 1)\$ plot(acsch(x), x, 0, 1)\$</code>
Magnus	
Maple	<code>plot({x, arccsch(x)}, x = 0..1):</code>
Mathcad	
Mathematica	<code>Plot[{x, ArcCsch[x]}, {x, 0, 1}];</code>
Maxima	<code>plot(x, x, 0, 1)\$ plot(acsch(x), x, 0, 1)\$</code>
MuPAD	<code>plotfunc(x, acsch(x), x = 0..1):</code>
Octave	
Pari	
Reduce	<code>load_package(gnuplot)\$ plot(y = x, x = (0 .. 1))\$</code> <code>plot(y = acsch(x), x = (0 .. 1))\$</code>
Singular	<code>LIB "surf.lib";</code> <code>plot(((x+3)³+2*(x+3)²-y²)*(x³-y²)*((x-3)³-2*(x-3)²-y²));¹³</code>
Yacas	

¹³Singular can only plot polynomial objects

	Simple 3D plotting
Axiom	<code>draw(abs(x*y), x = 0..1, y = 0..1);</code>
CoCoA	
Derive	<code>[Plot Overlay]</code>
DoCon	
GAP	
Giac	<code>plotfunc(abs(x*y), [x,y], [0,1], [0,1])</code>
Macsyma	<code>plot3d(abs(x*y), x, 0, 1, y, 0, 1)\$</code>
Magnus	
Maple	<code>plot3d(abs(x*y), x = 0..1, y = 0..1):</code>
Mathcad	
Mathematica	<code>Plot3D[Abs[x*y], {x, 0, 1}, {y, 0, 1}];</code>
Maxima	<code>plot3d(abs(x*y), x, 0, 1, y, 0, 1)\$</code>
MuPAD	<code>plotfunc(abs(x*y), x = 0..1, y = 0..1):</code>
Octave	
Pari	
Reduce	<code>load_package(gnuplot)\$</code> <code>plot(z = abs(x*y), x = (0 .. 1), y = (0 .. 1))\$</code>
Singular	<code>LIB "surf.lib"; plot(z²-x²*y);¹⁴</code>
Yacas	

¹⁴Singular can only plot polynomial objects