Semester 1 Reference Guide

Plots

In[54]:=
```
Plot[f, {x, xmin, xmax}]
```

In[55]:=
```
Plot[{f1, f2}, {x, xmin, xmax}]
```

In[56]:=
```
Plot[f, {x, xmin, xmax}, PlotLabel → "Text"]
```

Text Input

Word document style text

Select '+' for a new line then then select 'Plain text on the drop down menu'.

Text within code

Hidden text

Type text within (*these brackets*) to prevent it being 'seen' by the computer when code evaluates

Seen text

Text within "these" will be output as text, for example a title on a graph or units when used as part of a line of code

"Variables" text

Text within code that is not surrounded with either (*this*) or "this" will be treated as a variable by mathematica.

Defining functions

We can define a function one of two ways. In both cases on the left hand side, I specify what variables I am using. If I want the function to evaluate after I have specified the values of variable1 and variable2 I use ' **:** then **=** ' as below:

```
FuncDelay[variable1_, variable2_] := variable1 + variable2
```

If I want the right hand side of the function to evaluate before I specify the value of variable1 and variable2 then I simply use ' = '

```
FuncEval[variable1_, variable2_] = variable1 + variable2
```

Most of the time both ways work, an example is shown in the accompanying *Mathematica* Workbook or Week 7 of Semester 1

Mathematical operations

Differentiation

```
Dt[function[x], x]
```

Integration

For an indefinite integral:

```
Integrate[function[x], x]
```

For a definite integral:

```
Integrate[function[x], {x, xmin, xmax}]
```

Differential Equations

Form of command:

```
DSolve[LHS == RHS, y[x], x]
```

Recall, it is important to use a double equals '==’. If you use a single equals '=’ and get the following error:

DSolve::deqn : Equation or list of equations expected instead of –y[x] in the first argument –y[x]. >>

Use the Remove command

In[86]:=
```
Remove[y]
```

Initial conditions can be added:

```
DSolve[{LHS == RHS, cond1, cond2}, y[x], x]
```

Animation

Manipulate

```
Manipulate[f[n], {n, nmin, nmax}]
```

```
Manipulate[f[n], {{n, ninitial}, nmin, nmax}]
```