Glossary

The Maple procedures and operators used in this manual are listed, in alphabetical order. If appropriate, a brief description of syntax and output is also included. In all cases see the corresponding Maple Help page for more complete explanations.

@ Composition operator for functions. Syntax: f@g is the composition of functions f and g.
(f@@5) is f composed with itself 5 times.
(D@@5)(f) is the fifth derivative of f.

a..b The range of values from a to b.
`> int( sin(x), x=1..2);` output: cos(1) − cos(2)

Add A procedure in the **LinearAlgebra** package for adding vectors or matrices. Syntax: Add(A,B)

add The procedure to use when adding a few numbers, vectors, etc. Syntax: add(f(k), k=1..n).
`> add( k*(k+1), k=-1..3);` output: 20

alias Used to create an alias of a long Maple name.
Syntax: alias(H=Heaviside); Output: H

animate A plot procedure that animates plots. In the plot package.
Syntax: animate(plot command, [Input for plot command], param=a..b, frames=n)

animatecurve A procedure that creates an animation of the plotting of a 2-d graph. In the plots package.
Syntax: animatecurve(f(x), x=a..b, frames=n)
Also applies to parametrized curves and works in polar coordinates (funky).

anonymous function
A function with no name.
`> (x->x^2)(3);` Output: 9

args The name of the sequence of inputs in a procedure.
`> f:=proc() "Hello",args[1],"how are you." end proc: f(world,x,z);`
Output: "Hello",*world*,"how are you."

arguments The inputs in a procedure are called its arguments. See **args** and **nargs**.

array A Maple data structure like a list but more versatile because the index set can include negative integers and zero. Multiple indices can also be used. Syntax: array(a..b,c..d, ..., list).
`> A := array(-3..1,[1,2,3,4,5]): A[-1],A[0];` output: 3, 4

arrow Creates and plots arrows. In the plots package.
Syntax: arrow([a,b]) plots an arrow from [0,0] to [a,b]
arrow([a,b], [c,d]) plots an arrow at [a,b] with direction [c,d]
arrow([a,b], [c,d], difference=true) plots an arrow from [a,b] to [c,d]
Optional equations control the shape and size of the arrow.

arrow definition of a function

A function f can be created using the syntax f := x -> expression in x. This is referred to as an arrow definition.

```
> f := x -> sin(x)/x; f(0) := 1:
```

arrows The optional equation **arrows=none** in the **DEplot** procedure will suppress the arrows. Use **arrows=line** to draw line segments instead of the default harpoon arrows.

assign A procedure that converts an equation, equations, or a set or a list of equations into assignments. Syntax: assign(x=expression)

```
> assign(x=5);
```
output: none, but henceforth x will be a name for 5.
To free the x variable use either **unassign('x')** or **x := 'x'**

:= The assignment operator. Assigns a name to an expression. Syntax: name := expression

```
> y := sin(t): t = z: y + x;
```
output: $\sin(t) + x$
Note that the equation t = z does not assign the name t to the expression z.

assume The optional equation **assume=property** in the **simplify** procedure can be used to tell Maple to make the simplification under the stated assumption.

```
> simplify( sqrt(a^2), assume=positive);
```
Output: *a*
The entry

```
> assume(x::positive);
```
tells Maple to assume x is positive in all calculations.

assuming Place an assumption on x in a particular calculation by adding **assuming x::property**.

```
> simplify( sqrt(a^2) ) assuming a::negative;
```
Output: - *a*

backward quotes Use backward quotes ` ` to make any list of symbols into a name.

```
> `A big number`:=10: `A big number`, `A big number`^2;
```
output: 10, 100

base Keyword for **convert**. Converts an integer to a different base.

```
> convert(114,base,3);
```
Output: [0, 2, 0, 1, 1]

BesselJ Maple's symbol for the Bessel functions of the first kind. BesselY denotes the Bessel functions of the second kind.

binary function A function of two variables, **(x,y) -> expression in x and y**.

binomial The binomial coefficient function: binomial(m,n) = m!/n!/(m-n)!

boolean connectives

Maple supports the boolean connectives **implies**, **and**, **or**, **xor** (exclusive or) as well as the unary **not**.

```
> is( 2=5 or 2=2);
```
output: *true*

```
> is( 2=5 and 2=2);
```
output: *false*

boolean function A function having the values true, false.

```
> f := t-> type(t,numeric): f(sin(2));
```
Output: *false* (sin(2) is of type *constant*.)

break An execution control command. The entry **break** tells Maple to immediately leave the current repetition statement (for..do..end do) and continue execution with the next input. See **next** and **return**.
> for k from 1 to 10 do if isprime(k) then break end if; end do; k^2;
Output: 4

by Used to make a repetition statement.
> for k from 6 by -2 to 0 while not isprime(k) do k end do;
Output: The numbers 6, 4 on separate output lines.

cat Concatenation function: cat(a,b) concatenates symbols a and b. If a and b are strings the output is a string. The symbol || serves as a concatenation operator. a||b and cat(a,b) have same output when a and b are unassigned symbols.
> cat(a,b); a||b; Output: Line1, ab Line2, ab
> cat("Hello","World"); Output: "HelloWorld"
> cat(x,1..5); Output: $x1, x2, x3, x4, x5$
> cat(x,1..5,y); Output: $x1y, x2y, x3y, x4y, x5y$

collect A procedure that collects like terms in an expression. Use it to make polynomials.
> collect(x*(y+x*(w+x*t)), x); output: $tx^3 + wx^2 + xy$

:: (double colon) The double colon is used by Maple to assign a property to a name.
> simplify(abs(x)) assuming x::negative; Output: $-x$

coeff Calculates the coefficients in a polynomial. Syntax: coeff(polynom, x, 4) or coeff(polynom, x^4)
> coeff(x*y^6 + x^3 + 3*x,x,1); Output: $y^6 + 1$

Column A procedure in the **LinearAlgebra** package. Returns a column of a matrix. Syntax: Column(A,k)
> Column(<<1,2,3>|<4,5,6>|<7,8,9>>,2); output: The second column of the matrix.

ColumnSpace A procedure in the **LinearAlgebra** package. Returns a list of basis vectors for the column space of a matrix. Syntax: ColumnSpace(A)
> ColumnSpace(<<1,2,3>|<4,5,6>|<7,8,9>>); output: A list containing <1,0,1> and <0,1,2>

combine The opposite of expand. Use keywords to combine subexpressions.
> combine(exp(a)*exp(b)+sin(x)*sin(2*x)); Output: exp(a+b) + cos(x)/2 - cos(3x)/2

concatenation The symbol **||** (two vertical bars) is called the *concatenation operator*. Use it to form names by combining two sub names. The use of this operator is discouraged. Use **cat** instead.
> x||(1..4); output: x1, x2, x3, x4
> cat(x,1..4); output: x1, x2, x3, x4

confrac Keyword for **convert**, converts real numbers and rational polynomial expressions to continued fraction form.
> convert((t^2+1)/(t^3-t), confrac, t);

contourplot Plots 2-d contours of the form F(x,y) = c. In the plots package.
Syntax? contourplot(F(x,y), x=a..b, y=c..d, contours=n)
Also accepts contours=[list of values]

convert A procedure that converts expressions of one type into those of a similar type. Does partial fractions via **parfrac**. Syntax: convert(expression, type)

```
> convert( [[1,2],[3,4]], Matrix);
> convert( 1/t^2/(t-1)^2, parfrac, t);
```

coordplot Plots 2 dimensional coordinate lines. In the plots package.
Syntax: coordplot(polar)
coordplot3d plots 3 dimensional coordinate surfaces.

cylinderplot Plots in cylindrical coordinates. In the plots package. Plots r as a function of theta and z.
Syntax: cylinderplot(r(theta,z), theta=a..b, z=c..d)

D The derivative operator. Used to differentiate functions. Syntax: D(f)
> f := x->x*sin(x): D(f); output: x -> sin(x) + x cos(x)

DEtools A package of procedures used to analyze differential equations or a system of differential equations. Load it using **with(DEtools)**.

DEplot A procedure in the **DEtools** package. Plots numerical solutions to differential equations and systems. Can also generate direction fields and vector fields.
Syntax: DEplot(DE, y(x), x=-2..2, y=-2..2) will plot the direction field for a first order ode named DE if its independent variable is x and its dependent variable is y(x).
Add a set of initial conditions of the form { [a,b], [c,d] , etc. } to generate solution curves.

degree Calculates the degree of a polynomial. Include the principal variable.
> degree(x*y^6 + x^3 + 3*x,x); Output: 3

Determinant Procedure in LinearAlgebra to compute the detenminant of a matrix: Determinant(A).

diff The differentiation procedure used for the differentiation of expressions.
Syntax: diff(expression, variable). Also computes partial derivatives. Inert form: Diff.
> diff(sin(x), x); output: cos(x)
For the second derivative put in the independent variable twice.
> diff(sin(x), x, x); output: - sin(x)

Dirac The Dirac delta function. Dirac is the derivative of Heaviside. Dirac's derivative is denoted Dirac(1,x), etc.
> diff(Dirac(x),x); output: Dirac(1,x)

dirgrid The optional equation **dirgrid=[m,n]** can be used to control the number of arrows drawn by the **DEplot** procedure.

discont An optional equation for the plot procedure: **discont=true** tells Maple *not* to connect the end points at a jump discontinuity. The default is **discont=false**.

display A procedure that displays several plots at one time. In the **plots** package. Call it using **plots[display]**.
> P1 := plot(sin(x),x=0..Pi): P2 := plot(cos(t),t=Pi..2*Pi):
plots[display](P1,P2); output: The two plots on the same set of axes.

divide Boolean to determine if polynomial g divides polynomial f: divide(f,g); The entry divide(f,g,'q') tells Maple to store the quotient with the name q.
> divide(x^3+x+2,x+1); Output: *true*
> divide(x^3+x+2,x+1,'q'); q; Output: Line1, *true*, Line2, x^2 - x + 2

do..end do A do statement. Used in Maple programming. Typically appears as part of a **for** statement.
> S := {}: for k from 1 to 10 do S := S union {k}: end do: S;
output: {1, 2, ... , 10}

dsolve A procedure for solving differential equations, both symbolically and numerically. Syntax: dsolve(DE), dsolve({DE, inits}), dsolve({DE, inits}, y(x), type=numeric) where DE is a differential equation and inits is a sequence of initial values.
> dsolve(diff(y(t),t) = y(t)); output: y(t) = _C1 e^t

Eigenvalues Procedure in LinearAlgebra packate. Eigenvalues(A) returns A's eigenvalues in a column vector.

Eigenvectors Procedure in LinearAlgebra packate. Eigenvectors(A) returns A's eigenvalues in a column vector and its eigenvectors as columns in a matrix.

elif Abbreviation for "or else if". Used in if..then statements.
> a:=1: b:=2:
if a>b then OK elif a=b then NOT else BIG end if; Output: BIG

else Used in if..then statements. See the example above.

eval A procedure used to evaluate an expression or the expression represented by a symbol.
Syntax: eval(expression) or eval(expression, variable=value)
> eval(sin(2*x), x=Pi); output: 0
The entry eval(symbol,n) evaluates symbol to level n.
> x:=y: y:=z: z:=3: eval(x,2),eval(x,1),eval(x); Output: *z*, *y*, 3

evalf A procedure used to evaluate a symbolic constant to floating point form.
> evalf(sin(Pi/4), 4); output: 0.7070

evaluation rules Maple applies the rule of *full evaluation* to most names that are defined in a worksheet. *First level evaluation* applies to all local variables in a procedure. *Last name evaluation* applies to symbols representing some objects (tables and arrays). Use **eval** or **print** to force full evaluation.

expand Applied to an expression, **expand** multiplies out, applies trig identities, expands exponential and log expressions, etc. to break the input into smaller pieces (not necessarily simpler).
> expand(sin(x+y); Output: sin(x) cos(y) + cos(x) sin(y)

expression Any group of symbols that makes sense to Maple. That is, Maple can process it and produce an output

(which might be NULL).

expression tree Expression tree refers to a graph constructed from the data structure of a Maple expression. The procedures and the symbols in the expression appear as nodes. An expression sequence does not have an expression tree.

factor Used to factor polynomials, not integers. Use **ifactor** to factor integers.
> factor(x^4-x); Output: x(x-1)(x^2+x+1)

factorset A procedure in the numtheory package. Finds the prime factors of an integer.
> factorset(1234567890); output: {2, 3, 5, 3607, 3803}

fieldplot Plots vectors in a 2-d vector field. In the plots package.
Syntax: fieldplot([f(x,y),g(x,y)], x=a..b, y=c..d, arrows=slim, dirgrid=[12,12])
Other arrow styles are available. Default, harpoon arrows.

first level evaluation
An evaluation rule. First level evaluation applies to local variables in a Maple procedure.
> f:=proc() local x,y,z; x:=y; y:=z; z:=3; x; end proc: f(); Output: *y*

floor A function that returns the largest integer less than or equal to the input number.
> floor(Pi), floor(-Pi); output: 3, -4

for Starts a *for statement*, useful in Maple programming. Often used in conjunction with a *do statement*.
Syntax: for k from a to b by c (or, for k from a by c to b)
> S:=0: for k from -1 to 3 by 1/2 do S:=S+k: end do: S,k; output: 7/2, 9

full evaluation An evaluation rule. A name or symbol used in a Maple worksheet is generally given *full evaluation* when called in subsequent entries.
> x:=y: y:=z: z:=3: x; Output: 3

fsolve A procedure for obtaining floating point (approximate) solutions to an equation in one variable. Specify a search interval by adding a range a..b.
> fsolve(sin(x) = sqrt(x)*cos(x), x, 0..1); output: .6948992874

function Can refer to the classic function concept. To create such a function use the arrow operator.
> f := x -> x*sin(x); output: f:= x -> x sin(x)

GAMMA The Gamma function: GAMMA(x) = (x-1)! when *x* is a positive integer.
> GAMMA(3), GAMMA(4), GAMMA(1/2); output: 2, 6, sqrt(Pi)

gamma Euler's constant.

gcd Polynomial greatest common divisor function.
> gcd(x^2+4*x+4,x^2+x-2); Output: x + 2

global A variable used in a Maple procedure can be declared global. Such variables, which are fully evaluated,

will interact with variables used outside of the procedure as if they were defined outside the procedure. See *full evaluation*. Maple discourages the use of global variables in a procedure. Use a name declaration in the input instead. See *name declaration*.

```
> f:=proc(y,x,der::name) der:=diff(y,x); end proc:
y:=sin(t^2): f(y,t): yprime;
```

Output: 2 cos(t^2) t

has Binary boolean to determine if an expression has a certain sub-expression.

```
> has( x^2 + sin(x) - 2, sin);
```

Output: *true*

Heaviside The unit step function. Heaviside(0) is undefined.

identify Attempts to identify a floating point approximation as an exact number

```
> identify(3.14159);
```

Output: Pi

Do not use it for "easy" numbers like 5.2.

```
> identify(5.2);
```

Output: 5.

if..then if..then statements are useful programming tools in Maple. Finish with **end if**.

```
> a:=1: b:=2: if (a+1<b-1) then a+b else a-b end if;
```

output: -1

ifactor Used to factor integers. Use **factor** to factor polynomials.

```
> ifactor(123456);
```

Output: (2^6)(3)(643)

igcd Integer greatest common divisor.

```
> igcd(123,456,789);
```

Output: 3

ilcm Integer least common multiple.

```
> ilcm(123,456,789);
```

Output: 4917048

implicitplot Plots equations in two variables (implicitly). In the plots package.
Syntax: implicitplot(x^2 + y^2 = 1, x=-2..2, y=-2..2)
Use grid=[m,n] to get smoother curves. Use **implicitplot3d** for 3-d plots.

intersect Set intersection. Syntax: A intersect B

```
> {1,2,3} intersect {2,3,4};
```

output: {2,3}.

inequal Plots inequalities. In the plots package. Put the inequalities in a list.
Syntax: inequal([Inequalities], x=a..b, y=c..d)
Use optional equations: optionsfeasible=(color=red), optionsexcluded=(color=yellow) to color the regions.

infinity Used to designate “larger than any positive number”. Prints as the infinity symbol (sideways 8)

```
> limit( sin(x)/x, x=infinity);
```

output: 0

infolevel A Maple procedure that can be used to request more information about a particular Maple procedure. For example, the entry **infolevel[dsolve] := 2** causes Maple to list the methods it applies when **dsolve** is used to solve a differential equation.

int The integration procedure, both indefinite and definite. Syntax: int(expression,variable). Inert form: Int.
> int(sin(x), x); output: - cos(x)
> int(sin(x), x=0..Pi); output: 2

interface Maple procedure used to set interface variables (those Maple variables that effect the interaction between Maple and the user).
> interface(verboseproc=2); Output: Current value for **verboseproc** (default = 1). Henceforth Maple will print more information about procedures.

iquo The integer quotient function: iquo(a,b) is the greatest integer less than or equal to a/b, (a, b integers). The entry iquo(a,b,'r') stores the remainder with the name r.
> iquo(10,3); Output: 3
> iquo(10,3,'r'): r; Output: 1

irem The integer remainder function: irem(a,b) is the remainder when a is divided by b (a, b integers). The entry irem(a,b,'q') stores the quotient with the name q.
> irem(10,3); Output: 1
> irem(10,3,'q'): q; Output: 3

is A procedure used to ask a question having an answer that is either true or false.
Syntax: is(expression, property or type)
> n:=1.5: m:=3/2: is(10*n,integer), is(10*m,integer); output: *false, true*
Question: What happened here?
Answer: 10*n evaluates as the floating point number 15.0 which Maple does not consider to be an integer. 10*m evaluates as the integer 15.

isprime Boolean function that determines if an integer is prime.
> isprime(1234567); Output: *false*

last name evaluation
An evaluation rule. Maple applies *last name evaluation* to symbols representing some types of data (including tables and arrays). Use **eval** or **print** to force full evaluation.
> A := array(0..2,[1,2,3]): B := A: B; Output: *A*
> eval(B); Output: *ARRAY*([0..2], [(0)=1, (1)=2, (2)=3])

lcm Polynomial least common multiple function.
> lcm(x^2+4*x+4,x^2+x-2); Output: (x+2)(x^2 + x - 2)

ldegree Calculates the least degree of a polynomial. Include the principal variable.
> ldegree(x*y^6 + x^3 + 3*x,x); Output: 1

left Used to make a left hand limit.
> limit(sin(x)/x^2, x=0, left); output: - infinity

lhs Left hand side procedure. Outputs the left hand side of an equation.
> lhs(E = m*c^2); output: *E*

length The length function returns the length of an integer.
> length(123^321); Output: 671

limit The limit procedure. Syntax: limit(expression,variable=value). Inert form: Limit.
> limit(sin(x)/x, x=0); output: 1

LinearAlgebra
A package of procedures to handle vectors and matrices. Load it using **with(LinearAlgebra)**.

LinearSolve A procedure in the **LinearAlgebra** package for solving a matrix equation AX = B. Syntax: LinearSolve(A,B)
> LinearSolve(<<1,2>|<3,4>>,<5,6>); output: The column vector **<-1,2>**.

linestyle The optional equation **linestyle=2** in a plot procedure draws curves using dotted lines. There are four linestyles. The default is **linestyle=1**. Use the following input to see all four styles.
> plot([1,2,3,4],0..5,0..5,linestyle=[1,2,3,4],color=black);

list A sequence enclosed in square brackets. Syntax: [a, b, c, ... , d]

local A variable used in a Maple procedure can be declared local. Such variables, which are only evaluated to the first level, do not interact with variables used outside of the procedure. See *first level evaluation*.

map Used to map functions into expressions.
> map(f,[x,y,z]); Output: [f(x), f(y), f(z)]

map2 Used to map functions f(x,y) into expressions.
> map2(f,a,[x,y,z]); Output: [f(a,x), f(a,y), f(a,z)]

Matrix A procedure used to create matrices. Use it whenever the matrix is to be manipulated by procedures in the **LinearAlgebra** package.
Syntax: Matrix(m,n,[[a,b,..,c],...,[d,e,..f]]) creates an m x n matrix with rows [a,b,..,c], ... , [d,e,...,f].
The same matrix can be made by entering **<<a,b,..,c>|...|<d,e,...,f>>**.
An entry of the form Matrix(m,n,(i,j)->f(i,j)) can also be used.

maximize A procedure that returns the maximum value of an expression of one variable.
Syntax: maximize(expression, variable=a..b).
> maximize(sin(x), x=0..Pi); output: 1
minimize works in the same way.

Maximize A numerical procedure in the **Optimization** package (Maple 9.5).
Syntax: Maximize(expression, set of constraints).
> Maximize(sin(x),{x >= 0, x<=Pi}); output: [1.,x = 1.57079632679504266].
Minimize works the same way.

member Binary boolean function to determine membership (set, list).
> **member(3, {1,2,3});** Output: *true*

minus Set subtraction. Syntax: A minus B
> {1,2,3} minus {2,3,4}; output: {1}.

Multiply A procedure in the **LinearAlgebra** package for multiplying matrices.

Syntax: Multiply(A,B). The syntax **A.B** can also be used for matrix multiplication.

name declaration Maple discourages the use of global variables in a procedure. Use a name declaration in the input instead.
```
> f:=proc(y,x,der::name) der:=diff(y,x); end proc:
y:=sin(t^2): f(y,t): yprime;
```
Output: 2 cos(t^2) t

nargs The number of inputs in a procedure. The inputs are called the *arguments* of the procedure.
```
> f:=proc() "Hello you",nargs,"how are you." end proc: f(world,x,z);
```
Output: "Hello you",3,"how are you."

next An execution control command. The entry **next** tells Maple to immediately increment the counter in a for..while do loop. See **break** and **return**.
```
> for k from 1 to 5 do if isprime(k) then next end if; k; end do;
```
Output: 1 and 4 on separate output lines.

nextprime A Maple procedure, nextprime(n) returns the next prime after the integer n.
```
> nextprime(11111111111111);
```
Output: 11111111111123

numpoints An option in plot procedures. For example, the equation **numpoints=200** tells Maple to plot at least 200 points when making a line graph.

numtheory The name of a package of special procedures to deal with questions in number theory. See **factorset**.

op, nops The procedure op extracts the operands from an expression, nops counts them.
Syntax: op(expression) returns a sequence of all the operands in an expression,
op(k,expression) returns the kth operand,
op(0,expression) returns the top level type of the expression
nops(expression) returns the number of operands in an expression.
```
> op(a+b), nops(a+b), op(2,a+b), op(0,a+b);
```
output: a, b, 2, b, +

Optimization A package of numerical optimization procedures. New to Maple 9.5.

output=array Use the optional equation **output=array(list)** in the numeric **dsolve** procedure **dsolve(..., type=numeric)** to get an output consisting of an array of approximate solution values.

parfrac Keyword for **convert**, converts rational polynomial expressions to partial fraction form.
```
> convert((t^2+1)/(t^3-t), parfrac, t);
```
Output: 1/(t+1) - 1/t + 1/(t-1)

patchcontour One of several optional plot styles for 3d graphs. This one draws contour lines on the surface.
Syntax: style=patchcontour. Other styles include patch, patchnogrid, contour, wireframe.

% Percent sign, also called the *ditto operator*. Refers to the last Maple output, in time.
```
> sin(Pi/4): evalf(%);
```
output: .7071067810

Pi Maple's symbol for the famous mathematical constant.

piecewise A procedure used to make piecewise defined expressions.
Syntax: piecewise(x<0, x, x<=2, x^2, sin(x)) makes the expression that evaluates to x if x < 0, x^2 if x is not less than 0 but is less than or equal to 2, and to sin(x) otherwise.

plot The Maple procedure for making 2 dimensional plots. Use plot3d for 3 dimensional plots.
> plot(sin(x), x=0..Pi); output: The graph of sin(x) from x = 0 to x = Pi.

plots A Maple package of specialized plot procedures. Load it using **with(plots)**. To call a particular procedure use **plots[procedure]**.
> plots[implicitplot](x^2 = y^2 -1, x=-3..3, y=-3..3);
output: a 2d plot of the equation $x^2 = y^2 - 1$ in the window $-3 \le x \le 3$, $-3 \le y \le 3$.

pointplot Plots points in 2-dimensions. In the plots package. Different symbols can be used (circle, box, cross, diamond).
Syntax: pointplot([[a,b], [c,d], ...], symbol=cross, symbolsize=18)

polygonplot Plots 2-d convex polygons. In the plots package. Put the vertices in a list.
Syntax: polygonplot([Vertices])
Use polygonplot3d for polygons in 3 dimensions.

polyhedraplot Plots 3 dimensional polyhedra. In the plots package.
Syntax: polyhedraplot([a,b,c], polytype=dodecahedron, polyscale=1)
Enter polyhedra_supported() to see a set of polyhedra that Maple can plot.

polynom Abbreviation for polynomial. Used, for example, in the **convert** procedure.
> convert(taylor(sin(x),x=0), polynom); output: x - x^3/6 + x^5/120

prevprime A Maple procedure, prevprime(n) returns the prime preceding the integer n.
> prevprime(11111111111111); Output: 11111111111053

print A procedure used to make Maple print something. Text can be printed if enclosed in backwards quotes as illustrated below.
> print(`My name is Joe.`,5+4,int(sin(x),x));
output: My nam is Joe., 9, - cos(x)

printlevel A Maple variable whose value determines how much information is included in a Maple output. Default value is **printlevel := 1**, **printlevel := -1** tells Maple to suppress all output. See **infolevel**.

proc Makes user-defined procedures.
> f := proc(x) x^2-x+sin(x) end proc: f(car); output: car^2 - car + sin(car)
The same effect can be accomplished with the arrow notation.
> f := x -> x^2-x+sin(x); f(car);
output: line1:f:= x -> x^2 - x + sin(x), line 2: car^2 - car + sin(car)

quo Polynomial quotient function: quo(f,g,x) is the quotient when polynomial f is divided by polynomial g

(rational coefficients). The entry quo(f,g,x,'r') computes the quotient and stores the remainder with the name r. Inert form Quo, used in conjunction with mod p (p a prime), tells Maple to calculate over the ring of integers mod p.

```
> quo(x^4+x^3+x+1,x^2-1,x);   Output: 1 + x + x^2
> quo(x^4+x^3+x+1,x^2-1,x,'r'): r;   Output: 2 + 2x
> Quo(x^4+3*x^3+2*x^2+4,x^2-1,x) mod 3   Output: x
```

rationalize Rationalizes expressions with fractional exponents.

```
> rationalize((x+x^(1/3))/(x^(3/2)+x^3));
```

rand A Maple procedure with no arguments. The entry **rand()** outputs a (pseudo) random 12 digit positive integer. An entry of the form **rand(a..b)()** outputs a random integer in the range **a..b**.

Rank A procedure in the **LinearAlgebra** package. Returns the rank of a matrix. Syntax: Rank(A)

```
> Rank(<<1,2,3>|<4,5,6>|<7,8,9>>); output: 2
```

ReducedRowEchelonForm

A procedure in the **LinearAlgebra** package. Returns the reduced row echelon form of a matrix. Syntax: ReducedRowEchelonForm(A)

```
> ReducedRowEchelonForm(<<1,2,3>|<4,5,6>|<7,8,9>>);
```

output: <<1,0,1,>|<0,1,0>|-1,2,0>>

rem Polynomial remainder function: rem(f,g,x) is the remainder when polynomial f is divided by polynomial g (rational coefficients). The entry rem(f,g,x,'q') computes the remainder and stores the quotient with the name q. Inert form Rem, used in conjunction with mod p (p a prime), tells Maple to calculate over the ring of integers mod p.

```
> rem(x^4+x^3+x+1,x^2-1,x);   Output: 2 + 2x
> rem(x^4+x^3+x+1,x^2-1,x,'q'): q;   Output: 1 + x + x^2
> Rem(x^4+x^3+x+1,x^2-1,x) mod 2;   Output: 0
```

remember An option in a procedure. Tells Maple to remember values that it has calculated. They are stored in a table and used if needed in a subsequent calculation.

remove Removes elements from and expression based on a boolean function criterion.
Syntax: remove(boolean,expression).

```
> remove(t->is(degree(t,x)>3),[x^2+x,x-x^4,x,x^5+3]);
```

Output: [x^2 + x, x]

return An execution control command used only in procedures. The entry **return** tells Maple to immediately leave the procedure and output what follows the word **return**. See **break** and **next**.

```
> f:=proc(n) local k;
  for k from n+1 do if isprime(k) then return k end if; end do;
  end proc;
> f(7);   Output:  11.
```

rhs Right hand side procedure. Outputs the right hand side of an equation.

```
> rhs( E = m*c^2);
```

output: m c^2

right Used to make a right hand limit.

> limit(sin(x)/x^2, x=0, right); output: infinity

RootOf — A function that represents the roots (zeros) of an expression. Often appears in the output of the solve procedure to represent an exact value.
> solve(tan(x) = x^2, x); output: RootOf(-tan(_Z) + _Z^2)

$ — The sequence operator. Used to make sequences. Syntax: f(k) $ k=a..b
> k/(k^2+1) $ k=-1..3; output: -1/2, 0, 1/2, 2/5, 3/10

select — Selects elements in and expression based on a boolean function criterion.
Syntax: select(boolean,expression).
> select(t->is(degree(t,x)>3),[x^2+x,x-x^4,x,x^5+3]);
Output: [x - x^4, x^5 + 3]

selectremove — Selects and removes elements in and expression based on a boolean function criterion.
Syntax: selectremove(boolean,expression).
> selectremove(t->is(degree(t,x)>3),[x^2+x,x-x^4,x,x^5+3]);
Output: [x - x^4, x^5 + 3], [x^2 + x, x]

seq — The sequence procedure. Used to make sequences. Syntax: seq(f(k), k=a..b)
> seq(k/(k^2+1), k=-1..3); output: -1/2, 0, 1/2, 2/5, 3/10
> seq(k/(k^2+1), k=[a,3,car]); output: a/(a^2+1), 3/10, car/(car^2+1)

sequence — A family of expressions, separated by commas.

series — Generates a series representation of an expression. Syntax: series(expression, variable=value). The default order of the representation is 6.
> series(BesselJ(0,x), x=0);

setoptions — A procedure in the plots package used to set global options for the plots in a worksheet.
Syntax: setoptions(color=black, font=[times,roman,10])

simplify — Applies standard simplification rules to an expression. Use it as a first attempt at simplifying a complicated expression.
> simplify(cos(x)*(x^2 + 2*x + 1)/(x+1)); output: (x+1) cos(x)

sincos — Keyword for **convert**. Converts trig expressions to sines and cosines.
> convert(tan(x)^2 + sec(x)^2, sincos);

single quotes — Single quotes ' ' are used in Maple to postpone evaluation one time.
> 'int(sin(x), x=0..1)'; value(%);
output: line 1: unevaluated integral, line 2: - cos(1) + 1

Sketch — A sketch pad can be inserted into a Maple worksheet. See the **Insert** menu.

solve — A procedure for finding exact solutions to an equation in one variable.
> solve(x^3 - x - 1 = 0, x); output: 0, 1/2 + sqrt(5)/2, 1/2 - sqrt(5)/2

sort Sorts a polynomial in decreasing order of degree. Sorts a list of strings lexicographically. Add an optional binary boolean function to determine a sort criterion for a list.
`> sort(1+x^3-2*x^2);` Output: x^3 - 2x^2 + 1
`> sort( [1,2,x,1-x^2,1/x],(u,v)->is(degree(u)>degree(v)));`
Output: [1-x^2, x, 2, 1, 1/x]

spacecurve Plots parametrized curves in 3 dimensions. In the plots package.
Syntax: spacecurve([f(t),g(t),h(t)], t=a..b)
Use plot3d for parametrized surfaces in 3 dimensions.

Spellcheck On the **Tools** menu. Use it.

sphereplot Plots in spherical coordinates. In the plots package. Plots rho as a function of theta and phi.
Syntax: sphereplot(rho(theta,phi), theta=a..b, phi=c..d)

Spreadsheet A spreadsheet can be inserted into a Maple worksheet. See the **Insert** menu.

stepsize Use the optional equation **`stepsize=h`** to control the step size for **`dsolve(type=numeric)`** and **`DEplot`**.

string Any number of typed symbols enclosed in double quotation marks.
> "How are you? 123 cos(x)"; output: "How are you? 123 cos(*x*) "

subs A procedure used to make substitutions. Syntax: subs(equations,expression). See algsubs and subsop.
`> subs(x=3,y=2,x^2 + x*y);` output: 15

sum The procedure to use when adding an infinite number of numbers, functions, etc. Inert form is **`Sum`**.
Syntax: sum(f(k), k=1..infinity). Also works for finite sums but is not recommended.
`> sum( 1/k/(k+1), k=1..infinity);` output: 1
`> sum( x^k/k!, k=0..infinity);` output: exp(x)

table A data structure like an array but more flexible. The indices can be a number or a name. An entry of the form T[1,2] := expression tells Maple to set up a table with the name T and start to fill it with expression stored with index (1,2).

taylor A procedure that generates the Taylor series representation of an expression.
Syntax: taylor(expression, variable=value). The default order of the representation is 6.
`> taylor( sin(x), x=0);` output: x - x^3/6 + x^6/120 + O(x^6)

textplot Creates text that can be displayed using display. In the plots package.
Syntax: textplot([a,b,"text"]) or, for 3-d, textplot([a,b,c,"text")
Accepts optional equations for font control and alignment. Default is align=CENTER.

thickness The optional equation **`thickness=2`** tells Maple to draw thicker lines in a 2 dimensional plot. The default thickness is 1. Thickness values can be 1, 2, 3, or 4.

tickmarks An optional equation for the plot procedure: **`tickmarks=[m,n]`** sets the minimum number of values

for the horizontal and vertical axes. Maple may refuse if m or n is too large.

time A procedure for timing a Maple calculation. Syntax: time(input). The entry time( ) returns the total computing time from the beginning of the current session.
> time(plot(sin(1/x), x=0..1, numpoints=300)); output: 0.580
> time(); output: 6.060

transparency The optional equation transparency=n where n is between 0 and 1 controls the transparency of a 3-d surface. Default: transparency=0.

Transpose A procedure in the **LinearAlgebra** package. Transposes a matrix. Syntax: Transpose(A)
> Transpose(<<1,2>|<3,4>>); output: <<1,3>|<2,4>>

trace A Maple debugging tool, **trace(proc_name)** causes Maple to display the whole sequence of outputs when **proc_name** is invoked. Use **untrace(proc_name)** to stop tracing. See **printlevel**.

tubeplot Plots tubular surfaces. In the plots package. Control the radius of the tube with radius=r(t).
Syntax: tubeplot([f(t),g(t),h(t)], t=a..b, radius=r(t),

type Used to inquire about the data type of an expression. Output is true or false (binary boolean)
Syntax: type(3, integer); output: true

unapply The procedure to use to turn an expression into a function.
Syntax: unapply(expression, variables)
> g := unapply(x*sin(x*y), x, y); Output: g := (x,y) -> x sin(xy)

unassign A procedure that is used to unassign values. Syntax: unassign('y')
> y := sin(t): y = z; unassign('y'); y + x;
output: line 1: sin(t) = z, line 2: y + x
The assignment y := 'y' has the same effect as unassign('y').

union Set union. Syntax: A union B.
> {1,2,3} union {2,3,4}; output: {1,2,3,4}.

value A procedure used to ask Maple the value of an expression or a name. Note the backward quotes in the second input.
> X := 'int(sin(x), x)': value(X); output: - cos(x)
> X := `int( sin(x), x)`: value(X); output: int(sin(x), x)

Vector A procedure that makes column vectors. Syntax: Vector([a,b,...,c]).
Such a vector can also be created by entering **<a,b,...,c>**. To create a row vector use either **Vector[row]([a,b,...,c])** or **<a|b|...|c>**.

verboseproc A Maple variable whose value determines the amount of information that Maple outputs when a procedure is evaluated via eval(procedure) or print(procedure). The default value is 1.

whattype A procedure that is used to ask Maple the data type of an expression. Syntax: whattype(expression)

> whattype(3), whattype(3.0), whattype([1,2]); output: *integer, float, list*
Also whattype(eval(expression)).

while Used to make a repetition statement.
> for k from -5 to 10 by 2 while not isprime(k) do k end do;
Output: The numbers -5, -3, -1, 1 on separate output lines.

xtickmarks An optional equation for the plot procedure: **xtickmarks=m** tells Maple to put at least m numbers on the horizontal axis. If m is too large, Maple will refuse.

ytickmarks An optional equation for the plot procedure: **ytickmarks=n** tells Maple to put at least n numbers on the vertical axis. If n is too large, Maple will refuse.

zip Procedure used to merge lists, vectors, matrices. The merge criterion is entered as a binary function.
> zip((x,y)->x/(y+x),[1,2,3],[a,b,c]); Output: [1/(a+1), 2/(b+2), 3/(c+3)]