1 Introduction

The following is a collection of synonyms for various operations in the computer algebra systems Axiom, Derive, GAP, Gmp, DoCon, Macsyma, Magnus, Maxima, Maple, Mathematica, MuPAD, Octave, Pari, Reduce, Scilab, Sumit 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.

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.

Originally written by Michael Wester. Modified for Rosetta by Timothy Daly, Alexander Hulpke (GAP).

2 System availability

System	License	Status (May 2002)	Web Location
Axiom	BSD	available	http://www.aldor.org
Axiom	open source	pending	http://home.earthlink.net/ jgg964/axiom.html
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
Gmp	GPL	Rosetta	http://www.swox.com/gmp
Macsyma	commercial	dead	unavailable
Magnus	GPL	Rosetta	http://sourceforge.net/projects/magnus
Maxima	GPL	Rosetta	http://www.ma.utexas.edu/maxima.html
Maple	commercial	available	http://www.maplesoft.com
Mathematica	commercial	available	http://www.wolfram.com
MuPAD	commercial	available	http://www.mupad.de
Octave	GPL	Rosetta	http://www.octave.org
Pari	GPL	Rosetta	http://www.parigp-home.de
Reduce	commercial	available	http://www.zib.de/Symbolik/reduce
Scilab	Scilab	available	http://www-rocq.inria.fr/scilab
Sumit		available	http://www-sop.inria.fr/cafe/soft-f.html
Yacas	GPL	available	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 or Compiled
Axiom	General Purpose	both
Derive	General Purpose	
DoCon	General Purpose	Interpreted in Haskell
GAP	Group Theory	
Gmp	arb. prec. arithmetic	
Macsyma	General Purpose	
Magnus	Infinite Group Theory	
Maxima	General Purpose	
Maple	General Purpose	
Mathematica	General Purpose	
MuPAD	General Purpose	
Octave	Numerical Computing	
Pari	Number Theory	
Reduce	General Purpose	
Scilab	General Purpose	
Sumit	Functional Equations	
Yacas	General Purpose	

3 Programming and Miscellaneous

	Unix/Microsoft user initialization file	
Axiom	`~/axiom.input`	
GAP	`~/.gaprc`	`GAP.RC`
Gmp		
DoCon		
Derive		`derive.ini`
Macsyma	`~/macsyma-init.macsyma`	`mac-init.mac`
Magnus		
Maxima	`~/macsyma-init.macsyma`	`mac-init.mac`
Maple	`~/.mapleinit`	`maplev5.ini`
Mathematica	`~/init.m`	`init.m`
MuPAD	`~/.mupadinit`	`\mupad\bin\userinit.mu`
Octave		
Pari		
Reduce	`~/.reducerc`	`reduce.rc`
Scilab		
Sumit		
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.

	Describe *keyword*	Find keywords containing *pattern*
Axiom		)what operations pattern
Derive		
DoCon		
GAP	?keyword	??keyword
Gmp		
Macsyma	describe("keyword")$	apropos("pattern");
Magnus		
Maxima	describe("keyword")$	apropos("pattern");
Maple	?keyword	?pattern [1]
Mathematica	?keyword	?*pattern*
MuPAD	?keyword	?*pattern*
Octave	help -i keyword	
Pari		
Reduce		
Scilab		
Sumit		
Yacas		

	Comment	Line continuation	Prev. expr.	Case sensitive	Variables assumed
Axiom	-- comment	input _<CR>input	%	Yes	real
Derive	"comment"	input ~<CR>input		No	real
DoCon					
GAP	# comment	input\<CR>input	last	Yes	no assumption
Gmp					
Macsyma	/* comment */	input<CR>input;	%	No	real
Magnus					
Maxima	/* comment */	input<CR>input;	%	No	real
Maple	# comment	input<CR>input;	%	Yes	complex
Mathematica	(* comment *)	input<CR>input	%	Yes	complex
MuPAD	# comment #	input<CR>input;	%	Yes	complex
Octave	##			Yes	
Pari					
Reduce	% comment	input<CR>input;	ws	No	complex
Scilab					
Sumit					
Yacas					

[1] Only 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.

	Load a file	Time a command	Quit
Axiom	`)read "file" )quiet`	`)set messages time on`	`)quit`
Derive	`[Transfer Load Derive]`		`[Quit]`
DoCon			
GAP	`Read("file");`	`time;` (also see `Runtime();`)	`quit;`
Gmp			
Macsyma	`load("file")$`	`showtime: all$`	`quit();`
Magnus			
Maxima	`load("file")$`	`showtime: all$`	`quit();`
Maple	`read("file"):`	`readlib(showtime): on;`	`quit`
Mathematica	`@<< file`	`Timing[command]`	`Quit[]`
MuPAD	`read("file"):`	`time(command);`	`quit`
Octave	`load file`	`tic(); cmd ; toc()`	`quit` *or* `exit`
Pari			
Reduce	`in "file"$`	`on time;`	`quit;`
Scilab			`quit`
Sumit			
Yacas			

	Display output	Suppress output	Substitution: $f(x,y) \to f(z,w)$
Axiom	`input`	`input;`	`subst(f(x, y), [x = z, y = w])`
Derive	`input`	`var:= input`	`[Manage Substitute]`
DoCon			
GAP	`input;`	`input;;`	`Value(f,[x,y],[z,w]);`[2]
Gmp			
Macsyma	`input;`	`input$`	`subst([x = z, y = w], f(x, y));`
Magnus			
Maxima	`input;`	`input$`	`subst([x = z, y = w], f(x, y));`
Maple	`input;`	`input:`	`subs({x = z, y = w}, f(x, y));`
Mathematica	`input`	`input;`	`f[x, y] /. {x -> z, y -> w}`
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));`
Scilab			
Sumit			
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.

	Set	List	Matrix
Axiom	set [1, 2]	[1, 2]	matrix(@[[1, 2],[3, 4]])
Derive	{1, 2}	[1, 2]	@[[1,2], [3,4]]
DoCon			
GAP	Set([1,2])	[1, 2]	@[[1,2], [3,4]][3]
Gmp			
Macsyma	[1, 2]	[1, 2]	matrix([1, 2], [3, 4])
Magnus			
Maxima	[1, 2]	[1, 2]	matrix([1, 2], [3, 4])
Maple	{1, 2}	[1, 2]	matrix(@[[1, 2], [3, 4]])
Mathematica	{1, 2}	{1, 2}	{{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))
Scilab		list(1,2)	A=[1,2;3,4]
Sumit			
Yacas			

	Equation	List element	Matrix element	Length of a list
Axiom	x = 0	l . 2	m(2, 3)	#l
Derive	x = 0	l SUB 2	m SUB 2 SUB 3	DIMENSION(l)
DoCon				
GAP	x=0	l[2]	m[2][3]	Length(l)
Gmp				
Macsyma	x = 0	l[2]	m[2, 3]	length(l)
Magnus				
Maxima	x = 0	l[2]	m[2, 3]	length(l)
Maple	x = 0	l[2]	m[2, 3]	nops(l)
Mathematica	x == 0	l@[[2]]	m@[[2, 3]]	Length[l]
MuPAD	x = 0	l[2]	m[2, 3]	nops(l)
Octave				
Pari				
Reduce	x = 0	part(l, 2)	m(2, 3)	length(l)
Scilab		l(2)		
Sumit				
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.

	Prepend/append an element to a list		Append two lists
Axiom	`cons(e, l)`	`concat(l, e)`	`append(l1, l2)`
Derive	`APPEND([e], l)`	`APPEND(l, [e])`	`APPEND(l1, l2)`
DoCon			
GAP	`Concatenation([e],l)`	`Add(l,e)`	`Append(l1, l2)`
Gmp			
Macsyma	`cons(e, l)`	`endcons(e, l)`	`append(l1, l2)`
Magnus			
Maxima	`cons(e, l)`	`endcons(e, l)`	`append(l1, l2)`
Maple	`[e, op(l)]`	`[op(l), e]`	`[op(l1), op(l2)]`
Mathematica	`Prepend[l, e]`	`Append[l, e]`	`Join[l1, l2]`
MuPAD	`[e, op(l)]`	`append(l, e)`	`l1 . l2`
Octave			
Pari			
Reduce	`e . l`	`append(l, e)`	`append(l1, l2)`
Scilab			
Sumit			
Yacas			

	Matrix column dimension	Convert a list into a column vector
Axiom	`ncols(m)`	`transpose(matrix([l]))`
Derive	`DIMENSION(m SUB 1)`	``[l]` ``
DoCon		
GAP	`Length(mat[1])`	objects are identical
Gmp		
Macsyma	`mat_ncols(m)`	`transpose(matrix(l))`
Magnus		
Maxima	`mat_ncols(m)`	`transpose(matrix(l))`
Maple	`linalg[coldim](m)`	`linalg[transpose](matrix([l]))`
Mathematica	`Dimensions[m]@[[2]]`	`Transpose[{l}]`
MuPAD	`linalg::ncols(m)`	`transpose(matrix([l]))` [4]
Octave		
Pari		

Reduce	`load_package(linalg)$`
	`column_dim(m)`
		`matrix v(length(l), 1)$`
		`for i:=1:length(l) do`
		`    v(i, 1):= part(l, i)`

Scilab		
Sumit		
Yacas		

[4] See the definition of `matrix` above.

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.

	Convert a column vector into a list
Axiom	`[v(i, 1) for i in 1..nrows(v)]`
Derive	``v` SUB 1``
DoCon	
GAP	objects are identical
Gmp	
Macsyma	`part(transpose(v), 1)`
Magnus	
Maxima	`part(transpose(v), 1)`
Maple	`op(convert(linalg[transpose](v), listlist))`
Mathematica	`Flatten[v]`
MuPAD	`[op(v)]`
Octave	
Pari	
Reduce	`load_package(linalg)$` `for i:=1:row_dim(v) collect(v(i, 1))`
Scilab	
Sumit	
Yacas	

	True	False	And	Or	Not	Equal	Not equal
Axiom	`true`	`false`	`and`	`or`	`not`	`=`	`~=`
Derive	`TRUE`	`FALSE`	`AND`	`OR`	`NOT`	`=`	`/=`
DoCon							
GAP	`true`	`false`[5]	`and`	`or`	`not`	`=`	`<>`
Gmp							
Macsyma	`true`	`false`	`and`	`or`	`not`	`=`	`#`
Magnus							
Maxima	`true`	`false`	`and`	`or`	`not`	`=`	`#`
Maple	`true`	`false`	`and`	`or`	`not`	`=`	`<>`
Mathematica	`True`	`False`	`&&`	`||`	`!`	`==`	`!=`
MuPAD	`true`	`false`	`and`	`or`	`not`	`=`	`<>`
Octave							
Pari							
Reduce	`t`	`nil`	`and`	`or`	`not`	`=`	`neq`
Scilab	`%t`	`%f`					
Sumit							
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.

	If+then+else statements	Strings (concatenated)
Axiom	`if _ then _ else if _ then _ else _`	`concat(["x", "y"])`
Derive	`IF(_, _, IF(_, _, _))`	`"xy"`
DoCon		
GAP	`if _ then _ elif _ then _ else _ fi`	`Concatenation("x","y")`
Gmp		
Macsyma	`if _ then _ else if _ then _ else _`	`concat("x", "y")`
Magnus		
Maxima	`if _ then _ else if _ then _ else _`	`concat("x", "y")`
Maple	`if _ then _ elif _ then _ else _ fi`	`"x" . "y"`
Mathematica	`If[_, _, If[_, _, _]]`	`"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)`
Scilab		
Sumit		
Yacas		

	Simple loop and Block	Generate the list $[1, 2, \ldots, n]$
Axiom	`for i in 1..n repeat ( x; y )`	`[f(i) for i in 1..n]`
Derive	`VECTOR([x, y], i, 1, n)`	`VECTOR(f(i), i, 1, n)`
DoCon		
GAP	`for i in [1..n] do _ od;`	`[1..n]` or `[1,2..n]`
Gmp		
Macsyma	`for i:1 thru n do (x, y);`	`makelist(f(i), i, 1, n);`
Magnus		
Maxima	`for i:1 thru n do (x, y);`	`makelist(f(i), i, 1, n);`
Maple	`for i from 1 to n do x; y od;`	`[f(i) $ i = 1..n];`
Mathematica	`Do[x; y, {i, 1, n}]`	`Table[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);`
Scilab		
Sumit		
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)`
Derive	
DoCon	
GAP	`for x in [2, 3, 5] do while x^2<10 do Print(x);od;od;`
Gmp	
Macsyma	`for x in [2, 3, 5] while x^2 < 10 do print(x)$`
Magnus	
Maxima	`for x in [2, 3, 5] while x^2 < 10 do print(x)$`
Maple	`for x in [2, 3, 5] while x^2 < 10 do print(x) od:`
Mathematica	`For[l = {2, 3, 5}, l != {} && l@[[1]]^2 < 10,` `  l = Rest[l], Print[l@[[1]]] ]`
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)$`
Scilab	
Sumit	
Yacas	

	Assignment	Function definition	Clear vars and funs
Axiom	`y:= f(x)`	`f(x, y) == x*y`	`)clear properties y f`
Derive	`y:= f(x)`	`f(x, y):= x*y`	`y:= f:=`
DoCon			
GAP	`y:= f(x);`	`f:=function(x, y) return x*y; end;`	There are no symbolic variables
Gmp			
Macsyma	`y: f(x);`	`f(x, y):= x*y;`	`remvalue(y)$` `remfunction(f)$`
Magnus			
Maxima	`y: f(x);`	`f(x, y):= x*y;`	`remvalue(y)$` `remfunction(f)$`
Maple	`y:= f(x);`	`f:= proc(x, y) x*y end;`	`y:= 'y': f:= 'f':`
Mathematica	`y = f[x]`	`f[x_, y_]:= x*y`	`Clear[y, f]`
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;`
Scilab			
Sumit			
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)`
Derive	
DoCon	
GAP	`f:=function(x) local n; n:=2;return n*x; end;`
Gmp	
Macsyma	`f(x):= block([n], n: 2, n*x);`
Magnus	
Maxima	`f(x):= block([n], n: 2, n*x);`
Maple	`f:= proc(x) local n; n:= 2; n*x end;`
Mathematica	`f[x_]:= Module[{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;`
Scilab	
Sumit	
Yacas	

	Return unevaluated symbol	Define a function from an expression
Axiom	`e:= x*y;  'e`	`function(e, f, x, y)`
Derive	`e:= x*y  'e`	`f(x, y):== e`
DoCon		
GAP	No unevaluated symbols[6]	
Gmp		
Macsyma	`e: x*y$  'e;`	`define(f(x, y), e);`
Magnus		
Maxima	`e: x*y$  'e;`	`define(f(x, y), e);`
Maple	`e:= x*y:  'e';`	`f:= unapply(e, x, y);`
Mathematica	`e = x*y;  HoldForm[e]`	`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;`
Scilab		
Sumit		
Yacas		

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

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.

	Fun. of an indefinite number of args	Apply "+" to sum a list
Axiom		reduce(+, [1, 2])
Derive	LST l:= l	
DoCon		
GAP	lst:=function(args) _ end;	Sum([1,2])
Gmp		
Macsyma	lst([l]):= l;	apply("+", [1, 2])
Magnus		
Maxima	lst([l]):= l;	apply("+", [1, 2])
Maple	lst:=proc() [args[1..nargs]] end;	convert([1, 2], `+`)
Mathematica	lst[l___]:= {l}	Apply[Plus, {1, 2}]
MuPAD	lst:= proc(l) begin [args()] end_proc;	_plus(op([1, 2]))
Octave		
Pari		
Reduce		xapply(+, {1, 2}) [6]
Scilab		
Sumit		
Yacas		

	Apply a fun. to a list of its args	Map an anonymous function onto a list
Axiom	reduce(f, l)	map(x +-> x + y, [1, 2])
Derive		x:= [1, 2] VECTOR(x SUB i + y, i, 1, DIMENSION(x))
DoCon		
GAP	List(l,f)	List([1,2],x->x+y)
Gmp		
Macsyma	apply(f, l)	map(lambda([x], x + y), [1, 2])
Magnus		
Maxima	apply(f, l)	map(lambda([x], x + y), [1, 2])
Maple	f(op(l))	map(x -> x + y, [1, 2])
Mathematica	Apply[f, l]	Map[# + y &, {1, 2}]
MuPAD	f(op(l))	map([1, 2], func(x + y, x))
Octave		
Pari		
Reduce	xapply(f, l) [6]	for each x in {1, 2} collect x + y
Scilab		
Sumit		
Yacas		

[6] procedure xapply(f, lst); lisp(f . cdr(lst))$

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.

<table>
<tr><th></th><th>Pattern matching: $f(3y) + f(zy) \to 3f(y) + f(zy)$</th></tr>
<tr><td>Axiom</td><td>f:= operator('f);
(rule f((n | integer?(n)) * x) == n*f(x))(_
    f(3*y) + f(z*y))</td></tr>
<tr><td>Derive</td><td></td></tr>
<tr><td>DoCon</td><td></td></tr>
<tr><td>GAP</td><td></td></tr>
<tr><td>Gmp</td><td></td></tr>
<tr><td>Macsyma</td><td>matchdeclare(n, integerp, x, true)$
defrule(fnx, f(n*x), n*f(x))$
apply1(f(3*y) + f(z*y), fnx);</td></tr>
<tr><td>Magnus</td><td></td></tr>
<tr><td>Maxima</td><td>matchdeclare(n, integerp, x, true)$
defrule(fnx, f(n*x), n*f(x))$
apply1(f(3*y) + f(z*y), fnx);</td></tr>
<tr><td>Maple</td><td>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));</td></tr>
<tr><td>Mathematica</td><td>f[3*y] + f[z*y] /. f[n_Integer * x_] -> n*f[x]</td></tr>
<tr><td>MuPAD</td><td>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);</td></tr>
<tr><td>Octave</td><td></td></tr>
<tr><td>Pari</td><td></td></tr>
<tr><td>Reduce</td><td>operator f;
f(3*y) + f(z*y)
    where {f(~n * ~x) => n*f(x) when fixp(n)};</td></tr>
<tr><td>Scilab</td><td></td></tr>
<tr><td>Sumit</td><td></td></tr>
<tr><td>Yacas</td><td></td></tr>
</table>

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.

<table>
<tr><th></th><th>Define a new infix operator and then use it</th></tr>
<tr><td>Axiom</td><td></td></tr>
<tr><td>Derive</td><td></td></tr>
<tr><td>DoCon</td><td></td></tr>
<tr><td>GAP</td><td>One can overload existing infix operators for ones own purposes</td></tr>
<tr><td>Gmp</td><td></td></tr>
<tr><td>Macsyma</td><td><code>infix("~")$ "~"(x, y):= sqrt(x^2 + y^2)$ 3 ~ 4;</code></td></tr>
<tr><td>Magnus</td><td></td></tr>
<tr><td>Maxima</td><td><code>infix("~")$ "~"(x, y):= sqrt(x^2 + y^2)$ 3 ~ 4;</code></td></tr>
<tr><td>Maple</td><td><code>`&~`:= (x, y) -> sqrt(x^2 + y^2):   3 &~ 4;</code></td></tr>
<tr><td>Mathematica</td><td><code>x_ \[Tilde] y_:= Sqrt[x^2 + y^2];   3 \[Tilde] 4</code></td></tr>
<tr><td>MuPAD</td><td><code>tilde:= proc(x, y) begin sqrt(x^2 + y^2) end_proc:</code>
<code> 3 &tilde 4;</code></td></tr>
<tr><td>Octave</td><td></td></tr>
<tr><td>Pari</td><td></td></tr>
<tr><td>Reduce</td><td><code>infix |$ procedure |(x, y); sqrt(x^2 + y^2)$ 3 | 4;</code></td></tr>
<tr><td>Scilab</td><td></td></tr>
<tr><td>Sumit</td><td></td></tr>
<tr><td>Yacas</td><td></td></tr>
</table>

	Main expression operator	1st operand	List of expression operands
Axiom[7]		`kernels(e) . 1`	`kernels(e)`
Derive			*various*[8]
DoCon			
GAP	There are no formal unevaluated expressions		
Gmp			
Macsyma	`part(e, 0)`	`part(e, 1)`	`args(e)`
Magnus			
Maxima	`part(e, 0)`	`part(e, 1)`	`args(e)`
Maple	`op(0, e)`	`op(1, e)`	`[op(e)]`
Mathematica	`Head[e]`	`e@[[1]]`	`ReplacePart[e, List, 0]`
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)`
Scilab			
Sumit			
Yacas			

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

[6] `TERMS`, `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))]));`
Derive	`"sin(0)" = sin(0)`
DoCon	
GAP	`Print("There is no sin, but factors(10)= ",Factors(10), "\n")`
Gmp	
Macsyma	`print("sin(", 0, ") =", sin(0))$`
Magnus	
Maxima	`print("sin(", 0, ") =", sin(0))$`
Maple	`printf("sin(%a) = %a\n", 0, sin(0)):`
Mathematica	``Print[StringForm["sin(``) = ``", 0, Sin[0]]];``
MuPAD	`print(Unquoted, "sin(".0.")" = sin(0)):`
Octave	
Pari	
Reduce	`write("sin(", 0, ") = ", sin(0))$`
Scilab	
Sumit	
Yacas	

	Generate FORTRAN	Generate TEX/LATEX
Axiom	`outputAsFortran(e)`	`outputAsTex(e)`
Derive	`[Transfer Save Fortran]`	
DoCon		
GAP		`Print(LaTeX(e));`
Gmp		
Macsyma	`fortran(e)$` *or* `gentran(eval(e))$`	`tex(e);`
Magnus		
Maxima	`fortran(e)$` *or* `gentran(eval(e))$`	`tex(e);`
Maple	`fortran([e]);`	`latex(e);`
Mathematica	`FortranForm[e]`	`TexForm[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;`
Scilab		
Sumit		
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	
Derive	`[Transfer Load daTa]` (from `file.dat`)
DoCon	
GAP	
Gmp	
Macsyma	`xy: read_num_data_to_matrix("file", nrows, 2)$`
Magnus	
Maxima	`xy: read_num_data_to_matrix("file", nrows, 2)$`
Maple	`xy:= readdata("file", integer, 2):`
Mathematica	`xy = ReadList["file", Number, RecordLists -> True]`
MuPAD	
Octave	
Pari	
Reduce	
Scilab	
Sumit	
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`[7]

Axiom	`)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`

Derive	`xy [Transfer Print Expressions File]` (creates `file.prt`)
DoCon	
GAP	`PrintTo("file");for i in [1..n] do` `  AppendTo("file",xy[i][1]," ",xy[i][2],"\n");od;`
Gmp	
Macsyma	`writefile("file")$   for i:1 thru n do` `  print(xy[i, 1], xy[i, 2])$   closefile()$`
Magnus	
Maxima	`writefile("file")$   for i:1 thru n do` `  print(xy[i, 1], xy[i, 2])$   closefile()$`
Maple	`writedata("file", xy);`

Mathematica	`outfile = OpenWrite["file"];`
	`Do[WriteString[outfile,`
	`  xy@[[i, 1]], " ", xy@[[i, 2]], "\n"], {i, 1, n}]`
	`Close[outfile];`

MuPAD	`fprint(Unquoted, Text, "file",` `  ("\n", xy[i, 1], xy[i, 2]) $ i = 1..n):`
Octave	
Pari	
Reduce	`out "file";   for i:=1:n do` `  write(xy(i, 1), " ", xy(i, 2));   shut "file";`
Scilab	
Sumit	
Sumit	
Yacas	

4 Mathematics and Graphics

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

[7] Some 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)
Derive	#e	pi	#i	inf	SQRT(2)	2^(1/3)
DoCon						
GAP			E(4)	infinity	ER(2)[8]	
Gmp						
Macsyma	%e	%pi	%i	inf	sqrt(2)	2^(1/3)
Magnus						
Maxima	%e	%pi	%i	inf	sqrt(2)	2^(1/3)
Maple	exp(1)	Pi	I	infinity	sqrt(2)	2^(1/3)
Mathematica	E	Pi	I	Infinity	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)
Scilab						
Sumit						
Yacas						

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

[8] ER 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 1st kind
Axiom	legendreP(n, x)	chebyshevT(n, x)
Derive	LEGENDRE_P(n, x)	CHEBYCHEV_T(n, x)
DoCon		
GAP		
Gmp		
Macsyma	legendre_p(n, x)	chebyshev_t(n, x)
Magnus		
Maxima	legendre_p(n, x)	chebyshev_t(n, x)
Maple	orthopoly[P](n, x)	orthopoly[T](n, x)
Mathematica	LegendreP[n, x]	ChebyshevT[n, x]
MuPAD	orthpoly::legendre(n, x)	orthpoly::chebyshev1(n, x)
Octave		
Pari		
Reduce	LegendreP(n, x)	ChebyshevT(n, x)
Scilab		
Sumit		
Yacas		

	Fibonacci number	Elliptic integral of the 1st kind
Axiom	fibonacci(n)	
Derive	FIBONACCI(n)	ELLIPTIC_E(phi, k^2)
DoCon		
GAP	Fibonacci(n)	
Gmp		
Macsyma	fib(n)	elliptic_e(phi, k^2)
Magnus		
Maxima	fib(n)	elliptic_e(phi, k^2)
Maple	combinat[fibonacci](n)	EllipticE(sin(phi), k)
Mathematica	Fibonacci[n]	EllipticE[phi, k^2]
MuPAD	numlib::fibonacci(n)	
Octave		
Pari		
Reduce		EllipticE(phi, k^2)
Scilab		
Sumit		
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)`
Derive	`GAMMA(x)`	`PSI(x)`	`CI(x)`	`BESSEL_J(n, x)`
DoCon				
GAP				
Gmp				
Macsyma	`gamma(x)`	`psi[0](x)`	`cos_int(x)`	`bessel_j[n](x)`
Magnus				
Maxima	`gamma(x)`	`psi[0](x)`	`cos_int(x)`	`bessel_j[n](x)`
Maple	`GAMMA(x)`	`Psi(x)`	`Ci(x)`	`BesselJ(n, x)`
Mathematica	`Gamma[x]`	`PolyGamma[x]`	`CosIntegral[x]`	`BesselJ[n, x]`
MuPAD	`gamma(x)`	`psi(x)`		`besselJ(n, x)`
Octave				
Pari				
Reduce	`Gamma(x)`	`Psi(x)`	`Ci(x)`	`BesselJ(n, x)`
Scilab				
Sumit				
Yacas				

	Hypergeometric fun. ${}_2F_1(a, b; c; x)$	Dirac delta	Unit step fun.
Axiom			
Derive	`GAUSS(a, b, c, x)`		`STEP(x)`
DoCon			
GAP			
Gmp			
Macsyma	`hgfred([a, b], [c], x)`	`delta(x)`	`unit_step(x)`
Magnus			
Maxima	`hgfred([a, b], [c], x)`	`delta(x)`	`unit_step(x)`
Maple	`hypergeom([a, b], [c], x)`	`Dirac(x)`	`Heaviside(x)`
Mathematica	`HypergeometricPFQ[{a,b},{c},x]`	``@<< Calculus`DiracDelta` ``	
MuPAD		`dirac(x)`	`heaviside(x)`
Octave			
Pari			
Reduce	`hypergeometric({a, b}, {c}, x)`		
Scilab			
Sumit			
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	
Derive	`a(x):= -x*CHI(-inf, x, 0) + x*CHI(0, x, inf)`
DoCon	
GAP	
Gmp	
Macsyma	`a(x):= -x*unit_step(-x) + x*unit_step(x)$`
Magnus	
Maxima	`a(x):= -x*unit_step(-x) + x*unit_step(x)$`
Maple	`a:= x -> piecewise(x < 0, -x, x):`
Mathematica	``@<< Calculus`DiracDelta` `` `a[x_]:= -x*UnitStep[-x] + x*UnitStep[x]`
MuPAD	`a:= proc(x) begin -x*heaviside(-x) + x*heaviside(x)` `end_proc:`
Octave	
Pari	
Reduce	
Scilab	
Sumit	
Yacas	

	Assume x is real	Remove that assumption
Axiom		
Derive	`x :epsilon Real`	`x:=`
DoCon		
GAP		
Gmp		
Macsyma	`declare(x, real)$`	`remove(x, real)$`
Magnus		
Maxima	`declare(x, real)$`	`remove(x, real)$`
Maple	`assume(x, real);`	`x:= 'x':`
Mathematica	`x/: Im[x] = 0;`	`Clear[x]`
MuPAD	`assume(x, Type::RealNum):`	`unassume(x, Type::RealNum):`
Octave		
Pari		
Reduce		
Scilab		
Sumit		
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		
Derive	`x :epsilon (0, 1]`	`x:=`
DoCon		
GAP		
Gmp		
Macsyma	`assume(x > 0, x <= 1)$`	`forget(x > 0, x <= 1)$`
Magnus		
Maxima	`assume(x > 0, x <= 1)$`	`forget(x > 0, x <= 1)$`
Maple	`assume(x > 0);`	`x:= 'x':`
	`additionally(x <= 1);`	
Mathematica	`Assumptions -> 0 < x <= 1` [8]	
MuPAD	`assume(x > 0): assume(x <= 1):`	`unassume(x):`
Octave		
Pari		
Reduce		
Scilab		
Sumit		
Yacas		

	Basic simplification of an expression e
Axiom	`simplify(e)` *or* `normalize(e)` *or* `complexNormalize(e)`
Derive	`e`
DoCon	
GAP	`e`
Gmp	
Macsyma	`ratsimp(e)` *or* `radcan(e)`
Magnus	
Maxima	`ratsimp(e)` *or* `radcan(e)`
Maple	`simplify(e)`
Mathematica	`Simplify[e]` *or* `FullSimplify[e]`
MuPAD	`simplify(e)` *or* `normal(e)`
Octave	
Pari	
Reduce	`e`
Scilab	
Sumit	
Yacas	

[8] This is an option for `Integrate`.

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.

	Use an unknown function	Numerically evaluate an expr.
Axiom	`f:= operator('f);   f(x)`	`exp(1) :: Complex Float`

Derive	`f(x):=`
	`f(x)`
		`Precision:= Approximate`
		`APPROX(EXP(1))`
		`Precision:= Exact`

DoCon		
GAP		`EvalF(123/456)`
Gmp		
Macsyma	`f(x)`	`sfloat(exp(1));`
Magnus		
Maxima	`f(x)`	`sfloat(exp(1));`
Maple	`f(x)`	`evalf(exp(1));`
Mathematica	`f[x]`	`N[Exp[1]]`
MuPAD	`f(x)`	`float(exp(1));`
Octave		
Pari		
Reduce	`operator f;   f(x)`	`on rounded;   exp(1);` `off rounded;`
Scilab		
Sumit		
Yacas		

	n mod m	Solve $e \equiv 0$ mod m for x
Axiom	`rem(n, m)`	`solve(e = 0 :: PrimeField(m), x)`
Derive	`MOD(n, m)`	`SOLVE_MOD(e = 0, x, m)`
DoCon		
GAP	`n mod m`	solve using finite fields
Gmp		
Macsyma	`mod(n, m)`	`modulus: m$   solve(e = 0, x)`
Magnus		
Maxima	`mod(n, m)`	`modulus: m$   solve(e = 0, x)`
Maple	`n mod m`	`msolve(e = 0, m)`
Mathematica	`Mod[n, m]`	`Solve[{e == 0, Modulus == m}, x]`
MuPAD	`n mod m`	`solve(poly(e = 0, [x], IntMod(m)), x)`
Octave		
Pari		
Reduce	`on modular;` `setmod m$   n`	`load_package(modsr)$   on modular;` `setmod m$   m_solve(e = 0, x)`
Scilab		
Sumit		
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)`
Derive	`FACTOR(a/b + c/d, Trivial)`	`EXPAND((a*d + b*c)/(b*d))`
DoCon		
GAP	`a/b+c/d`	
Gmp		
Macsyma	`xthru(a/b + c/d)`	`expand((a*d + b*c)/(b*d))`
Magnus		
Maxima	`xthru(a/b + c/d)`	`expand((a*d + b*c)/(b*d))`
Maple	`normal(a/b + c/d)`	`expand((a*d + b*c)/(b*d))`
Mathematica	`Together[a/b + c/d]`	`Apart[(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)`
Scilab		
Sumit		
Yacas		

	Manipulate the root of a polynomial
Axiom	`a:= rootOf(x**2 - 2);   a**2`
Derive	
DoCon	
GAP	`x:=X(Rationals,"x"); a:=RootOfDefiningPolynomial(AlgebraicExtension(Rationals,x^2-2)); a^2`
Gmp	
Macsyma	`algebraic:true$   tellrat(a^2 - 2)$   rat(a^2);`
Magnus	
Maxima	`algebraic:true$   tellrat(a^2 - 2)$   rat(a^2);`
Maple	`a:= RootOf(x^2 - 2):   simplify(a^2);`
Mathematica	`a = Root[#^2 - 2 &, 2]   a^2`
MuPAD	
Octave	
Pari	
Reduce	`load_package(arnum)$   defpoly(a^2 - 2);   a^2;`
Scilab	
Sumit	
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		`solve([eqn1, eqn2], [x, y])`

Derive	`x :epsilon Nonscalar`
	`y :epsilon Nonscalar`
	`x . y`
		`SOLVE([eqn1, eqn2], [x, y])`

DoCon		
GAP	`*`	
Gmp		
Macsyma	`x . y`	`solve([eqn1, eqn2], [x, y])`
Magnus		
Maxima	`x . y`	`solve([eqn1, eqn2], [x, y])`
Maple	`x &* y`	`solve({eqn1, eqn2}, {x, y})`
Mathematica	`x ** y`	`Solve[{eqn1, eqn2}, {x, y}]`
MuPAD		`solve({eqn1, eqn2}, {x, y})`
Octave		
Pari		

Reduce	`operator x, y;`
	`noncom x, y;`
	`x() * y()`
		`solve({eqn1, eqn2}, {x, y})`

Scilab		
Sumit		
Yacas		

	Decrease/increase angles in trigonometric functions	
Axiom	`simplify(normalize(sin(2*x)))`	
Derive	`Trigonometry:= Expand` `sin(2*x)`	`Trigonometry:= Collect` `2*sin(x)*cos(x)`
DoCon		
GAP		
Gmp		
Macsyma	`trigexpand(sin(2*x))`	`trigreduce(2*sin(x)*cos(x))`
Magnus		
Maxima	`trigexpand(sin(2*x))`	`trigreduce(2*sin(x)*cos(x))`
Maple	`expand(sin(2*x))`	`combine(2*sin(x)*cos(x))`
Mathematica	`TrigExpand[Sin[2*x]]`	`TrigReduce[2*Sin[x]*Cos[x]]`
MuPAD	`expand(sin(2*x))`	`combine(2*sin(x)*cos(x), sincos)`
Octave		
Pari		
Reduce	`load_package(assist)$` `trigexpand(sin(2*x))`	 `trigreduce(2*sin(x)*cos(x))`
Scilab		
Sumit		
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, ...])`
Derive	
DoCon	
GAP	
Gmp	
Macsyma	`grobner([p1, p2, ...])`
Magnus	
Maxima	`grobner([p1, p2, ...])`
Maple	`Groebner[gbasis]([p1, p2, ...], plex(x1, x2, ...))`
Mathematica	`GroebnerBasis[{p1, p2, ...}, {x1, x2, ...}]`
MuPAD	`groebner::gbasis([p1, p2, ...])`
Octave	
Pari	
Reduce	`load_package(groebner)$   groebner({p1, p2, ...})`
Scilab	
Sumit	
Yacas	

	Factorization of e over $i = \sqrt{-1}$
Axiom	`factor(e, [rootOf(i**2 + 1)])`
Derive	`FACTOR(e, Complex)`
DoCon	
GAP	`Factors(GaussianIntegers,e)`
Gmp	
Macsyma	`gfactor(e);` *or* `factor(e, i^2 + 1);`
Magnus	
Maxima	`gfactor(e);` *or* `factor(e, i^2 + 1);`
Maple	`factor(e, I);`
Mathematica	`Factor[e, Extension -> I]`
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;`
Scilab	
Sumit	
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	`real(f(z))`	`complexForm(f(z))`
Derive	`RE(f(z))`	`f(z)`
DoCon		
GAP	`(f(z)+GaloisCyc(f(z),-1))/2`	
Gmp		
Macsyma	`realpart(f(z))`	`rectform(f(z))`
Magnus		
Maxima	`realpart(f(z))`	`rectform(f(z))`
Maple	`Re(f(z))`	`evalc(f(z))`
Mathematica	`Re[f[z]]`	`ComplexExpand[f[z]]`
MuPAD	`Re(f(z))`	`rectform(f(z))`
Octave		
Pari		
Reduce	`repart(f(z))`	`repart(f(z)) + i*impart(f(z))`
Scilab		
Sumit		
Yacas		

	Matrix addition	Matrix multiplication	Matrix transpose
Axiom	`A + B`	`A * B`	`transpose(A)`
Derive	`A + B`	`A . B`	``A` ``
DoCon			
GAP	`A + B`	`A * B`	`TransposedMat(A)`
Gmp			
Macsyma	`A + B`	`A . B`	`transpose(A)`
Magnus			
Maxima	`A + B`	`A . B`	`transpose(A)`
Maple	`evalm(A + B)`	`evalm(A &* B)`	`linalg[transpose](A)`
Mathematica	`A + B`	`A . B`	`Transpose[A]`
MuPAD	`A + B`	`A * B`	`transpose(A)`
Octave			
Pari			
Reduce	`A + B`	`A * B`	`tp(A)`
Scilab			
Sumit			
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 ___`
Derive	
DoCon	
GAP	`SolutionMat(TransposedMat(A),b)`
Gmp	
Macsyma	`xx: genvector('x, mat_nrows(b))$` `x: part(matlinsolve(A . xx = b, xx), 1, 2)`
Magnus	
Maxima	`xx: genvector('x, mat_nrows(b))$` `x: part(matlinsolve(A . xx = b, xx), 1, 2)`
Maple	`x:= linalg[linsolve](A, b)`
Mathematica	`x = LinearSolve[A, b]`
MuPAD	
Octave	
Pari	
Reduce	
Scilab	
Sumit	
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)`
Derive	`SUM(f(i), i, 1, n)`	`PRODUCT(f(i), i, 1, n)`
DoCon		
GAP	`Sum([1..n],f)`	`Product([1..n],f)`
Gmp		
Macsyma	`closedform(` `  sum(f(i), i, 1, n))`	`closedform(` `  product(f(i), i, 1, n))`
Magnus		
Maxima	`closedform(` `  sum(f(i), i, 1, n))`	`closedform(` `  product(f(i), i, 1, n))`
Maple	`sum(f(i), i = 1..n)`	`product(f(i), i = 1..n)`
Mathematica	`Sum[f[i], {i, 1, n}]`	`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)`
Scilab		
Sumit		
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: $\lim_{x\to 0-} f(x)$	Taylor/Laurent/etc. series
Axiom	`limit(f(x), x = 0, "left")`	`series(f(x), x = 0, 3)`
Derive	`LIM(f(x), x, 0, -1)`	`TAYLOR(f(x), x, 0, 3)`
DoCon		
GAP		
Gmp		
Macsyma	`limit(f(x), x, 0, minus)`	`taylor(f(x), x, 0, 3)`
Magnus		
Maxima	`limit(f(x), x, 0, minus)`	`taylor(f(x), x, 0, 3)`
Maple	`limit(f(x), x = 0, left)`	`series(f(x), x = 0, 4)`
Mathematica	`Limit[f[x], x->0, Direction->1]`	`Series[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)`
Scilab		
Sumit		
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)`
Derive	`DIF(DIF(f(x, y), x), y, 2)`	`INT(f(x), x, 0, 1)`
DoCon		
GAP		
Gmp		
Macsyma	`diff(f(x, y), x, 1, y, 2)`	`integrate(f(x), x, 0, 1)`
Magnus		
Maxima	`diff(f(x, y), x, 1, y, 2)`	`integrate(f(x), x, 0, 1)`
Maple	`diff(f(x, y), x, y$2)`	`int(f(x), x = 0..1)`
Mathematica	`D[f[x, y], x, {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)`
Scilab		
Sumit		
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)`
Derive	`LAPLACE(e, t, s)`	
DoCon		
GAP		
Gmp		
Macsyma	`laplace(e, t, s)`	`ilt(e, s, t)`
Magnus		
Maxima	`laplace(e, t, s)`	`ilt(e, s, t)`
Maple	`inttrans[laplace](e,t,s)`	`inttrans[invlaplace](e,s,t)`
Mathematica	`@<< Calculus`LaplaceTransform``	
	`LaplaceTransform[e, t, s]`	`InverseLaplaceTransform[e,s,t]`
MuPAD	`transform::laplace(e,t,s)`	`transform::ilaplace(e, s, t)`
Octave		
Pari		
Reduce	`load_package(laplace)$`	`load_package(defint)$`
	`laplace(e, t, s)`	`invlap(e, t, s)`
Scilab		
Sumit		
Yacas		

	Solve an ODE (with the initial condition $y'(0) = 1$)
Axiom	`solve(eqn, y, x)`
Derive	`APPLY_IC(RHS(ODE(eqn, x, y, y_)), [x, 0], [y, 1])`
DoCon	
GAP	
Gmp	
Macsyma	`ode_ibc(ode(eqn, y(x), x), x = 0, diff(y(x), x) = 1)`
Magnus	
Maxima	`ode_ibc(ode(eqn, y(x), x), x = 0, diff(y(x), x) = 1)`
Maple	`dsolve({eqn, D(y)(0) = 1}, y(x))`
Mathematica	`DSolve[{eqn, y'[0] == 1}, y[x], x]`
MuPAD	`solve(ode({eqn, D(y)(0) = 1}, y(x)))`
Octave	
Pari	
Reduce	`odesolve(eqn, y(x), x)`
Scilab	
Sumit	
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	`DD : LODO(Expression Integer, e +-> D(e, x)) := D();` `L:= DD + 1;   L(sin(x))`
Derive	
DoCon	
GAP	
Gmp	
Macsyma	`load(opalg)$   L: (diffop(x) - 1)$   L(sin(x));`
Magnus	
Maxima	`load(opalg)$   L: (diffop(x) - 1)$   L(sin(x));`
Maple	`id:= x -> x:   L:= (D + id):   L(sin)(x);`
Mathematica	`L = D[#, x]& + Identity;   Through[L[Sin[x]]]`
MuPAD	`L:= (D + id):   L(sin)(x);`
Octave	
Pari	
Reduce	
Scilab	
Sumit	
Yacas	

	2D plot of two separate curves overlayed
Axiom	`draw(x, x = 0..1);   draw(acsch(x), x = 0..1);`
Derive	`[Plot Overlay]`
DoCon	
GAP	
Gmp	
Macsyma	`plot(x, x, 0, 1)$   plot(acsch(x), x, 0, 1)$`
Magnus	
Maxima	`plot(x, x, 0, 1)$   plot(acsch(x), x, 0, 1)$`
Maple	`plot({x, arccsch(x)}, x = 0..1):`
Mathematica	`Plot[{x, ArcCsch[x]}, {x, 0, 1}];`
MuPAD	`plotfunc(x, acsch(x), x = 0..1):`
Octave	
Pari	
Reduce	`load_package(gnuplot)$   plot(y = x, x = (0 .. 1))$` `plot(y = acsch(x), x = (0 .. 1))$`
Scilab	
Sumit	
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.

	Simple 3D plotting
Axiom	`draw(abs(x*y), x = 0..1, y = 0..1);`
Derive	`[Plot Overlay]`
DoCon	
GAP	
Gmp	
Macsyma	`plot3d(abs(x*y), x, 0, 1, y, 0, 1)$`
Magnus	
Maxima	`plot3d(abs(x*y), x, 0, 1, y, 0, 1)$`
Maple	`plot3d(abs(x*y), x = 0..1, y = 0..1):`
Mathematica	`Plot3D[Abs[x*y], {x, 0, 1}, {y, 0, 1}];`
MuPAD	`plotfunc(abs(x*y), x = 0..1, y = 0..1):`
Octave	
Pari	
Reduce	`load_package(gnuplot)$`
	`plot(z = abs(x*y), x = (0 .. 1), y = (0 .. 1))$`
Scilab	
Sumit	
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.