ITS Information Technology Service

Guide 65

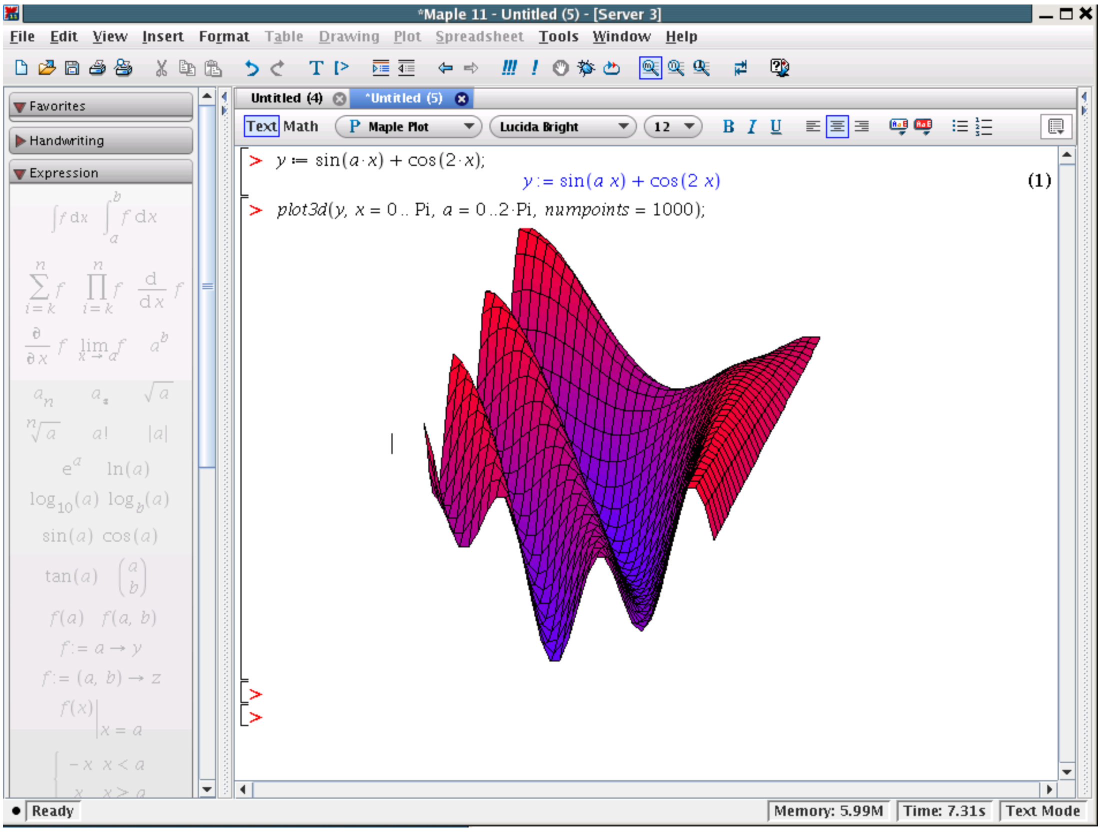
Version 5.0

Basic MAPLE: a beginner's guide

Maple is a package in which you can handle mathematical problems both symbolically and numerically. It allows you to work interactively, generate graphics and create mathematical documents. Maple is available to all users of the ITS Linux, HPC and Unix services and the ITS Networked PC service.

This guide provides an introduction to Maple's graphical user interface *xmaple* and to the Maple programming language.

Previous programming experience will be useful but is not essential. The guide is based on Linux so it is assumed that you have some experience with working in Linux or Unix. A brief section highlights the differences that Windows Maple users can expect.



Durham University

70p

Document code:	**Guide 65**
Title:	**Basic MAPLE: a beginner's guide**
Version:	**5.0**
Date:	**July 2007**
Produced by:	**University of Durham Information Technology Service**

Conventions:

In this document, the following conventions are used:

- A **bold typewriter font** is used to represent the actual characters you type at the keyboard.
- A ***slanted typewriter font*** is used for items such as example filenames which you should replace with particular instances.
- A typewriter font is used for what you see on the screen.
- A **bold font** is used to indicate named keys on the keyboard, for example, **Esc** and **Enter**, represent the keys marked Esc and Enter, respectively.
- Where two keys are separated by a forward slash (as in **Ctrl/B**, for example), press and hold down the first key (**Ctrl**), tap the second (**B**), and then release the first key.
- A **bold font** is also used where a technical term or command name is used in the text.

Contents

1 Introduction

Maple is a package in which you can handle mathematical problems both symbolically and numerically. Its graphical interface allows you to work interactively, generate graphics and create mathematical documents. Maple is available to all users of the ITS Linux, HPC and Unix services and the ITS Networked PC service.

1.1 The Basic Maple guide

This guide is designed as a tutorial in which you will learn to use the Maple language and Maple's graphical user interface, **xmaple**. It assumes that you will be working in Linux, although the Maple commands you learn can also be used in Windows. Information about running Maple on the ITS Networked PC Service can be found in section 14.

By the end of the tutorial, you should be able to use Maple to

- manipulate mathematical expressions
- solve mathematical problems
- visualise expressions in 2D, 3D and in animations
- write structured mathematical documents
- export documents in HTML and LaTeX formats

You will also have seen how to

- work with Maple spreadsheets
- use the specialist facilities available in Maple's package libraries
- work with numerical data files
- translate Maple expressions into Fortran or C

The guide is intended as an introduction and cannot cover every aspect of Maple. If you wish to learn more, please consult the resources listed in section 16.

1.2 Before you start

As you work through this guide you will generate a number of Maple files. So, before you start, create a Maple directory in which these files can be stored:

```
cd
mkdir Maple
```

These notes assume that you will run Maple from your home directory. This is the default if you run Maple from the Gnome 'foot' menu.

2 Getting started

2.1 Starting the xmaple program

There are two main ways to access **xmaple** on the ITS Linux system:

2.1.1 From the Gnome 'foot' menu

If you are working at an ITS workstation that runs the ITS Linux service, or if you log into the ITS Linux service from Windows using the **vega** item on the Durham University Network menu, you will work in the Gnome desktop, which has been customised to give you easy access to key items of software.

1. In the desktop, click on the 'foot' menu on the far left of the panel.
2. Select **Durham** from the pop-up menu, then **Mathematics** from the sub-menu, and then **Maple**. The **xmaple** window will appear on your screen.

2.1.2 From the command line

If you are working in a different desktop, for example, at a departmental workstation, it is recommended that you run **xmaple** as follows.

1. In a window showing your local workstation's prompt, type

   ssh -X xxxxx

   where **xxxxx** is the name of an ITS time-sharing computer (e.g. vega) on which you will run Maple.
2. On the time-sharing computer, type

   xmaple

If you wish to access other maple commands from the command line, you will need to create a 'terminal window' in which the environment is correct for Maple. All the Maple commands and manual pages will be available in this window. To do this,

1. On the time-sharing computer, type:

   maple.init

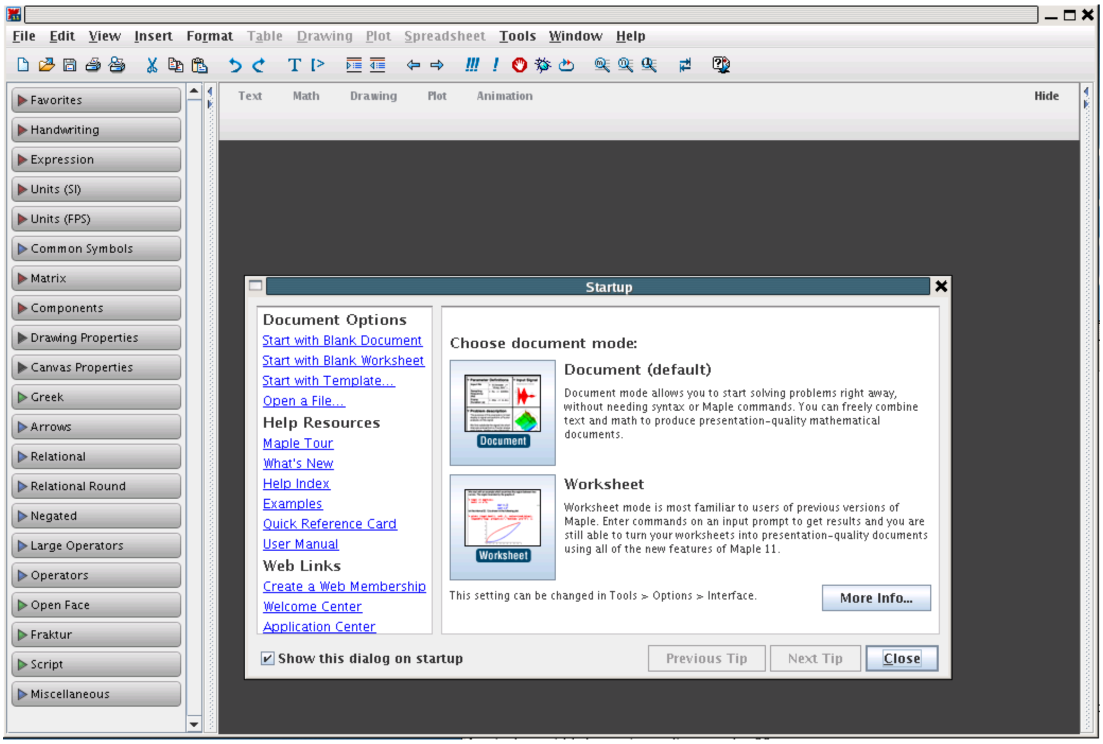
The window will be created with the title **Maple 11**.

2. Maximise the terminal window and then enter the command:

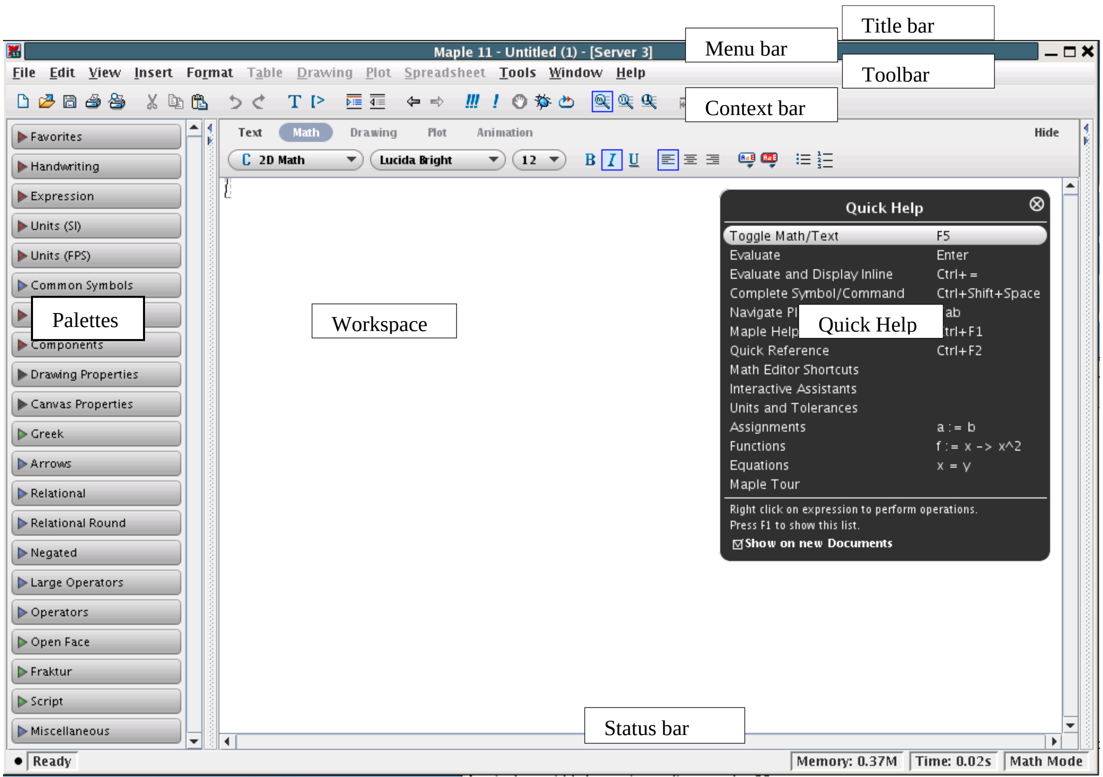
   xmaple

(You can also run a command **maple** from this window. This gives you access to 'command-line Maple', a non-graphical version of Maple that is discussed in section 13.)

2.2 Finding your way around the xmaple window



When you run Maple for the first time, a Startup dialogue box will be displayed. It offers the choice of document mode or worksheet mode. The difference between these modes is discussed in section 8 and below. For now, choose Document.



Before composing your first Maple document, take a few minutes to find your way around the **xmaple** window. It consists of four main areas:

2.2.1 The workspace

The workspace occupies most of the Maple window. It can contain one or more documents or worksheets, held as tabs, in which you will type Maple commands and see the results. The Quick Help navigator also appears in the workspace.

You may find it easier to use **xmaple** if you resize the Maple window to make it larger.

2.2.2 The palettes

The palettes give you easy access to a wide range of mathematical notation. Clicking on the triangle to the left of a palette will reveal or hide the items within that palette. Clicking on a palette item will insert it into the current document or worksheet.

2.2.3 The title and status bars

The title bar, at the top of your Maple window, gives the version of Maple that you are using.

The status bar, at the very bottom of the window, is used by Maple to display information about some operations and about your system usage.

2.2.4 The menu bar, toolbar and context bar

Directly underneath the window's title bar are three rows of buttons. The entries on these rows depend on whether you are entering text or Maple input, working on a spreadsheet or displaying graphics.

The top row is the menu bar. Some of the items on this bar will become active only in an appropriate context.

Underneath the menu bar is the toolbar. A single click on a toolbar button will execute the associated command.

Underneath the toolbar is the context bar. As you selected Document mode, you will be in the default '2-D Math' mode and the context bar will be set for this mode.

The document prompt will look like this:

/

Maple's default Document mode is useful for solving shorter or simpler mathematical problems and for generating maths-rich documents in which standard mathematical notation should be displayed. Its counterpart is worksheet mode, which is stronger as a problem-solving and programming environment. The rest of this Guide will use Worksheet mode to explore Maple's problem-solving functionality, as this is the main way in which Maple is used in Durham, but the Maple skills you will learn can also be used to write Maple documents.

To open a new worksheet,

3 Click on the File menu, select New from the sub-menu, and click on Worksheet mode.

A new tab will be opened in worksheet mode. In this mode, the prompt will look like:

[> /

4 At the worksheet prompt, type

y:=sin(a*x)+cos(2*x);

5 Press the Return key to execute the command you entered.

Notice that Maple has formatted your input as '2-D Math', which is closer to normal mathematical notation. If you prefer to see exactly what you typed, which can be useful if you need to find mistakes,

6 Highlight the expression that you typed in and right-click on it.

7 From the pop-up menu, select 2D Math, then Convert to, and click on 1D Math Input.

1D Math Input, which is also called Maple Input, is shown in red, and it appears exactly as you typed it.

8 Check that the cursor is on your line of input and press the Return key again to re-execute the command you entered.

9 Now click on the Text tool-bar button [T]. (Note: use the [T] button on the tool bar, not the button marked Text on the context bar.)

10 Notice that the document prompt has changed from

[>

to

[

1 In the context bar, click on the **bold** button **[B]**.

2 In the worksheet, type

The line above assigns an expression to the variable y.

and press **Return**.

Notice that text is 'inert' and is not executed when you press Return.

There is more information about Maple documents, worksheets and syntax in the Maple Help.

3 From the Help menu, select **Maple Help**.

4 When the Maple 11 Help window appears, search for **document** or **worksheet**.

3 An example Maple worksheet

Work through this example session to see how to

- assign variables
- perform calculus
- simplify expressions
- substitute into equations
- generate graphics
- save and print your work

Look at the first line you typed into the worksheet. It contains some of the most important elements of Maple's syntax.

```
y:=sin(a*x)+cos(2*x);
```

The symbol **:=** assigns the expression *sin(ax)+cos(2x)* to *y*. Make sure you always use **:=** instead of **=** in a statement like this, and never put a space between the two symbols (e.g. **: =**), although you can put spaces in other places, such as

```
y := sin( a*x ) + cos(2 * x) ;
```

Also note that *2x* must be typed as **2*x**, as on a calculator.

The command ends with a semi-colon (**;**) to tell Maple where your expression ends. Maple needs this because long expressions can extend over several lines on the screen.

Notes:

- Maple is case sensitive, so **Y** is not the same as **y**.
- To enter a complex number a+b*i*, type it as **a+b*I**
- To work with π, type it as **Pi**
- Use the notation **exp()** for expressions involving *e*. E.g. exp(1) is 2.71828…
- To enter a number such as 3.5×10^{-14}, type it in the format **3.5e-14**
- To split a line on the screen, type Shift-Return instead of Return.

The command and its output form a simple *'execution group'* whose extent is defined by the long bracket to the left of the prompts. As you add more commands and text lines, you can gather them into execution groups by highlighting them with the mouse and then using **Split or Join** from the **Edit** menu. All the commands in an execution group are executed when you place the cursor on an executable command within the group and type **Return**.

1 Place the cursor *below* your last execution group and click on the **[>** tool-bar button to create a new execution group in 2DMath input mode.

2 Define $z = \int y \, dx$ and evaluate *z* symbolically:

```
z := Int(y,x);
value(z);
```

Check that Maple has calculated a plausible answer.

3 Now, go back two lines and edit your input to calculate the definite integral $\int_0^{\pi} y\,dx$:

z := Int(y,x=0..Pi);

4 Press **Return** twice to execute the commands.

Maple knows that π has a numerical value but only if you type it as Pi, not PI, pi or pI.

5 Try changing **Pi** to **pi** and re-execute the worksheet. Then change it back.

6 Next, substitute a value for **a** into **z**:

subs(a=3,%);

Although the answer given by Maple is correct, it could be written much more simply.

7 Type

simplify(%);

The **%** symbol in these commands is like a ditto, so the commands effectively read **subs(a=3, value(z));** and **simplify(subs(a=3, value(z)));** Be warned that Maple does not have to execute commands in the order in which they appear on the screen and that the **%** symbol refers to the last command executed, regardless of where it appears.

Next, plot *y* in three dimensions over the chosen ranges in *x* and *a*. Maple includes a large number of plot types, such as field plots, plots in special coordinate systems, inequality plots, contour plots and animations. The Help page on **plots** gives a full list. To plot *y*, you will use the **plot3d** command:

1 Type:

plot3d(y,x=0..Pi,a=0..3,axes=frame);

Once the plot has been drawn you can edit its appearance:

2 Right-click on the plot to display the plot menu.

3 Use the menu to choose a new plot style, with boxed axes and colouring that changes only with height (z).

4 To change the angle at which the plot is viewed, hold down the left mouse button over the plot and drag the mouse gently.

To finish your worksheet, add a title and then group the commands and comments into just two execution groups:

11 Place the cursor on the very first line of input. Use the **Insert** menu to insert a **Paragraph** *before* this line.

12 Use the context bar to change the paragraph style from **Normal** to **Title.** In the worksheet, type:

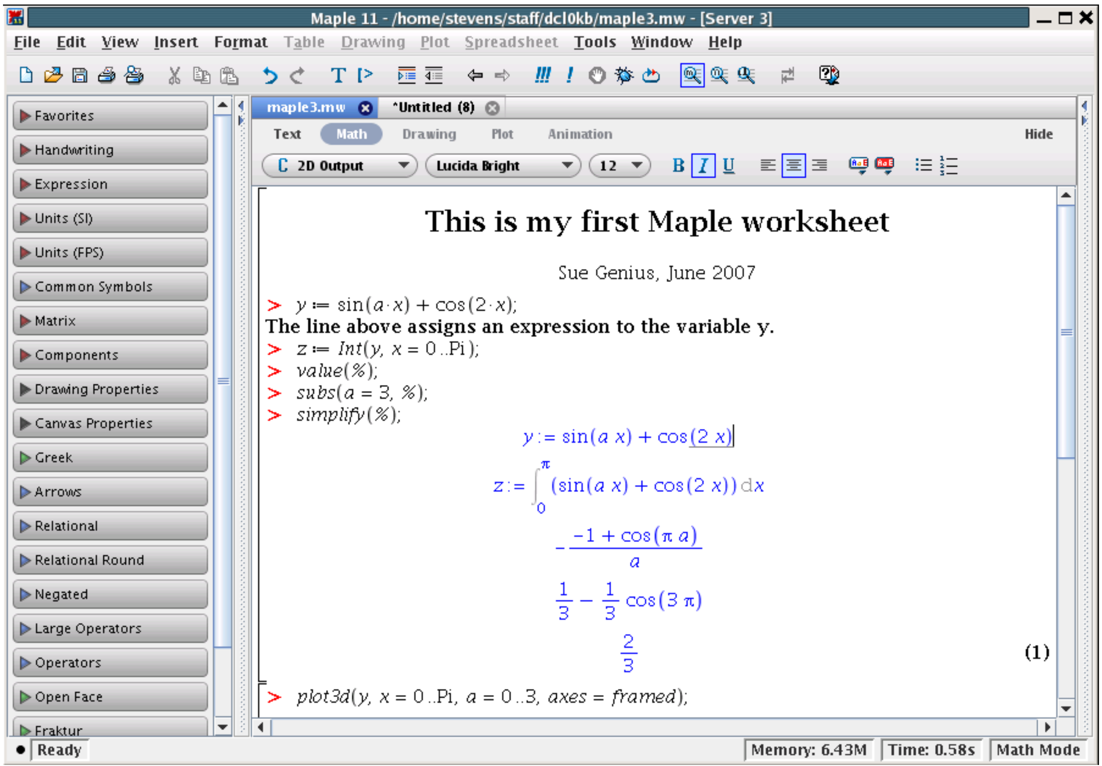
This is my first Maple worksheet.

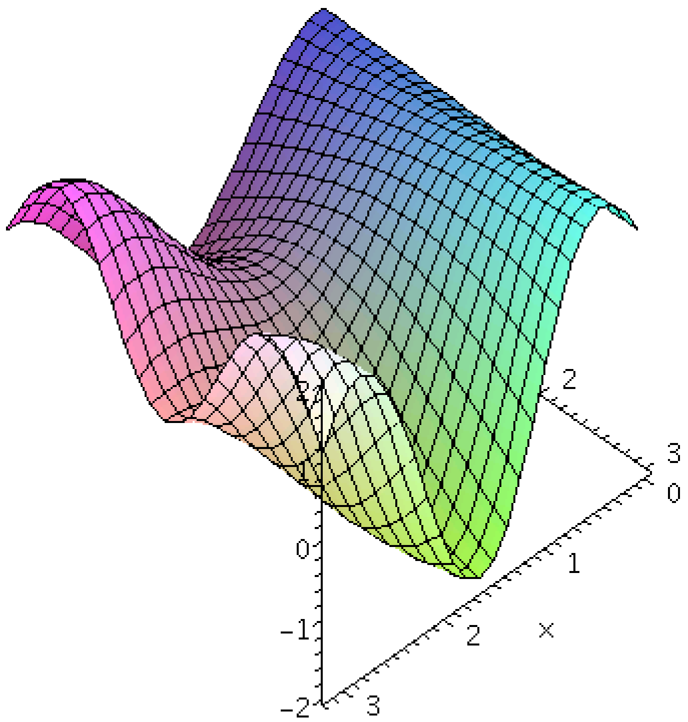
and add your name and the date on the line below in 'author' style.

13 Now group the commands: highlight all the command lines except the title and the plot command and select **Edit | Split or Join | Join Execution Groups**.

14 Place the cursor on an executable command in the new execution group and press **Return**.

The top of your completed worksheet and the plot should look similar to these:





4 Saving and printing

4.1 Saving a worksheet

1. Use **File | Save** to save your work as a file **mymaple1.mw** in your **Maple** directory. (Double-click on the directory to select it.)

The file will be in Maple worksheet format. You will be able to load it again later, complete with the displayed output lines and graphics. Note that you will need to re-run the worksheet after reloading to restore the values of variables.

4.2 Printing a worksheet

The first time you print from Maple, you will need to tell it to use a page size of A4 (rather than Letter).

1. Go to **File | Page Setup..** and change the size to **A4**.

Maple should remember this for future sessions. Now print the file:

2. Select **File | Print** to display the Print dialogue box.
3. In the **General** tab, select an appropriate printer **name** from the list.
4. In the **Page Setup** tab, check that **A4** is selected.
5. If you want to print to a colour printer, make sure that the **Color** button is ticked in the **Page Setup** box.

6 Click on **Print**.

4.3 Saving graphics

An easy way to save a single graph from a worksheet is:

1. Position the mouse over the graph and click on the right-hand button.
2. Select **Export** from the pop-up menu.
3. Choose a format for the file.
4. In the **Export** dialogue window, select a location and a name for your file, then click on **Save**.

This method does not allow you much control over the saved graph. To specify the format more precisely, or if you want to generate the file automatically, use the **plotsetup** command to specify the destination for graphics:

1. Immediately before the **plot** command in your worksheet, add the line:

```
plotsetup(ps,plotoutput=`Maple/plot1.ps`,plotoptions=`portrait,noborder`);
```

   (Make sure that the quotation marks are backquotes.)

2. Re-execute your **plot3d** command.

This will save your plot as an encapsulated PostScript file called **plot1.ps**. The page will be in portrait format and it will not have a border drawn round it.

3. To view the PostScript file, move to a Linux terminal window and then use **ghostview**. Type:

```
gv Maple/plot1.ps
```

4. The **plotsetup** command will continue to send graphics to this destination until you change it. To return to plotting on the screen, type:

```
plotsetup (default);
```

For other printing and plotting options, view the Help pages on *plot* and *device*.

5 Making your Maple work accessible to other people

If you write a worksheet or document in Maple, you may want other people to be able to use it, or to incorporate it into a longer document. This section looks at some ways of sharing your Maple work.

5.1 Making your worksheets available on the World Wide Web

Maple worksheets can be loaded directly into Maple from a URL:

1. In the **File** menu, select **Open URL...**
2. In the pop-up window, type the address of a worksheet: **http://www.dur.ac.uk/its/software/maple/wavesdemo.mws**

The worksheet will be loaded into Maple and is ready for you to work on it.

If you want to make your own worksheets available in this way, place them in your **public_html** directory.

5.2 Exporting worksheets as HTML

An alternative way to make your Maple work available on the World Wide Web is to export it as HTML. Follow the instructions below to test this on the worksheet **mymaple1.mws** you created earlier. (Re-load the worksheet if necessary, using **File | Open**.)

1. Use **File | Export As** to save the **mymaple1** worksheet in **HTML** format in your **Maple** directory.
2. In the **Export Options** dialogue box, set the image location to **mapleimages**, accept the default format (GIF) for mathematical expressions and check that the box is ticked to use Frames. Then click on **OK**.
3. View the resulting files with your usual browser. For example, in the Firefox browser, go to the **File** menu, select **Open File...**, use the File Browser window to select **Maple/mymaple1.html** and click on **Open**.

As above, if you want other people to be able to access these files, place them in your **public_html** directory.

Exporting your worksheet as HTML allows it to form part of a series of web pages and to be viewed by people who do not have access to Maple. However, creating the HTML document can result in a large number of files:

- sections and subsections are stored separately and accessed as Frames via a Table of Contents.
- a non-Frames version of the whole document is saved.
- Maple creates a subdirectory for images, in which it stores its mathematical and graphical displays, including animations, as GIF files.

5.3 Exporting worksheets as LaTeX

LaTeX is a typesetting language much used for scientific and mathematical papers because of the ease in which mathematical formulae can be expressed in it. Exporting your Maple work as LaTeX will allow you to incorporate it into a longer LaTeX document.

1 Return to your Maple window and select **File | Export As** and export a copy of **mymaple1.mws** to your Maple directory in **LATEX** format.

Your worksheet will be exported as **mymaple1.tex** and the plot will be saved as **mymaple1plot3d1.eps**. If you had included more than one 3D plot in your worksheet, they would be numbered **mymaple1plot3d2.eps**, **mymaple1plot3d3.eps**, etc.

2 Move to a Linux terminal window to process and view your LATEX file:

```
latex mymaple1
dvips mymaple1
gv mymaple1.ps
```

5.4 Exporting worksheets in Rich Text format

If you wish to include all or part of a worksheet in a document written on a word processor such as Microsoft Word, you may find Rich Text format (RTF) a convenient way to transfer your Maple work. To export your worksheet in RTF, use the **File | Export as** menu item and select file type **Rich Text Format.**

6 Examples to try

Now that you know a little about using Maple, try these examples. The examples are designed not only to help you explore Maple, but also to encourage you to use the on-line help for information about how to use Maple for specific tasks. (Answers are at the end of this document.)

Example 1

Use the **solve** command to solve the equation $4x^2 + 129x - 703 = 0$. Substitute the solutions back into the equation to check that they are valid.

Example 2

Maple is good for problems that involve very large numbers of digits. Using the Expressions palette and the Numerical Calculations section of the on-line Maple Tour to guide you, express the following sum as a fraction, and then as a floating point approximation to 30 significant figures. The Maple Tour is listed under the **Help** menu.

$$\sum_{j=1}^{10} \frac{j!(j-1)}{(j+1)^j}$$

Example 3

Many well-known functions are already defined in Maple. Plot the 1st-order BesselY function (the Bessel function of the second kind) of x^3 over the range $1 \le x \le 5$. Improve the smoothness of the plot by increasing the number of plotted points to 100.

Example 4

In example 3, add a title to the worksheet. Create a section, add a section heading that describes the problem, and put the plot command into this section. Then hide the section contents.

7 Maple spreadsheets

Maple's symbolic and numerical functionality can also be used in spreadsheets. The following example involves a simple spreadsheet which uses the function

$$\frac{1}{a\sqrt{2\pi}} e^{(-u^2/2a^2)}$$

1. First, open a new Maple document.
2. Click on **Insert | Spreadsheet**.
3. Accept the default name for the spreadsheet.

An empty spreadsheet will appear in your worksheet and the Spreadsheet menu will become active.

7.1 Entering values into cells

In column A of your spreadsheet, enter some values for **a**:

1. In cell A1, type **a**. Press the **Return** key to complete the entry and display the value in the cell. If you wish to edit the contents of a cell, double-click on the cell.
2. In cell A2, type **1** (the number one, not letter 'l') and press the **Return** key.

Now fill cells A3 to A6:

3. Select cells A2 to A6 so that they are highlighted.
4. Right-click on the highlighted cells and select **Fill | Detailed…** from the pop-up menu.
5. Set a step size of **1** (one) and press **OK** to fill downwards.

Cells A3 to A6 should be filled with the numbers 2 to 5.

7.2 Referring to the contents of other cells

Now fill column B with the values **u** to **u-5**. Start by filling cell B1:

1. In cell B1, type ***u*** and press the **Return** key.

You will generate the rest of this column from the values in column A. Each cell will contain the value in cell B1 minus the value in the cell to the left. The cell values can be generated from a single formula that uses cell references.

Summary of cell references:
~B1 refers to relative cell B1.
~$B1 refers to absolute column B, relative row 1.
~B$1 refers to relative column B, absolute row 1
~B1 refers to absolute cell B1.

Cell B2 will contain the value in cell B1 minus the value in the cell A2.

2 In cell B2, type:

~B$1 – ~A2

and press Return.

~B$1 is an absolute reference to row 1 of this column (B) and ~A2 is a reference to the cell to the left of B2.

3 Select cells B2 – B6.

4 Fill these cells as you did for cells in column A. There is no need to set a step size — the cell references will change automatically.

Column B should now contain the values *u* to *u – 5*. Double-click on cell B4 and look at the panel in the context bar. The contents of B4 are ~B$1 – ~A4. The reference to B1 is unchanged, while the reference to column A has been changed so that it is appropriate for row 4.

You are now ready to enter the formula for column C.

5 In cell C1, type:

1/~A1/sqrt(2*Pi)*exp(-~B1^2/(2*~A1^2))

and press Return.

Check that the expression displayed in C1 now matches the one given at the beginning of this section. (Maple automatically converts 1/sqrt(2) to sqrt(2)/2.)

6 Fill the rest of column C.

Your spreadsheet should now be similar to this:

	A	B	C	D
1	a	u	$\frac{1}{2}\frac{\sqrt{2}\,\mathbf{e}^{\left(-\frac{1}{2}\frac{u^2}{a^2}\right)}}{a\sqrt{\pi}}$	
2	1	$u-1$	$\frac{1}{2}\frac{\sqrt{2}\,\mathbf{e}^{\left(-\frac{1}{2}(u-1)^2\right)}}{\sqrt{\pi}}$	
3	2	$u-2$	$\frac{1}{4}\frac{\sqrt{2}\,\mathbf{e}^{\left(-\frac{1}{8}(u-2)^2\right)}}{\sqrt{\pi}}$	
4	3	$u-3$	$\frac{1}{6}\frac{\sqrt{2}\,\mathbf{e}^{\left(-\frac{1}{18}(u-3)^2\right)}}{\sqrt{\pi}}$	
5	4	$u-4$	$\frac{1}{8}\frac{\sqrt{2}\,\mathbf{e}^{\left(-\frac{1}{32}(u-4)^2\right)}}{\sqrt{\pi}}$	
6	5	$u-5$	$\frac{1}{10}\frac{\sqrt{2}\,\mathbf{e}^{\left(-\frac{1}{50}(u-5)^2\right)}}{\sqrt{\pi}}$	

Note: if you go back and edit a spreadsheet cell, other cells that reference this cell will become 'stale'. Stale cells are hatched with grey lines. Highlight the stale cells and go to **Spreadsheet | Evaluate Selection** to re-evaluate these cells.

Example 5

Create a function in column D which integrates the function in column C over the range *u=0..5* and evaluates the integral.

8 Programming in Maple

Most of the Maple language that this Guide has shown so far has been suitable for both Documents and Worksheets. While Maple documents are useful for short calculations and mathematical documents, many Maple users prefer worksheets, which are better suited to work that involves solving longer, more complex problems. This section contains information about more of Maple's mathematical language and about using worksheets to write Maple programs.

8.1 Elements of programming

The following Maple program illustrates several concepts that will be useful to you in designing Maple programs.

1 Open a new worksheet using **File | New | Worksheet** and type in these lines, using Shift-Return between lines. The comments that follow it will help you understand any points on which you are uncertain.

```
#Program 2.
#lines that begin with a # count as comment lines
#and are not executed

restart;

y:=b->int(sin(3*x)+cos(2*x),x=0..b);

arr1:=Array(1..10):
arr2:=Array(1..10):

for ii from 1 to 10 do
   if (ii<6) then
     arr1[ii]:=y(ii)
   else
     arr1[ii]:=ii
   fi
od;

print(arr1);

for jj from 1 to 10 do arr2[jj]:= evalf(arr1[jj]) od;

file1:=fopen("Maple/mapledata",WRITE);
fprintf(file,"%10.7f",arr2);
fclose(file1);
```

Understanding the program line by line:

restart;

reset Maple, removing previous assignments to variables. Including the restart command at the beginning of your worksheets means that Maple will ignore assignments made earlier in your current or other worksheets.

y:=b->int(sin(3*x)+cos(2*x),x=0..b);

Define y to be a function of b. Although you can use the notation *y(b):=* , the 'arrow' notation allows you to calculate values such as *y(1.03)* and *y(pqr)* as well as *y(b).*

arr1:=Array(1..10):

Define *arr1* to be an array of length 10. Notice that this line ends in a colon instead of a semi-colon. The colon suppresses the screen output.

for ii from 1 to 10 do ...... od;

Loop 10 times over the enclosed commands, stepping the value of *ii* from *ii=1 to ii=10*. If you attempt to enter a loop using Return between the lines instead of Shift-Return, Maple will display errors.

if(ii<6) then ..... else ..... fi;

Execute the first set of commands if the statement *ii<6* is true, otherwise execute the alternative. As above, Shift-Return should be used between these lines instead of Return.

arr1[ii]:=y(ii)

Calculate *y(ii)* and assign it to the ii'th element of *arr1*. Square brackets are used to refer to elements in the array.

print(arr1);

Print *arr1* on the screen.

evalf(arr1[jj]);

Calculate a floating-point approximation for the jj'th element of *arr1*.

file1:=fopen("Maple/mapledata",WRITE);

Open the file *Maple/*mapledata ready for writing. If the file does not exist, it will be created. The *file descriptor* **file1** will be used for all further references to this file.

fprintf(file1, "%10.7f",arr2);

Write array **arr2** to the file **file1**, writing each element in the format **10.7f**. Maple's format specifiers are based on those in the C programming language; this one specifies a floating point number with a total width of 10 characters, with room for , of which 7 come after the decimal point. Information about format specifiers can be found in any book on C and on the World Wide Web.

fclose(file1);

Close file **file1**.

The mapledata file will now be in your **Maple** directory.

8.2 Data structures

Arrays were introduced in the section above. They are one of a number of data structures, summarised in the table below, that can be used to represent blocks of data. Careful use of these structures can greatly improve some programs.

Type	Example	Features
sequence	z := a,b,c,d;	Simple list.
Set	z := {a,b,c,d};	Order and repetition not preserved.
List	z := [a,b,b,a];	Order and repetition preserved.
Array	z:= Array([[a,b],[c,d]]);	Like a list, but can have more than one dimension; use for numerical computations with **LinearAlgebra** package.
Matrix	z:= ([[a,b],[c,d]]);	Like **Array**, use for numerical computations with **LinearAlgebra**
Vector	z:= ([a,b,c]);	Like **Array**, use for numerical computations with **LinearAlgebra**.
array (old style)	z := ([a,b,b,a]); z := ([[a,b],[c,d]]);	Like list, but can have more than one dimension. Superseded by Array
Table	z := ([(a)=x,(b)=y]);	Like array, but indices need not be integers.

The set of structures **Array**, **Vector** and **Matrix** are designed for more efficient numerical computations with the LinearAlgebra package. These 'rtable'-based constructs are distinct from the older structures **array**, **vector** and **matrix**.

8.3 Maple's package library

Many of Maple's commands are grouped into 'packages' of commands specific to one area of mathematics. For example, there are packages for linear algebra, statistics and tensors. For a list of all the packages and descriptions of their contents, search the online help pages for the index of **packages.**

The program below illustrates use of the **LinearAlgebra** package. The contents of the package are loaded by the **with(LinearAlgebra)** command**.** The package's commands are recognised only after this line.

1 Open a new worksheet and type in this program:

```
restart:

mat:=Matrix([[100, 81, 64], [49, 36, 25], [16, 9, 4]]);

with(LinearAlgebra);
if Determinant(mat) <> 0 then inv:=MatrixInverse(mat) fi;
```

2 Edit the program to check that the matrix multiplied by its inverse gives the identity matrix:

```
test:=MatrixMatrixMultiply(mat,inv);
```

If your work involves scientific constants, you may find the **ScientificConstants** package useful. This gives access to a wide range of physical constants and information about chemical elements. First load the package by typing:

```
with(ScientificConstants);
```

The ratio of an electron's charge to its mass, for example, is then calculated as:

```
evalf(Constant(elementary_charge)/Constant(electron_mass));
```

8.4 Computations with floating point numbers

While Maple often produces exact symbolic or numerical answers, such as 2π or 1/3, some Maple programs are required to produce a numerical approximation using floating point numbers, e.g. 6.283185308 or 0.3333333333. Maple makes a firm distinction between these types. This section discusses floating point numbers (numbers that contain a decimal point), which are also the subject of the chapter on Numerical Programming in the *Maple Advanced Programming Guide* and the Help page on 'numerics'.

Maple has two ways of making floating point calculations. Most calculations use the *software mode*, which is flexible and allows you to define the precision, but software floating point calculations can also be much slower than their *hardware mode* equivalent.

A software floating point calculation will, by default, be evaluated using a precision of 10 digits:

```
evalf(2*Pi);

        6.283185308
```

Since each evaluated step in the calculation has finite precision, rounding errors can accumulate. Do not expect floating point calculations to give exact answers such as 0 or 1: look for a number close to the exact answer instead. E.g:

```
(evalf(2/3*Pi) + evalf(Pi/3)) / evalf(Pi/3);

        3.000000001
```

One extreme example of rounding errors is called 'catastrophic cancellation', which occurs typically when two very similar numbers are subtracted. In the following example, the smaller number *b* is lost during the operation *evalf(a+b)* even though it later becomes important. Programs can usually be designed so that they avoid this problem.

```
a:=12345678901;

b:=1;
```

```
a:=12345678901;

b:=1;
```

```
evalf(a + b) - a;
```

```
0.0
```

If you suspect that the default precision is insufficient for a particular program, increase it by changing the environment variable **Digits**:

Digits:=*N*;

where ***N*** is a number greater than 10. Note that increasing Digits may cause your program to run more slowly.

The precision can also be changed for a single **evalf** command, e.g.:

```
evalf[20](2 * Pi);
```

```
6.2831853071795864770
```

The **evalhf** command is similar to **evalf** except that it uses hardware floating point numbers. The precision of hardware floating point calculations is dependent on your computer and is unaffected by the value of **Digits**. For an approximation to the number of digits used in hardware floating point calculations, use the command:

```
evalhf(Digits);
```

Although hardware-based calculations can be very much faster, especially for calculations involving arrays of numbers, **evalhf** cannot evaluate all Maple expressions. The Help page on **evalhf** and the *Maple Advanced Programming Guide* have more information.

8.5 Exercises

Example 6

The following program is very slow. How could it easily be made much faster?

```
restart:
y:=array(1..100):
eqn:=n->log(n!)-exp(n/evalf(sum(BesselY(3,x),x=1..100)));

for i from 1 to 100 do
  y[i]:=eqn(i)
od;
```

Example 7

Write a Maple program that finds the Fourier transform of the function $f(x) = \exp(-x^2 / c^2)$. You will need to tell Maple to **assume** that *c* is real. Plot the original function and its transform over the range $-5 \le x \le 5$ for the case $c = 1$. Assign the two plots to variables **plot1** and **plot2** and display them on a single plot using the **display** command. Change the plots to animations over $1 \le c \le 2$ and repeat the display, running it forwards and backwards. (Click on the display area to produce the animation controls on the context bar.)

Example 8

Write a program to perform a simple numerical integration (*without* using **int** or **Int**) of the function $\sin(x)e^{-x}$ in 10^6 steps over the range 0 -- 100000. Use the **int** function to check your answer.

8.6 Writing procedures

Procedures are a way of grouping sets of commands into units that can be used several times within a program or in different programs.

The rules governing evaluation and the nature of variables in procedures are not all the same as the rules governing simple Maple programs. If you intend to use procedures extensively, it is strongly recommended that you consult the *Maple Advanced Programming Guide*, which is available on loan from the IT Service Desk and from the University Library.

A simple procedure can contain a single command enclosed by **proc** and **end** statements:

```
ans := proc(x::integer)
  y:=x^2;
end;
```

Typing **ans(2);** would then give the answer **4**.

Local variables, such as *y* in the example above, are internal to the procedure, so x^2 is not assigned to *y* in the main program. Variables that are needed outside the procedure should be defined as **global** instead of the default **local.**

Environment variables can be changed within a procedure but are reset when the procedure is left.

9 Improving Maple programs

9.1 What to do if your answers seem wrong

Sometimes Maple may return an answer that is in a strange format or that seems to be wrong. Incorrect answers are often the result of simple programming errors or a misunderstanding of the way in which a Maple command works. If you suspect that an answer is wrong, following the steps below will help you to locate and solve the error.

- Try substituting the result back to check that it is really wrong. It could be correct but in an unfamiliar form.
- Use the **evalf** and **print** commands to check the values of your variables and constants.

- Check that you are using Maple functions and commands correctly.
- Exit from Maple and see if you can reproduce the problem in a fresh Maple session.
- See if the problem can be recreated with just a few lines or a single command.
- Consider using the Maple debugger, described in Chapter 8 of the *Maple Introductory Programming Guide*, to locate the problem.
- Ask other Maple users for help or e-mail itservicedesk@durham.ac.uk.
- Search online for the archives of the Maple User Group mailing list. There may be relevant messages if other users have encountered similar problems.
- Occasionally, a bug is found in Maple. However, complete the checks listed above before you suspect Maple itself. Most problems do have simpler solutions.

9.2 Avoiding common errors

- Use **:=** instead of **=** in assignment statements so that Maple keeps the assignment throughout your program. Use **=** for equations and within other commands, e.g.

```
y:=x^2+2*x+1;
if (a=1) then b:=2 fi;
subs(r=3,p);
```

- Be careful with capital letters in functions, variables and symbols such as **Pi** and **I.** Some functions come in two different forms, one with an initial capital letter and one just in lower case. In general, functions with an initial capital letter are not automatically evaluated, so **Int(sin(x),x);** gives $\int \sin(x)\, dx$ while **int(sin(x),x);** gives $-\cos(x)$.
- The three types of quotation marks **`**, **’** and **”** all have specific meanings in Maple. ` is used to name symbols, ” denotes strings, and ’ delays evaluation of the enclosed expression. Look at the examples in relevant Help pages to be sure you are using the correct type in a function, and read the Help page on ‘quotes’.
- Names that are used for Maple functions, such as **sin, cos, Int** and **write,** are ‘protected’ and cannot be used for other purposes, such as naming variables.
- If you use a function name without loading its parent package, Maple will echo the function name but may not have recognised it as a function or have calculated a result.
- The same function name may appear in more than one package. Be sure that you know which one you are using and, if this is a particular problem, call the function specifically as *package[function]*.
- When you execute part of a worksheet or jump between worksheets, previously calculated terms are not necessarily reset. Avoid this problem by beginning your worksheets with the **restart;** command and executing the entire worksheet at once using **Edit | Execute | Worksheet.**
- When you write to a file, check whether your data are symbolic or numerical. Commands such as **writedata** or **printf** cannot handle symbolic data easily. **Close** the file when you finish writing to it or it

may appear to be empty until you close the worksheet or exit from Maple.

9.3 Tips on efficient programming

- Maple does not assume that variables are real and it does not discard imaginary solutions unless you request this. See the Help pages on **assume, assuming** and **Re** for more information.
- If a calculation involves a series with many small terms that can be neglected, consider truncating the series. The Help page on **series** has advice on removing terms.
- If you can simplify or evaluate part of an expression at an early stage, it may reduce the work to be done in later stages of a calculation.
- Consider whether your program will run faster if you replace **evalf** with **evalhf**, which will use your computer's hardware floating point arithmetic instead of Maple's software system. Hardware floating point calculations *can* be much faster but you cannot change the precision.
- A visually elegant program is not always the most efficient.
- Read the Help page on 'efficiency'.

10 Reading command files and exporting commands

10.1 Reading commands from text files

It is not always convenient to create Maple programs entirely within the worksheet environment. Similarly, you may wish to save a group of commands for use in several worksheets or in a form understood by command-line Maple. The **read** and **save** commands provide this facility.

```
A:=[[1,1],[2,2],[3,3],[4,4],[5,5]];
save A,"Maple/mplcoms";
read("Maple/mplcoms");
```

1. Type the lines above into your worksheet.

Files saved in this way are simple text files containing Maple commands (but see the Help page on **save** for information on files whose names end in ***.m**).

2. Now move to a Linux terminal window and edit the file **mplcoms**. It will be in your **Maple** directory.
3. In a Linux editor, add a line to **mplcoms** stating that **twoA:=2*A;** and save the file.
4. Re-execute the **read** line in your worksheet.

If you prefer, you can compose whole Maple programs in a text editor rather than in a worksheet and then run them using **read**.

Example 9

The Maple program below solves a system of differential equations. Use either a text editor or the Maple worksheet environment to enter the program, and then execute it.

```
restart;

# curly brackets like {} are used to group items together in sets
sys := { (D@@2)(y)(t)=-y(t), (D@@2)(x)(t)=-x(t)+2*D(y)(t)) };
ics := { D(x)(0)=0, D(y)(0)=0, x(0)=1, y(0)=1 };

# 'union' is used to give the union of the sets of equations and
# initial conditions.
ans := dsolve(sys union ics, {x(t),y(t)} );

# substituting the solution from dsolve into x(t) and y(t) gives
# the two solutions separately.
xans := subs(ans,x(t));
yans := subs(ans,y(t));
```

Plot the results for *x(t)* and *y(t)* together on a single graph over the range $-2\pi \le t \le 2\pi$

Add a loop which evaluates numerical values for *xans* at 32 values of *t* in the range $0 \le t \le 2\pi$.

10.2 Creating Fortran and C expressions

Maple is not always an efficient tool for lengthy numerical calculations. If you suspect that this is a problem for you, use Maple for symbolic calculations and then perform the numerical part of your calculations elsewhere by translating and exporting your output as Fortran or C. Note that some features, such as special functions, may not be translatable.

The **Fortran** and **C** conversion functions are part of the **CodeGeneration** package. Conversions to other languages are also available.

```
y := int(x*sin(2*x) + x*cos(3*x),x);
with(CodeGeneration);
Fortran(y);
C(y);
```

Note: Maple also allows you to make calls to external functions. If you wish to make use of this facility, please consult Chapter 7 of the *Maple Advanced Programming Guide*.

Example 10

In the example above, write an optimised **C** expression for *y* and send the expression directly to a file in your Maple directory.

11 Working with data files

11.1 Writing data files

Numerical results can easily be saved to a file with the **writedata** command:

```
A:=[[1,1],[2,2],[3,3],[4,4],[5,5]];
writedata(”Maple/simple.out”,A);
close(”Maple/simple.out”);
```

1 Type the lines above into a new worksheet.

They will generate a small text file, **simple.out,** in your **Maple** directory.

2 Move to a Linux terminal window and view the **simple.out** file.

In it, the five pairs of values from *A* are stored on separate lines:

```
1        1
2        2
3        3
4        4
5        5
```

3 In your Maple worksheet, add an extra pair of values, [6,6], to *A* and re-execute the worksheet.

4 View the new **simple.out** file from your Linux window.

Notice that **writedata** has overwritten the previous **simple.out** file; you cannot add to a file using **writedata.**

writedata is very easy to use and allows you to write tabulated data or lists of values to a file using a single command. The online Help page contains information on extending **writedata** to cover symbolic and complex data. However, if you want to add to an existing file, write a file in binary format or save data in a particular layout, you will need to use Maple's input and output (I/O) routines, which are similar to the I/O routines in the C programming language.

The following lines show a simple example in which the contents of the array *A* from above are written in a specific format to the file **formatted.out** in your **Maple** directory. Add the lines to the program above.

```
for loop from 1 to 6 do
  fprintf(”Maple/formatted.out”,”%d %f Text\n”,A[loop,1],A[loop,2]);
od;
close(”Maple/formatted.out”);
```

The **fprintf** command contains the file's name, the format of the output, and the items to print.

As before, the array *A* is displayed as two columns. The first column contains integers, as specified by the **%d** format. (Note: the **%** symbol has a separate meaning in format specifiers; it is not a ditto.) The second column is now in floating point format, **%f**. A third column has the text

‘Text’. At the end of the format specification, the **\n** is an instruction to move to a new line. **close** closes the file.

5 In a Linux terminal window, view the file **formatted.out**.

```
1 1.000000 Text
2 2.000000 Text
3 3.000000 Text
4 4.000000 Text
5 5.000000 Text
6 6.000000 Text
```

11.2 Reading data files

The easiest data files to read with Maple contain tabular, numerical data, like the data in file **simple.out** written above.

1 In your worksheet, type

```
B:=readdata(”Maple/simple.out”

);
```

By default, **readdata** reads only the first column of the file, so this is all that array *B* contains. *B* is also in floating point format.

2 Edit the command so that **readdata** reads both columns into *B* and stores them as integers*:*

```
B:=readdata(”Maple/simple.out”,[integer,integer]);
```

Data should normally be read from a formatted file using the format that was used to write them. The example below illustrates one way of reading a file of unknown length. The first row of the file is read into receiving array **B2** and then subsequent sets of elements are added as the file is read. In each iteration of the loop, the **op** command extracts the elements from the existing version of **B2** so that they form the basis of the new, expanded version.

3 Add the lines below to your worksheet. Use the **Linefeed** key or **Shift-Return** at the end of each line to help you construct several lines of the program without executing each line as it is entered.

```
# begin by reading the first row of the file
B2:=fscanf(”Maple/formatted.out”,”%d %f %4c”);
#
# then loop over the remaining rows until the end
# of the file is reached, adding the contents of
# each row to B2.
do
indata:=fscanf(”Maple/formatted.out”,”%d %f %4c”);
if indata = 0 then break fi;
B2:=[op(B2),indata];
od;
```

Example 11

Edit the program so that the string '*Text*' is discarded instead of being entered into the third column of *B2*. You may need to read the Help pages on fscanf to find out more about format specifiers. The answer is at the end of this document.

Example 12

Write a procedure that evaluates x^2 if $x \geq 0$ and $-x^2$ if $x < 0$, where x is a floating point number. Change the **Digits** environment variable within the procedure so that the answer is given to 15 digits.

Example 13

Write a procedure that solves a system of two differential equations for two functions of a variable *t*, given the equations and initial conditions as arguments. Use the equations in example 10 to test your procedure.

12 Customising your Maple sessions

Many of Maple's generic settings can be set from the menu bar, using **Tools |Options.** The information will be kept in a subdirectory **.maple** in your home directory.

You can also create a file in your home directory called **.mapleinit**. This file will be read at the start of your Maple sessions. **For** example, you might always want to use 20 digits for all floating point calculations, pre-load the 'plots' package and do all your Maple work in your **Maple** directory. Then your **.mapleinit** file could contain these lines:

```
Digits:=20:
`plots/init`:=proc() NULL end:
with(plots):
currentdir(cat(getenv(”HOME”),”/Maple”)):
```

The second line in this example is required if you load the plots library in this way; it avoids an error. The back quotes (`) are used as a way to define **plots/init** as the name of a symbol even though it includes the **/** symbol that is usually forbidden in variable names.

The call to **currentdir** sets your initial working directory to be the **Maple** directory under your home directory. The path to your home directory is obtained from the **getenv** command and this path is con**cat**enated with the suffix **/Maple**. Note that this call requires you first to go to the menu item **Tools|Options** and enable system calls in the Security tab.

1. Use a text editor (such as **pico, emacs** or **vi**) to create a **.mapleinit** file in your home directory, including the **currentdir** commands as above.
2. Run **xmaple**.

All of your work will now automatically be in your **Maple** directory.

Once it has been created, the .**mapleinit** file is read whenever you execute **restart** and at the beginning of every new xmaple or maple session you run.

13 Other Maple modes

13.1 Command line Maple

Throughout this introduction you have used the graphical version of Maple, xmaple. If you use Maple for intensive computations or if your computer does not have much memory, you may sometimes find it useful to use the command line version. To run this version, follow the steps in section **Error! Reference source not found.** to create a xterm window in which the Maple commands are understood and then, in this window, type

maple

14 Running Maple on the Networked PC Service

Maple is located on the Start menu at **Start | Programs | Programming Languages | Maple 11**

From here you can start **Maple 11.**

Nearly all of the Maple language included in this Guide can be used on the Linux and Windows versions of Maple. If you need to specify a filename in Maple, e.g. in the plotsetup command, use *backslashes* in the path name, such as plotoutput=`j:\Maple\test.ps`. To create a customised setup as described in section 12, call the customisation file **maple.ini** and place it in your **J:** drive.

14.1 Links to other PC software

Maple is available as an add-in for Excel on the networked PC service.

There is also a Maple-Matlab link, which allows you to access some Matlab commands from within Maple. To initiate the link, type:

with(Matlab);

There is a full list of the available commands on the Help page for matlab.

15 Keyboard shortcuts and special characters

As you write your Maple programs, you may find the following key sequences useful:

Ctrl/E	Move cursor to end of line (use the **End** key if in Windows)
Ctrl/A	Move cursor to beginning of line (**Home** key if in Windows)
Ctrl/K	Create new execution group before the current line

linefeed or **Shift/Return**

Create a new line without executing the current line or creating a new execution group.

Maple reserves certain keyboard characters for particular actions. They are:

?   A question mark before a word summons the help page on that word.

****   The \ character has several meanings in Maple. If it is placed at the end of a line, the command is continued onto the next line, e.g.

```
y := 1 + x + x^2/2 + x^3/6 + \
x^4/24;
```

In the middle of a line it can be a control character if it precedes certain other characters. These are listed on the **backslash** help page.

$ The dollar symbol is used in Maple to form sequences. This example selects the first three elements of the array Z:

```
Z := [1,2,3,4,5];
Z[i] $i=1..3;
```

%   The % symbol is usually acts as a ditto. It refers to the last expression executed. % is also an escape character in format specifiers.

16 Where to find out more about Maple

Three good books on Maple, the *Maple User Manual,* the *Maple Introductory Programming Guide* and the *Maple Advanced Programming Guide*, are available for loan from the IT Service Desk.

The on-line Maple worksheets, found under the Introductory Help window menu, contain examples of many areas of mathematics, including some of the Maple packages.

There is a mailing list for people in Durham with a special interest in Maple. To subscribe to the list, send an e-mail to **majordomo@durham.ac.uk** with the message **subscribe sigmaple**. News of upgrades and new features will be sent to this list. You can also write to the **sigmaple** list if you have problems with Maple — or if you can provide an answer to someone else's problem.

Links to the Maple home page and other Maple web pages can be found at

http://www.dur.ac.uk/ITS/software/maple/

This page also contains local information and examples of Maple output.

17 Answers!

Example 1

```
eqn:=4*x^2 + 129*x - 703;
solve(eqn,x);
subs(x=-37,eqn);
subs(x=19/4,eqn);
```

Example 2

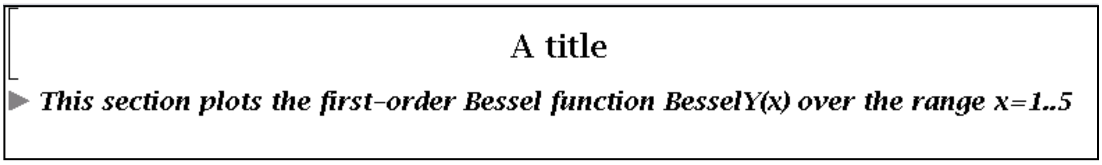
```
sum( j! * (j-1) / (j+1)^j, j=1..10);
evalf(%,30);
```

Example 3

```
plot(BesselY(1,x^3), x=1..5, numpoints=100);
```

Example 4

Your finished worksheet should show just the title and the section heading, as in this example.



Example 5

Enter the expression **evalf(int(~C1,u = 0 .. 5))** into cell C1. Fill cells C2 -- C6 from this.

Example 6

The most time-consuming part of the program is the evalf(sum(BesselY...))) command. Calculating this just once, outside the loop, makes the program much more efficient.

```
restart:
y2:=array(1..100):
inert:=evalf(sum(BesselY(3,x),x=1..100));
eqn:=n->log(n!)-exp(n/inert);

for i from 1 to 100 do
  y[i]:=evalf(eqn(i))
od;
```

Example 7

For the Fourier transform program you will need to use the Integral Transform (**inttrans**) and Plots libraries. After the **assume** command, *c* is displayed in output as *c*~ to indicate that assumptions have been made about it.

```
restart;
with(inttrans):
with(plots):
a:=exp(-x^2/c^2);
assume(c,real);
b:=fourier(a,x,w);

plot(subs(c=1,a),x=-5..5);
plot(subs(c=1,b),w=-5..5);

#end the following two lines with colons to avoid
#lengthy, unwanted screen output.
plot1:=animate(a,x=-5..5,c=1..2):
plot2:=animate(b,w=-5..5,c=1..2):

display(plot1,plot2);
```

Example 8

There are many ways of writing this program. Below is one:

```
# define range and number of steps
nsteps:=1000000;
nend:=100000;
intvl:=nend/nsteps;

# Initialise integral
tot:=0:

# Loop to calculate integral.  Suppress screen output.
for i from 0 to nend do
  step:=(i+0.5)*intvl:
  this:=evalhf(sin(step)*exp(-step))*intvl:
  tot:=tot+this:
od:

# Display answer
tot;

# Compare with Int
evalf(Int(sin(x)*exp(-x),x=0..10000));
```

Example 9

You could add the option **type=numeric** to the **dsolve** command and then use the **odeplot** command to plot the solutions:

```
with(plots);
odeplot(ans,[t,x(t)],-2*Pi..2*Pi);
```

This command is simple to use but is very slow to execute. Since the two separate solutions have already been extracted from **ans,** it is much faster in this example to use **plot**:

```
# curly brackets can also be used to plot several items on a single graph.
plot( {xans,yans},t = -2*Pi..2*Pi);
```

To create the array of numerical values, a loop is required. The screen output from the loop is suppressed by the colon after **od** even though the lines within the loop end in semicolons.

```
vals := array(1..32);

for i from 1 to 32 do
  ti := i*Pi/16;
  vals[i] := evalf(subs(t=ti,xans));
od:
```

Example 10

```
y := int(x*sin(2*x) + x*cos(3*x),x);
with(codegen);
readlib(C);
C(y, optimized, filename=”Maple/maplecoms.c”);
```

Example 11

Change the **fscanf** lines to read in the nine-character string but to ignore it. This is achieved by changing **%4c** to **%*4c**. The whole program will now read like this:

```
A := [[1,1],[2,2],[3,3],[4,4],[5,5],[6,6]];
writedata(”Maple/simple.out”,A);
close(”Maple/simple.out”);

for loop from 1 to 6 do \
  fprintf(”Maple/formatted.out”,”%d %f This is A\n”,A[loop,1],A[loop,2]) \
od;
fclose(”Maple/formatted.out”);

B := readdata(”Maple/simple.out”,[integer,integer]);

B2 := fscanf(”Maple/formatted.out”,”%d %f %*4c” );
do
indata := fscanf(”Maple/formatted.out”,”%d %f %*4c”);
if indata = 0 then break fi;
B2 := [op(B2),indata];
od;
```

Example 12

```
ans := proc(x::float)
  Digits := 15;
  if (x>=0) then
    y := x^2
  else
    y := -x^2
  fi;
y;
end;
```

Example 13

```
ans := proc();
dsolve(sys union ics, {x(t), y(t)} );
end;

sys := { (D@@2)(y)(t)=-y(t), (D@@2)(x)(t)=-x(t)) };
ics := { D(x)(Pi)=0, D(y)(0)=0, x(Pi)=1, y(0)=1 };

thisans := ans(sys,ics);
```