**Chapter 5 Short Summary**

\* *Strings*, *cat()*

Start and end a string with double-quotes.

Strings can be joined together with the *cat()* function, e.g.

**> **



**> **



**> **



(Remark: Maple provides sophisticated string manipulation tools called StringTools)

\* Creating a *2-D Plot*

-> Method 1) Creating a *Smart plot* using just the mouse.

Insert Menu -> Plot -> 2-D, inserts an empty default plot:



Plot a graph by copying (or dragging and dropping) the expression into the plot window. (If the expression is an input, hold CTRL key before dragging to copy it.)

e.g. ****



Change various plotting options in the context bar.

To add a title: plot menu -> title -> add title

To add a caption: plot menu -> title -> add caption

-> Method 2) The *plot()* function

(plot() accepts several forms of input.)

First argument:

Can plot an expression involving one indeterminate.

(Remark: *plot()* does not accept equations in its first argument so only include expressions in the independent variable)

(Remark: You can reconfigure this interactively like smart plot, except the image is not resized when ranges are edited.)

Second argument: (optional)

Horizontal axis range of the graph, e.g.

**> **



Third argument: (optional)

Specify the name and range for the vertical axis with an equation,

e.g. 

(Remark: You can specify just the name or just the range.)

More arguments: (optional)

1) *discont*

*discont* prevents Maple from drawing a vertical line at asymptotes.

e.g. (By default Maple regards all expressions as continuous.)

Graph of :



**>** 



(Remark: Avoiding discontinuities can be done only using the plot function and not in a smart plot.)

2) Plot annotation (*title*/*caption*/*legend*)

e.g. 



You can mix strings and mathematical expressions using the typeset function,

e.g. *title* = *typeset*(“Graph of ”, sin2(θ)).

3) *scaling=constrained*

This ensures the axis have the same scaling. (Important for geometrical plotting)

4) *view*

Axis ranges can be set more generally using the *view* option.

E.g. putting **** as an argument sets the x and y axis accordingly:



5) axes=none

This removes display of axis.

6) style=surface

Used to remove wireframe lines, (to make a 3-D plot look nicer).

\* *Plotting Points* and *Line segments*

-> Plotting Points (functions that map integers to integers)

Method 1) Using plot():

e.g.

List of lists of the first 10 values of ithprime:

**> **



**> **



Without the option *style=point*, plot() would have joined the points with straight line segments, by default.

(Remark: It is better to use a list when plotting points rather than a set.)

Method 2) Smart Plot:

Insert a 2-D plot, then copy or drag the list of coordinate pairs into it. Then change the style interactively to remove connecting lines between the points.

-> *Geometrical Plotting*(Polygons)

Plot points with plot() to create a polygon as a sequence of line segments.

e.g.

(Remark: Save typing, using the fact that Maple can multiply a list by a number, e.g.)

**> **



**> **



**> **



(Remark: Note the need to include the first point of a polygon, also as the last point in order to plot a closed polygon.)

\* *Multiple Plots*

-> Method 1) You can use the smart plot to plot multiple graphs. Just drag each equation onto the smart plot.

-> Add a legend: Plot menu -> Legend -> Show Legend.

-> Method 2) To plot two or more graphs using the plot() function, put all the expressions to be plotted into a list.

Optional arguments can then be specified using lists whose elements appear in the same order as the expressions.

e.g. (y is enclosed in uneval quotes as a precaution)

**> **



\* *Assigning* (labels to) *Plots*

To assign and display a plot, use e.g.

**> **



The colon after ****, supresses the output and also serves as a statement separator, so following it by another statement Q, prints the graph.

(Remark: Notice that just inputting something like ****produces useless output i.e. , since plot data structures are not displayed when part of another data structure, (in this case an assignment data structure))

\* *Packages*

-> Specialised maple functions are provided in packages. You can access a particular function in a particular package as either .

(Remark: The syntax *package[function]*, which is treated as a variable that is evaluated, also works. Better to use notation.)

-> *with()*

Make all the functions in a package available by applying the *with()* function to a package name. This returns a list of all the functions that are exported by the package, e.g.

**> **



After calling *with()*, you can use all the functions in the package without explicitly including the package name.

\* The *plots Package*

-> A package for plotting functions is called *plots* (not *plot()*, which is a function).

-> A function in the plots package is called *display()*. The display function can merge multiple plots into one plot and also change some plotting options.

Example: using *display()* to combine plot functions:

**> **



**> **



**> **



( accepts many plotting options that are accepted by all other plotting functions, such as *title*. However it does not accept *legend*, and it cannot change the colour of a plot in the function. However, it is possible to change plot colours and add a legend interactively.)

\* *Implicit plot*

Method 1) Use smart plot as usual (to plot the implicit function)

Method 2) Use *plots:-implicitplot*

This requires both indeterminates and the axis ranges to be specified. E.g.

**> **



\* *3D Plot*

Method 1) 3-D smart plot

Insert menu->Plot->3-D, and drag the expression into plot as usual.

(Remark: The mouse pointer is used to change viewing angle by default. Angles, ϑ, φ, ψ, in the context bar, describe orientation. The remaining controls are quite similar to those for 2-D plot.)

Method 2) *Use plot3d()*

To plot an expression,  requires at least three arguments: the expression as its first argument, and the names and ranges of the indeterminates in the form of equations as the second and third arguments.

(Remarks: plot3d() accepts several different forms of input. It can be configured interactively like a 3-D smart plot.)

e.g.

**> **



\* *Drawing*

You can draw inside a drawing canvas:

Insert menu->Canvas

Hold down the Shift key while inserting a shape to preserve its aspect ratio.

(Remarks: Any plot also provides a canvas on which you can draw. Add drawings using the Drawing menu in the context bar if displayed, otherwise use the drop-down menu at the right. You can add text to annotate plots or drawings by selecting the text tool denoted by a T in a box.)

\* *Animation*

Animations add the dimension time, to plots.

Method 1) Construct a plot that depends on a parameter.

e.g.

To animate a plot, say ****, with animation parameter t, we use:

**> **



The first argument of  must be a plotting function (or, more generally, a mapping that returns a plot structure), the second argument must be a list of arguments to which the plotting function will be applied, the third argument must be the animation parameter and range in the form of an equation, and any other arguments are normal plotting options.

(Remark: By default the plot title is the continuously changing value of the animation parameter.)

(Remark: The animation options become available when you click on the plot window.)

Method 2) Construct a list of plots that represent the frames of an animation and then use  to display them in sequence.

(This is how  works internally. Details not given here.)

-> (Remark: An example of geometric animation (a rotating square generated using a *plottools* package and animation), is given in the lecture notes.)

-> 3-D plots can be easily animated using the  option.

e.g. rotate a 3-D plot left or right using  (or "circleright"),

**> **

