

Mathematica Quick Reference

- To execute *Mathematica* code, press **Shift-Enter** on the keyboard or just **Enter** on the number pad.
- Also, *Mathematica* is **case-sensitive**. That means capital and lower-case letters matter.
- You may edit previously executed code by moving the mouse to it and making changes as you would in a word processor. However, the new output will erase the original output.

Basic Calculations

Mathematica does all of the basic functions of a calculator.

Addition: $3 + 4$ Subtraction: $3 - 4$ Multiplication: $3 * 4$ or $3(4)$ or $3 \text{ space } 4$ Division: $3 / 4$

Square Root: **Sqrt**[34] Exponent: 3^4 Parentheses: Use (and) only

Trigonometry: **Cos**[Pi/3], **Sin**[Pi/3], **Tan**[Pi/3], **ArcCos**[1/2], **ArcSin**[1/2], **ArcTan**[Sqrt[3]],
Sec[Pi/3], **Csc**[Pi/3], **Cot**[Pi/3], **ArcSec**[2/Sqrt[3]], etc.

Default is radians. To use degrees: **Cos**[60 Degree]

Factorial: **4!** Absolute Value: **Abs**[expression]

Exponential Function (*e*): **Exp**[expression] or **E**^(expression)

Natural Logarithm (ln): **Log**[expression]

Note: *Mathematica* gives you exact answers—fractions and reduced roots.

To get a numerical approximation use the function: **N**[expression] or expression //N

Example: **N**[Pi] or **N**[Pi, 50] to approximate pi to 50 decimal places.

Simplifying and Solving Equations

Mathematica will expand and factor expressions, reduce trigonometric expressions, and solve equations.

Simplifying

Expand[(x+3)(x-4)]

Factor[x² - x - 12]

Cos[x] / **Tan**[x] - **Sin**[ArcCos[x]]

Solving — *Be sure to use two equals signs.*

Solve[x + 4 == 2x - 3]

Solve[x² + 5x + 7 == 0]

Solve[Log[2x] == 2Log[3x], x]

Solve[x + 4 - 3y == 2x - 3 + y/3, y] solves for y in terms of x.

Sometimes an error message is produced with your result: **Solve**[3^x == 5, x]

NSolve[3^x == 5, x] gives you a numerical approximation to the result.

Defining a Function

To define a function $f_1(x)$, follow the example. Be sure to use an underscore “_” after the variable.

f1[x_] = x² + 1

You can now use this to calculate several values.

f1[3] produces the output “10” as expected.

As a sample mini-program:

f1[x_] = x² + 1;

f2[x_] = x³ - 1;

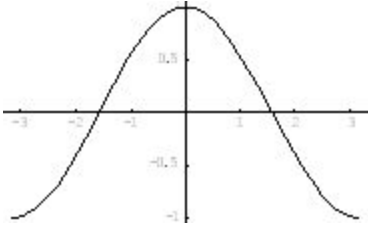
f1[3] + **f2**[2]

The semi-colon suppresses the intermediate output.

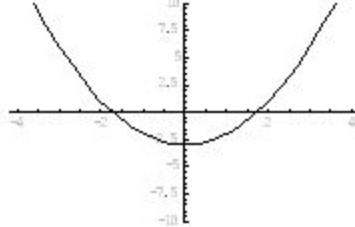
Graphing Curves and Surfaces

Below is some sample code you can follow to create some basic graphs.

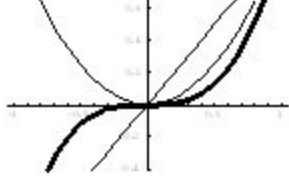
`Plot[Cos[x], {x, -Pi, Pi}]`



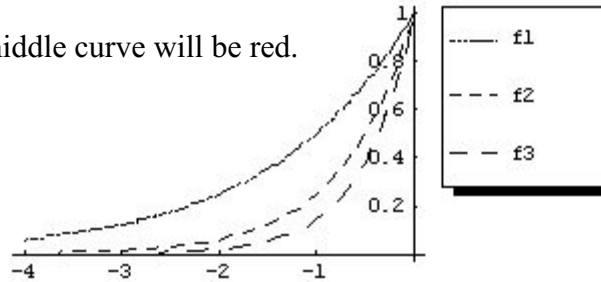
`Plot[x^2 - 3, {x, -4, 4}, PlotRange -> {-10, 10}]`



`Plot[{x, x^2, x^3}, {x, -1, 1}, PlotStyle -> {Thickness[.005], {RGBColor[1,0,0], Thickness[.01]}, Thickness[.02]}`



The middle curve will be red.



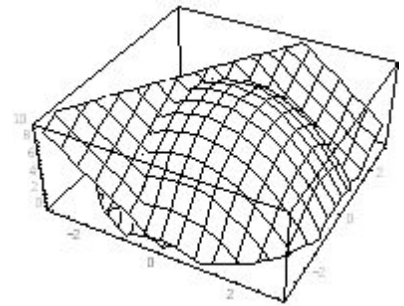
`<<Graphics`Legend``

`f1[x_] = 2^x;`

`f2[x_] = 4^x;`

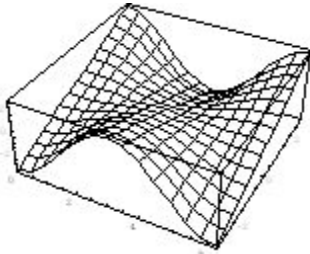
`f3[x_] = 7^x;`

`Plot[{f1[x], f2[x], f3[x]}, {x, -4, 0},
PlotStyle -> {Dashing[.01], Dashing[.03], Dashing[.05]},
PlotLegend -> {"f1", "f2", "f3"}, LegendPosition -> {1, -0.2}]`



In three dimensions:

`Plot3D[y*Cos[x], {x, 0, 2Pi}, {y, -3, 3}]`



You can show two graphs at the same time using “Show” in 2D or 3D.

`graph1 = Plot3D[10 - x^2 - y^2, {x, -3, 3}, {y, -3, 3};`

`graph2 = Plot3D[-3x + 2y + 6, {x, -3, 3}, {y, -3, 3};`

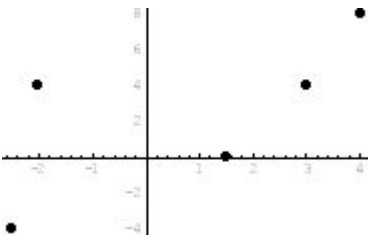
`Show[graph1, graph2, PlotRange -> {0, 10}]`

Graphing Coordinate Points

To plot a list of coordinates, type it as below. Use “PointSize” to control the size

of the points.

`ListPlot[{{3, 4}, {-2, 4}, {4, 8}, {1.5, 0}, {-2.5, -4}}, PlotStyle -> PointSize[0.05]]`



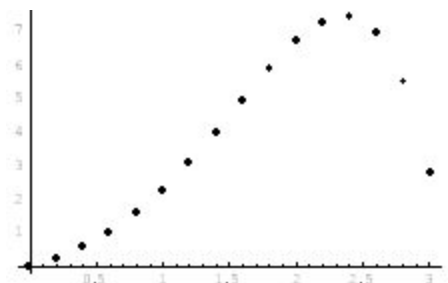
To plot the points of a function, define it and use “Table” as below.

`f[x_] = Exp[x]Sin[x];`

`ListPlot[Table[{x, f[x]}, {x, 0, 3, 0.2}], PlotStyle -> PointSize[0.03]]`

The “{x, 0, 3, 0.2}” tells it to make a point for x every 0.2 units between 0 and 3.
If you use

`PlotStyle -> {PointSize[0.02],`



`RGBColor[0.5, 0, 1]]`

the points will be purple.

R=amount of red, G=amount of green, B=amount of blue (#s between 0 & 1)