

Maple 7: A Quick Reference

Prepared by:
Douglas Meade
Department of Mathematics
University of South Carolina

January 2002, updated for Maple 7
(Earlier editions for Maple 6, Maple V, Release 5, and Maple V, Release 4)

Symbols and Abbreviations

| Symbol | Description | Example |
|---|---|---|
| <code>:=</code> | assignment | <code>f := x^2/y^3;</code> |
| <code>;</code> | terminate command; display result | <code>int(x^2, x);</code> |
| <code>:</code> | terminate command; hide result | <code>int(x^2, x):</code> |
| <code>..</code> | specify a range or interval | <code>plot(t*exp(-2*t), t=0..3);</code> |
| <code>{ }</code> | set delimiter (a set is an unordered list) | <code>{ y, x, y };</code> |
| <code>[]</code> | list delimiter (lists are ordered) | <code>[y, x, y];</code> |
| <code>%</code> | refers to previous result (percent) <i>Note:</i> Was " until Maple V, Release 5 | <code>Int(exp(x^2), x=0..1):</code> <code>% = evalf(%);</code> |
| <code>" "</code> (see <code>?strings</code>) | string delimiter (double quote) <i>Note:</i> Changed in Maple V, Release 5 (see <code>%</code>) | <code>plot(sin(10*x) + 3*sin(x), x=0..2*Pi,</code> <code style="padding-left: 2em;">title="An interesting plot");</code> |
| <code>` `</code> (see <code>?names</code>) | name delimiter (back quote) | <code>`A name` := `This is a name.`;</code> |
| <code> </code> (see also <code>?cat</code>) | concatenate string or name <i>Note:</i> Was . prior to Maple 6 | <code>a 3;</code> <code>a (1..3);</code> |
| <code>' '</code> (see <code>?uneval</code>) | delayed evaluation (single quote) | <code>x := 'x';</code> |
| <code>-></code> (see <code>?-></code> and <code>?proc</code>) | mapping (procedure) definition | <code>f := (x,y) -> x^2*sin(x-y);</code> <code>f(Pi/2,0);</code> |
| <code>@</code> | composition operator | <code>(cos@arcsin)(x);</code> |
| <code>@@</code> | repeated composition operator | <code>(D@@2)(ln);</code> |

Mathematical Operations, Functions, and Constants

| Symbol | Description | Example |
|---|---|---|
| <code>+</code> , <code>-</code> , <code>*</code> , <code>/</code> , <code>^</code> | add, subtract, multiply, divide, power | <code>3*x^(-4) + x/Pi;</code> |
| <code>sin</code> , <code>cos</code> , <code>tan</code> , <code>cot</code> , <code>sec</code> , <code>csc</code> | trigonometric functions | <code>sin(theta-Pi/5) - sec(theta^2);</code> |
| <code>arcsin</code> , <code>arccos</code> , <code>arctan</code> , <code>arccot</code> , <code>arcsec</code> , <code>arccsc</code> | inverse trigonometric functions | <code>arctan(2*x);</code> |
| <code>exp</code> | exponential function | <code>exp(2*x);</code> |
| <code>ln</code> | natural logarithm | <code>ln(x*y/2);</code> |
| <code>log10</code> | common logarithm (base 10) | <code>log10(1000);</code> |
| <code>abs</code> | absolute value | <code>abs((-3)^5);</code> |
| <code>sqrt</code> | square root | <code>sqrt(24);</code> |
| <code>!</code> | factorial | <code>k!;</code> |
| <code>=</code> , <code><></code> , <code><</code> , <code><=</code> , <code>></code> , <code>>=</code> | equations and inequalities <i>Note:</i> <code>E</code> no longer exists; use <code>exp(1)</code> | <code>diff(y(x), x) + x*y(x) = F(x);</code> <code>exp(Pi) > Pi^exp(1);</code> |
| <code>Pi</code> , <code>I</code> | π , i (mathematical constants) <i>Note:</i> Maple is case-sensitive | <code>exp(Pi*I);</code> |
| <code>infinity</code> | infinity (∞) | <code>int(x^(-2), x=1..infinity);</code> |

NOTES:

- The document is also available on the World Wide Web in either PDF (<http://www.math.sc.edu/meade/maple/maple-ref.pdf>) or PostScript (<http://www.math.sc.edu/meade/maple/maple-ref.ps>).
- Please send comments, corrections, and suggestions for improvements to meade@math.sc.edu.

Commands

| Command | Description | Example |
|--|---|---|
| <code>restart</code> | clear all Maple definitions | <code>restart:</code> |
| <code>with</code> | load a Maple package | <code>with(DEtools); with(plots):</code> |
| <code>help</code> (also <code>?</code>) | display Maple on-line help | <code>?DEplot</code> |
| <code>limit</code> | calculate a limit | <code>limit(sin(a*x)/x, x=0);</code> |
| <code>diff</code> | compute the derivative of an expression | <code>diff(a*x*exp(b*x^2)*cos(c*y), x)</code> |
| <code>int</code> | definite or indefinite integration | <code>int(sqrt(x), x=0..Pi);</code> |
| <code>Limit</code> | inert (unevaluated) form of <code>limit</code> | <code>Limit(sin(a*x)/x, x=0);</code> |
| <code>Diff</code> | inert (unevaluated) form of <code>diff</code> | <code>Diff(a*x*exp(b*x^2)*cos(c*y), x);</code> |
| <code>Int</code> | inert (unevaluated) form of <code>int</code> | <code>Int(sqrt(x), x=0..Pi);</code> |
| <code>value</code> | evaluate an inert expression (typically used with <code>Limit</code> , <code>Diff</code> , or <code>Int</code>) | <code>G := Int(exp(-x^2), x);</code> <code>value(G);</code> |
| <code>plot</code> | create a 2-dimensional plot | <code>plot(u^3, u=0..1, title="cubic");</code> |
| <code>plot3d</code> | create a 3-dimensional plot | <code>plot3d(sin(x)*cos(y),x=0..4*Pi,y=0..Pi);</code> |
| <code>display</code> | combine multiple plot structures into a single plot or modify optional settings in a plot (in <code>plots</code> package) | <code>F:=plot(exp(x), x=0..3, style=line);</code> <code>G:=plot(1/x, x=0..3, style=point);</code> <code>plots[display]([F,G], title="2 curves");</code> |
| <code>solve</code> | solve equations or inequalities | <code>solve(x^4 - 5*x^2 + 6*x = 2, { x });</code> |
| <code>fsolve</code> | solve using floating-point arithmetic | <code>fsolve(t/10 + t*exp(-2*t) = 1, t);</code> |
| <code>dsolve</code> | solve ordinary differential equations; see <code>?dsolve</code> for a list of available options | <code>dsolve(diff(y(x),x)-y(x)=1, y(x));</code> |
| <code>odeplot</code> | create 2D and 3D plots from solutions obtained by <code>dsolve</code> (with <code>type=numeric</code>); see <code>?odeplot</code> for more options (in <code>plots</code> package) | <code>S:=diff(x(t),t)=-y(t),diff(y(t),t)=x(t):</code> <code>IC:=x(0)=1,y(0)=1:</code> <code>P:=dsolve({S,IC}, {x(t),y(t)}, numeric):</code> <code>odeplot(P, [[t,x(t)],[t,y(t)]], 0..Pi);</code> <code>odeplot(P, [x(t),y(t)], 0..Pi);</code> |
| <code>DEplot</code> | create plot associated with an ODE or system of ODEs; see <code>?DEplot</code> for more information (in <code>DEtools</code> package) | <code>ODE := diff(y(x),x) = 2*x*y(x);</code> <code>DEplot(ODE, [y(x)], x=-2..2,</code> <code> y=-1..1, arrows=SMALL);</code> |
| <code>D</code> | differential operator (often used when specifying derivative initial conditions for <code>dsolve</code>) | <code>ODE := diff(y(x),x\$2) + y(x) = 1;</code> <code>IC := y(0)=1, D(y)(0)=1;</code> <code>dsolve({ ODE, IC }, y(x));</code> |
| <code>simplify</code> | apply simplification rules to an expression | <code>simplify(exp(a+ln(b*exp(c))));</code> |
| <code>factor</code> | factor a polynomial | <code>factor((x^3-y^3)/(x^4-y^4));</code> |
| <code>convert</code> | convert an expression to a different form | <code>convert(x^3/(x^2-1), parfrac, x);</code> |
| <code>collect</code> | collect coefficients of like powers | <code>collect((x+1)^3*(x+2)^2, x);</code> |
| <code>rhs</code> | right-hand side of an equation | <code>rhs(y = a*x^2 + b);</code> |
| <code>lhs</code> | left-hand side of an equation | <code>lhs(y = a*x^2 + b);</code> |
| <code>numer</code> | extract the numerator of an expression | <code>numer((x+1)^3/(x+2)^2);</code> |
| <code>denom</code> | extract the denominator of an expression | <code>denom((x+1)^3/(x+2)^2);</code> |
| <code>subs</code> | substitute values into an expression | <code>subs(x=r^(1/3), 3*x*ln(x^3));</code> |
| <code>eval</code> | evaluate an expression with specific values | <code>eval(3*x*ln(x^3), x=r^(1/3));</code> |
| <code>evalf</code> | evaluate using floating-point arithmetic | <code>evalf(exp(Pi^2));</code> |
| <code>evalc</code> | evaluate a complex-valued expression (returns a value in the form <code>a+I*b</code>) | <code>evalc(exp(alpha+I*omega));</code> |
| <code>evalb</code> | evaluate a Boolean expression (returns <code>true</code> or <code>false</code> or <code>FAIL</code>) | <code>evalb(evalf(exp(Pi) > Pi^exp(1)));</code> |
| <code>assign</code> | perform assignments (often used after <code>solve</code> or <code>dsolve</code>) | <code>S:=solve({x+y=1, 2*x+y=3}, {x,y});</code> <code>assign(S); x; y;</code> |
| <code>seq</code> | create a sequence | <code>seq([0,i], i=-3..3);</code> |
| <code>for ... from ...</code> <code>to ... by ... in ...</code> <code>while ... do</code> <code>... end do</code> | repetition statement; see <code>?do</code> for syntax (Note: <code>od</code> is an acceptable substitute for <code>end do</code>) | <code>tot := 0;</code> <code>for i from 11 by 2 while i < 100 do</code> <code> tot := tot + i^2</code> <code>end do;</code> |
| <code>if ... then ... elif</code> <code>... else ... end if</code> | conditional statement; see <code>?if</code> for syntax (Note: <code>fi</code> is an acceptable substitute for <code>end if</code>) | <code>if type(x,name) then 'f'(x) else x+1 end if;</code> |
| <code>assume</code> | inform Maple of additional properties of objects | <code>assume(t>0);</code> |
| <code>about</code> | check assumptions on Maple objects | <code>about(t);</code> |