A Maple Handbook

All you need to know
— and more

D H Mackay
J J O'Connor
MT1008 2009

You can find electronic versions of each of the lectures in the Maple folder on the L-drive on each of the computers in the microlab.

You will not be able to change them there, but you can copy them into your own filespace and then you can play with them and modify the code to see what happens. If you want to remove all the output you can use the *Remove output* command on the **Edit** menu.

The same files will be available on the MMS system:

`https://mms.st-andrews.ac.uk/mms/maths.html`

in the section: **MT1008: Student information area**

and you can download them from there.

Seven pages of summaries are at the end of this booklet.

Contents

Using Maple as a calculator	2	Lists and sets	30
Help	4	Procedures	33
Polynomial expressions	5	More procedures	38
Trigonometric expressions	6	Reversing	39
Assigning	7	Pythagorean triples	43
Substituting	9	Some more equations	44
Differentiating	11	A mysterious sequence	44
Formatting worksheets	12	Continued fractions	46
Defining functions	13	The sum of two squares	47
Integrating	15	Looking for prime numbers	48
Plotting	19	Linear algebra package	51
Solving	22	Sudoku	57
Looping	27		
If clauses	30	Summaries 1 — 7	60

Using Maple as a calculator

As a calculator, Maple works like any electronic calculator, except that you have of put **;** after each calculation and press the **Return** or **Enter** key. To move the cursor down to the next line without doing the calculation, use **Shift-Return**.
You can press the key when the cursor is anywhere in the red bit of the "group" and Maple will calculate for you.
You can go back and calculate with an earlier bit of code by clicking on the red bit and pressing **Return**.

```
22/7-355/113;
sqrt(45);
```

$$\frac{1}{791}$$

$$3\sqrt{5}$$

To get a decimal answer, use the **evalf** (evaluate as a floating point number) function
The % stands for the last result Maple calculated. Note that this might not be the previous result on the work-sheet if you went back and recalculated an earlier entry.

```
evalf(%);
```

$$6.708203931$$

%% stands for the last but one result. This time you can get the answer to 20 significant figures

```
evalf(%%,20);
```

$$6.7082039324993690892$$

Using a decimal point in your input tells Maple that you want the answer as a decimal.

```
sqrt(2.0);
```

$$1.414213562$$

Maple knows about Pi, which it calls **Pi** (the capital letter is important) and will give it to very great accuracy.
Over the centuries mathematicians spent a lot of time calculating many digits of Pi. The methods developed included a series for **arctan** discovered by *James Gregory*, the first Regius Professor of mathematics at St Andrews.
The English mathematician *William Shanks* published 707 places of Pi in 1873 and it was not discovered until 1943 that the last 179 of these were wrong.
The expansion of Pi is now known to many billions of places.

```
evalf(Pi,1000);
```

3.141592653589793238462643383279502884197169399375105820974944592307816406286208\
99862803482534211706798214808651328230664709384460955058223172535940812848111\
74502841027019385211055596446229489549303819644288109756659334461284756482337\
86783165271201909145648566923460348610454326648213393607260249141273724587006\
60631558817488152092096282925409171536436789259036001133053054882046652138414\
69519415116094330572703657595919530921861173819326117931051185480744623799627\
49567351885752724891227938183011949129833673362440656643086021394946395224737\
19070217986094370277053921717629317675238467481846766940513200056812714526356\
08277857713427577896091736371787214684409012249534301465495853710507922796892\

58923542019956112129021960864034418159813629774771309960518707211349999998372\
97804995105973173281609631859502445945534690830264252230825334468503526193118\
81710100031378387528865875332083814206171776691473035982534904287554687311595\
62863882353787593751957781857780532171226806613001927876611195909216420199

22/7 is a well-known approximation for Pi.
This was known to the Greek mathematician *Archimedes* about 250BC (and indeed earlier).
A better, but less well-known approximation is 355/113.
This was discovered by the Chinese mathematician *Ch'ung Chi Tsu* in about 500AD.
Maple will calculate the difference between these two approximations and Pi.

```
evalf(22/7);
evalf(355/113);
evalf(22/7-Pi);
evalf(355/113-Pi);
```

$$3.142857143$$

$$3.141592920$$

$$0.001264489$$

$$2.66\ 10^{-7}$$

Maple knows about all the functions you have on your calculator: **sqrt**, **sin**, **cos**, etc as well as **exp**, **log** = **ln** (log to base *e*), **log10, log[b]** (log to *any* base) and lots more besides.
It uses <u>lower case</u> letters for them.

To use them, put **( )** around what you evaluate. Maple works in radians, not degrees. If you want the answer as a decimal, you will have to ask for it.

```
sin(3);
sin(Pi/2);
sin(60*Pi/180);    the sine of 60°
evalf(%);
sqrt(2);
2^(1/2);
evalf(%,50);
```

$$\sin(3)$$

$$1$$

$$\frac{1}{2}\sqrt{3}$$

$$0.8660254040$$

$$\sqrt{2}$$

$$\sqrt{2}$$

$$1.4142135623730950488016887242096980785696718753769$$

Maple knows about some other functions your calculator (probably) can't handle. For example, **ifactor** (for *integer factorise*) will write an integer as a product of prime numbers.

```
ifactor(123456789);
```

$$(3)^2\ (3803)\ (3607)$$

The function **factorial** will calculate the product 1 × 2 × 3 × ... × *n* usually written *n*!
Maple recognises the **!** notation too.

```
factorial(5);
factorial(100);
ifactor(100!);
```

$$120$$

933262154439441526816992388562667004907159682643816214685929638952175999932299156\
6089414639761565182862536979208272237582511852109168640000000000000000000000000\
0

$$(2)^{97}\,(3)^{48}\,(5)^{24}\,(7)^{16}\,(11)^{9}\,(13)^{7}\,(17)^{5}\,(19)^{5}\,(23)^{4}\,(29)^{3}\,(31)^{3}\,(37)^{2}\,(41)^{2}\,(43)^{2}\,(47)^{2}\,(53)\,(59)\,(61)\,(67)\,(71)\,(73)\,(79)\,(83)\,(89)\,(97)$$

You can even apply **ifactor** to a fraction:

```
ifactor(123456/234567);
```

$$\frac{(2)^6\,(643)}{(3)\,(67)\,(389)}$$

Help

To see the help files on a Maple command, type the command and highlight it. Then go to the **Help** menu and you will see an entry for the command. Alternatively, type ? and then the command. (You don't even need a semi-colon!) .

```
?print
```

You can also use **help(command);** (and you do need the semi-colon!)

```
help(sin);
```

At the bottom of a help file, you will find some examples of how to use the command. (This is the most useful bit!) You can copy and paste these lines into your worksheet and look at what happens. Then you can change them to do what *you* want.

Each help file has a list of links to related topics at the bottom which may let you hunt down exactly what you want.

The **Help** menu also has a "Search" facility which will point you in the direction of any help files where the word or phrase you enter is mentioned. This tends to produce too much output to be very useful!

Polynomial expressions

One of the most important things Maple can do is to calculate with expressions as well as numbers.
Use the **expand** function to "multiply out".

```
(x+y)^5;
expand(%);
```

$$(x \quad y)^5$$

$$x^5 \quad 5\,x^4\,y \quad 10\,x^3\,y^2 \quad 10\,x^2\,y^3 \quad 5\,x\,y^4 \quad y^5$$

```
expand((sqrt(2*x)+sqrt(x))^6);
```

$$99\,x^3 \quad 70\,\sqrt{2}\,x^3$$

Maple will (sometimes) suceed in manipulating an expression to make it "simpler".
Use the function **simplify**.

```
(x^2-y^2)/(x-y);
simplify(%);
```

$$\frac{x^2 \quad y^2}{x \quad y}$$

$$x \quad y$$

Maple will factorise expressions as well — if it can! Use the **factor** function.

```
(x-y)^3*(x+y)^5;
expand(%);
factor(%);
factor((x-y)^3*(x+y)^5+1);
```

This last is too difficult!

$$x^8 \quad 2\,x^7\,y \quad 2\,x^6\,y^2 \quad 6\,x^5\,y^3 \quad 6\,x^3\,y^5 \quad 2\,x^2\,y^6 \quad 2\,x\,y^7 \quad y^8$$

$$(x \quad y)^3\,(x \quad y)^5$$

$$x^8 \quad 2\,x^7\,y \quad 2\,x^6\,y^2 \quad 6\,x^5\,y^3 \quad 6\,x^3\,y^5 \quad 2\,x^2\,y^6 \quad 2\,x\,y^7 \quad y^8 \quad 1$$

Maple will also handle ratios of polynomials in this way.

```
expand(((x-y)^2+(x+y)^2)/(x^3-y^3));
simplify(%);
factor(%);
```

$$\frac{2\,x^2}{x^3 \quad y^3} \quad \frac{2\,y^2}{x^3 \quad y^3}$$

$$\frac{2\,(x^2 \quad y^2)}{x^3 \quad y^3}$$

$$\frac{2\,(x^2 \quad y^2)}{(x \quad y)\,(x^2 \quad x\,y \quad y^2)}$$

Maple can simplify polynomials in some other ways. In particular, you can ask it to collect together the terms in (say) x^n using the **collect** function.

```
(x-2*y)^4+(3*x+y)^3;
expand(%);
collect(%,x);
collect(%,y);
```

$$(x \quad 2\,y)^4 \quad (3\,x \quad y)^3$$

$$x^4 \quad 8\,x^3\,y \quad 24\,x^2\,y^2 \quad 32\,x\,y^3 \quad 16\,y^4 \quad 27\,x^3 \quad 27\,x^2\,y \quad 9\,x\,y^2 \quad y^3$$

$$x^4 \quad (\quad 8\,y \quad 27)\,x^3 \quad (24\,y^2 \quad 27\,y)\,x^2 \quad (\quad 32\,y^3 \quad 9\,y^2)\,x \quad 16\,y^4 \quad y^3$$

$$16\,y^4 \quad (\quad 32\,x \quad 1)\,y^3 \quad (24\,x^2 \quad 9\,x)\,y^2 \quad (\quad 8\,x^3 \quad 27\,x^2)\,y \quad x^4 \quad 27\,x^3$$

Trigonometric expressions

Maple will handle many trigonometric identities using the **expand** function. It won't factor back again (though there is a **combine** function which you can investigate).

```
sin(x+y);
expand(%);
factor(%);
```

$$\sin(x \quad y)$$

$$\sin(x)\cos(y) \quad \cos(x)\sin(y)$$

$$\sin(x)\cos(y) \quad \cos(x)\sin(y)$$

You can use Maple to expand cos(n x) for different values of the integer n and get polynomials in cos (x).
These polynomials were first investigated by the Russian mathematician *Pafnuty Chebyshev* (1821 to 1894). They are very important in Numerical Analysis.

```
cos(5*x);
expand(%);
cos(12*x);
expand(%);
```

$$\cos(5\,x)$$

$$16\cos(x)^5 \quad 20\cos(x)^3 \quad 5\cos(x)$$

$$\cos(12\,x)$$

$$2048\cos(x)^{12} \quad 6144\cos(x)^{10} \quad 6912\cos(x)^8 \quad 3584\cos(x)^6 \quad 840\cos(x)^4 \quad 72\cos(x)^2$$
$$1$$

Maple will (sometimes) simplify trigonometric expressions.

```
sin(x)^2+cos(x)^2;
simplify(%);
```

$$\sin(x)^2 \quad \cos(x)^2$$

$$1$$

Sometimes the answer isn't what you might expect.


```
simplify(1/(1+tan(x)^2));
```

$$\frac{1}{1 \quad \tan(x)^2}$$

Though you can help it!

```
simplify(1/(1+tan(x)^2)-cos(x)^2);
```

$$0$$

Assigning

Maple will store things (numbers, expressions, functions, ...) in "containers" or "variables". Think of these as labelled boxes. This process is called *assignment.*

```
p:=15;
q:=75;
p/q;
```

$$p := 15$$

$$q := 75$$

$$\frac{1}{5}$$

One can also store expressions in these boxes.
One can then apply any Maple function to the contents of the box.

```
quad:=(x+2*y+3*z)^2;
expand(quad);
collect(quad^2,z);
```

$$quad := (x \quad 2\,y \quad 3\,z)^2$$

$$x^2 \quad 4\,x\,y \quad 6\,x\,z \quad 4\,y^2 \quad 12\,y\,z \quad 9\,z^2$$

$$81\,z^4 \quad (108\,x \quad 216\,y)\,z^3 \quad (18\,(x \quad 2\,y)^2 \quad (6\,x \quad 12\,y)^2)\,z^2 \quad 2\,(x \quad 2\,y)^2\,(6\,x \quad 12\,y)\,z$$
$$(x \quad 2\,y)^4$$

One has to be a bit careful, however.

```
c:=a+b;
a:=1;b:=3;
c;
```

$$c := a \quad b$$

$$a := 1$$

$$b := 3$$

$$4$$

If we now change either a or b, Maple remembers that c contains a+b and will change c too.

```
a:=5;
c;
```

$$a := 5$$

$$8$$

However, if we had already assigned numbers before we put them into the box, Maple will just put in the number !

```
x:=1;y:=3;
z:=x+y;
```

```
z;
```

$$x := 1$$
$$y := 3$$
$$z := 4$$
$$4$$

and this time altering one of the numbers will not alter anything else.

```
x:=5;
z;
```

$$x := 5$$
$$4$$

To "empty" one of our boxes or variables we unassign it, using:

```
a:='a';
a;
c;
```

$$a := a$$
$$a$$
$$a \quad 3$$

To unassign all the variables, use **restart;**

```
a;b;c;
restart;
a;b;c;
```

$$a$$
$$3$$
$$a \quad 3$$
$$a$$
$$b$$
$$c$$

One can use this process to evaluate an expression.

```
f:=x^2+1;
x:=1.5;f;
x:=2.5;f;
x:=3.5;f;
```

$$f := x^2 \quad 1$$
$$x := 1.5$$
$$3.25$$
$$x := 2.5$$
$$7.25$$
$$x := 3.5$$
$$13.25$$

Substituting

Evaluating an expression *f* in *x* can be done using the **subs** function.

This does not assign anything to *x*.

```
restart;
f:=x^2+1;
subs(x=1.5,f);
subs(x=2.5,f);
subs(x=3.5,f);
x;
```

$$f := x^2 \quad 1$$
$$3.25$$
$$7.25$$
$$13.25$$
$$x$$

Note, however, that if *x* has a value already assigned to it, you won't get what you want !

```
x:=1;
subs(x=3.5,f);
```

$$x := 1$$
$$2$$

You can also use the **subs** function to substitute one expression for another.
It will do several substitutions at the same time.

```
restart;
subs(x=y+5,x^5*sin(x));
subs(x=y+5,z=y-5,x^2+y^2+z^2);
simplify(%);
```

$$(y \quad 5)^5 \sin(y \quad 5)$$
$$(y \quad 5)^2 \quad y^2 \quad (y \quad 5)^2$$
$$3\,y^2 \quad 50$$

We can illustrate this with the process of simplifying a general cubic equation

```
cub:=a*x^3+b*x^2+c*x+d;
t:=subs(x=y+k,cub);
collect(t,y);
```

$$cub := a\,x^3 \quad b\,x^2 \quad c\,x \quad d$$
$$t := a\,(y \quad k)^3 \quad b\,(y \quad k)^2 \quad c\,(y \quad k) \quad d$$
$$a\,y^3 \quad (3\,a\,k \quad b)\,y^2 \quad \left(3\,a\,k^2 \quad c \quad 2\,b\,k\right)y \quad a\,k^3 \quad d \quad c\,k \quad b\,k^2$$

Now we replace *k* by *b/3a* to remove the y^2 term. This substitution is known as a *Tschirnhaus transformation* after the 17th Century German mathematician who first used it. It is analagous to the process of completing the square for quadratic equations and is the first stage in reducing a cubic equation to a form in which it can be solved.

```
subs(k=-b/(3*a),%);
```

$$a\,y^3 \quad \left(\frac{1}{3}\,\frac{b^2}{a} \quad c \right) y \quad \frac{2}{27}\,\frac{b^3}{a^2} \quad d \quad \frac{1}{3}\,\frac{c\,b}{a}$$

Differentiating

Maple will differentiate expressions. You have to tell it what to differentiate with respect to.
Anything else will be treated as if it were a constant.
This process is actually called *Partial Differentiation*.

```
diff(x^3,x);
diff(a*x^2+5,x);
diff(a*x^2+5,a);
```

$$3\,x^2$$

$$2\,a\,x$$

$$x^2$$

If you try to differentiate with repect to something which has had a value assigned to it, Maple will complain. Unassign the variable or use **restart** to be safe !

```
x:=1;
diff(x^4,x);
```

$$x := 1$$

```
Error, invalid input: diff received 1, which is not valid for its
2nd argument
```

```
x:='x';
diff(x^4,x);
```

$$x := x$$

$$4\,x^3$$

Maple uses the function **Diff** (with a capital letter) to write out the formula for the derivative, but without actually doing the differentiation.

```
Diff(x^4,x);
Diff(x^9,x)=diff(x^9,x);
```

$$\frac{d}{dx}\left(x^4\right)$$

$$\frac{d}{dx}\left(x^9\right) = 9\,x^8$$

Maple will differentiate more than once -- with respect to the same variable or different variables.

```
diff(x^3+y^3+3*x^2*y^2,x,x);
diff(x^3+y^3+3*x^2*y^2,x,y);
```

$$6\,x \quad 6\,y^2$$

$$12\,x\,y$$

As a short cut, if you want to differentiate (say) 4 times wrt the same variable, you can use **x$4**.

```
diff((1+x^2)^3,x$4);
```

$$360\,x^2 \quad 72$$

The French mathematician *Adrian-Marie Legendre* (1752 to 1833) defined some important polynomials in connection with solving the problem of how the gravitational effects of the moon and sun affected the tides.
In the past, mathematicians had to look up the coefficients in tables, but Maple can calculate them very

easily.

```
n:=7;
diff((x^2-1)^n,x$n)/(n!*2^n);
collect(%,x);
```

$$n := 7$$

$$x^7 \quad \frac{21}{2}\left(x^2 \quad 1\right) x^5 \quad \frac{105}{8}\left(x^2 \quad 1\right)^2 x^3 \quad \frac{35}{16}\left(x^2 \quad 1\right)^3 x$$

$$\frac{429}{16} x^7 \quad \frac{693}{16} x^5 \quad \frac{315}{16} x^3 \quad \frac{35}{16} x$$

Formatting worksheets

To put in comments (like this paragraph!) when the cursor is at a Maple prompt **>** either use the **Insert text** item from the **Insert** menu, or the keyboard shortcut **Control-T** If you click on the **T** on the Tool bar it puts in a paragraph for a comment after the present paragraph.

When you have finished, start a new *Execution group* (what Maple calls the group enclosed by the bracket at the left) by using the item **Execution group** in the **Insert** menu, by using one of the keyboard shortcuts **Control-K** (after) or **Control-J** (before) or clicking on the **[>** on the Tool bar.

You can use the same method to get a new execution group anywhere in your worksheet and then, if you wish, you can use this to insert some explanatory text. The **Edit** menu has a **Join** command which lets you put the comment in the same group as the command.

You can also put comments on the same line as Maple input.

```
x:=y^2;
```

this assigns the value y^2 to the variable x.
You get the y^2 by inserting "non-executable maths text".

$$x := y^2$$

Collapsing and expanding

You can make a "collapsible section" which you can 'expand' by clicking on the + symbol or 'contract 'by clicking on the - symbol. Do this by selecting what you want to put into it and then selecting **Indent** from the **Format** menu or the symbol from the Tool bar. To get rid of such a section select **Outdent** from the **Format** menu or the symbol from the Tool bar. When you have made such a section you can type a heading next to its symbol to label it.

Defining functions

One can store functions in Maple's "boxes" as well as numbers or expressions.
A function is a "rule" for assigning a value to a number.

Note that although we may use x in the definition of a function, the function itself is not an expression in x.
Here x is what is called a "dummy variable".

```
restart;
f:=x->x^3;
f(1.4);
f(y);
```

$$f := x \to x^3$$

$$2.744$$

$$y^3$$

```
g:=y->sin(y);
```

$$g := y \to \sin(y)$$

Once we have defined two such functions we can then compose them by applying one to the other.
It usually matters which order we do this in.

```
f(g(x));
g(f(x));
```

$$\sin(x)^3$$

$$\sin(x^3)$$

If we compose functions which are ratios of linear functions, then we get another such function.
The way in which the coefficients of this function are related to the original ones led the English mathematician *Arthur Cayley* (1821 to 1895) to the invention of matrix multiplication.

```
f:=x->(p*x+q)/(r*x+s);
g:=x->(P*x+Q)/(R*x+S);
f(g(x));
simplify(%);
collect(%,x);
```

$$f := x \to \frac{p\,x \quad q}{r\,x \quad s}$$

$$g := x \to \frac{P\,x \quad Q}{R\,x \quad S}$$

$$\frac{\frac{p\,(P\,x \quad Q)}{R\,x \quad S} \quad q}{\frac{r\,(P\,x \quad Q)}{R\,x \quad S} \quad s}$$

$$\frac{p\,P\,x \quad p\,Q \quad q\,R\,x \quad q\,S}{r\,P\,x \quad r\,Q \quad s\,R\,x \quad s\,S}$$

$$\frac{(p\,P \quad q\,R)\,x \quad p\,Q \quad q\,S}{(r\,P \quad s\,R)\,x \quad r\,Q \quad s\,S}$$

We may use the same method as above to define functions of two (or more) variables.
The pair of variables must be in brackets with a comma between them.

```
f:=(x,y)->x^2+y^2;
f(0,0);
f(1,2);
```

$$f := (x, y) \to x^2 \quad y^2$$

$$0$$

$$5$$

Maple differentiates *expressions*, not *functions*.
If you have defined a function f and want to differentiate it with respect to x, then you will have to turn it into an expression by evaluating it at x by using $f(x)$.

```
f:=x->x^3;
diff(f,x);
```

$$f := x \to x^3$$

$$0$$

```
diff(f(x),x);
```

$$3\,x^2$$

However the "operator" **D** will act on a *function* and return a *function*. Note where you have to put the brackets.

```
D(sin);
f:=x->sqrt(1+x^2);
D(f);
D(tan)(x);
```

$$\cos$$

$$f := x \to \sqrt{1 \quad x^2}$$

$$x \to \frac{x}{\sqrt{1 \quad x^2}}$$

$$1 \quad \tan(x)^2$$

You can even manage higher derivatives like this, though it is a bit tricky!

```
(D@@3)(sin);
```

$$\cos$$

Integration

The process of integrating is much older than differentiating and goes back to the Ancient Greeks. For example, Archimedes' efforts to measure the area of a circle (and hence calculate a value for Pi) in about 250BC are equivalent to trying to integrate a function.
Maple will calculate indefinite integrals when it can, but quite "easy" functions may be difficult even for Maple.
If you differentiate the integral, you should get back to where you started.

```
int(x^5,x);
diff(%,x);
```

$$\frac{1}{6}x^6$$

$$x^5$$

```
g:=int(sqrt(1+sqrt(x)),x);
simplify(g);
h:=diff(g,x);
simplify(h);
```

$$g := \frac{\frac{8}{15}\sqrt{\ } \quad \frac{4}{15}\sqrt{\ }\,(1 \quad \sqrt{x})^{3/2}\,(3\sqrt{x} \quad 2)}{\sqrt{\ }}$$

$$\frac{8}{15} \quad \frac{4}{15}\sqrt{1 \quad \sqrt{x}}\,\sqrt{x} \quad \frac{4}{5}\sqrt{1 \quad \sqrt{x}}\,x \quad \frac{8}{15}\sqrt{1 \quad \sqrt{x}}$$

$$h := \frac{\frac{1}{5}\sqrt{\ }\,\frac{\sqrt{1 \quad \sqrt{x}}\,(3\sqrt{x} \quad 2)}{\sqrt{x}} \quad \frac{2}{5}\sqrt{\ }\,\frac{(1 \quad \sqrt{x})^{3/2}}{\sqrt{x}}}{\sqrt{\ }\,\sqrt{1 \quad \sqrt{x}}}$$

Sometimes Maple can't do it. But it still knows how to get back when it differentiates.

```
int(cos(sqrt(1+x^2)),x);
diff(%,x);
```

$$\int \cos\left(\sqrt{1 \quad x^2}\right) dx$$

$$\cos\left(\sqrt{1 \quad x^2}\right)$$

Sometimes it can do the integral but it isn't much help.

```
int(cos(1+x^3),x);
```

$$\frac{1}{6}\cos(1)\sqrt{\ }\,2^{1/3}\left(\frac{9}{2}\,\frac{2^{2/3}\left(\frac{2}{7}x^6 \quad \frac{2}{3}\right)\sin(x^3)}{\sqrt{\ }\,x^2} \quad \frac{3\,2^{2/3}\,(\cos(x^3)\,x^3 \quad \sin(x^3))}{\sqrt{\ }\,x^2}\right.$$

$$\frac{9}{7}\,\frac{x^7\,2^{2/3}\sin(x^3)\,\mathrm{LommelS1}\left(\frac{11}{6},\frac{3}{2},x^3\right)}{\sqrt{\ }\,(x^3)^{11/6}}$$

$$\left.\frac{3\,x^7\,2^{2/3}\,(\cos(x^3)\,x^3 \quad \sin(x^3))\,\mathrm{LommelS1}\left(\frac{5}{6},\frac{1}{2},x^3\right)}{\sqrt{\ }\,(x^3)^{17/6}}\right)$$

$$\frac{1}{6}\sin(1)\sqrt{\ }\,2^{1/3}\left(\frac{3}{4}\,\frac{x\,2^{2/3}\sin(x^3)}{\sqrt{\ }} \quad \frac{9}{4}\,\frac{x\,2^{2/3}\,(\cos(x^3)\,x^3 \quad \sin(x^3))}{\sqrt{\ }}\right.$$

$$\frac{3}{4}\,\frac{x^7\,2^{2/3}\sin(x^3)\,\mathrm{LommelS1}\left(\frac{5}{6},\frac{3}{2},x^3\right)}{\sqrt{\ }\,(x^3)^{11/6}}$$

$$\left.\frac{9}{4}\,\frac{x^7\,2^{2/3}\,(\cos(x^3)\,x^3 \quad \sin(x^3))\,\mathrm{LommelS1}\left(\frac{11}{6},\frac{1}{2},x^3\right)}{\sqrt{\ }\,(x^3)^{17/6}}\right)$$

Remember, however, that integration should involve a "constant of integration" and so if you integrate a derivative, the answer may look different.

```
f:=(1+x^2)^2;
g:=diff(f,x);
h:=int(g,x);
f-h;
simplify(f-h);
```

$$f := (1 \quad x^2)^2$$

$$g := 4\,(1 \quad x^2)\,x$$

$$h := x^4 \quad 2\,x^2$$

$$(1 \quad x^2)^2 \quad x^4 \quad 2\,x^2$$

$$1$$

Maple will calculate integrals of trigonometric functions which would be very tedious to tackle "by hand". In every case, differentiation
should bring you back to where you started, but it might be a bit of a struggle.

```
f:=tan(x)^3;    g:=int(f,x);     h:=diff(g,x);
simplify(h);

f:=sin(x)^3/(1+cos(x)^3);    g:=int(f,x);     h:=diff(g,x);
simplify(h);    simplify(h-f);
```

$$f := \tan(x)^3$$

$$g := \frac{1}{2}\tan(x)^2 \quad \frac{1}{2}\ln\left(1 \quad \tan(x)^2\right)$$

$$h := \tan(x)\,\left(1 \quad \tan(x)^2\right) \quad \tan(x)$$

$$\tan(x)^3$$

$$f := \frac{\sin(x)^3}{1 \quad \cos(x)^3}$$

$$g := \frac{1}{2}\ln\left(1 \quad \cos(x) \quad \cos(x)^2\right) \quad \frac{1}{3}\sqrt{3}\arctan\left(\frac{1}{3}\,(\ 1 \quad 2\cos(x))\sqrt{3}\right)$$

$$h := \frac{1}{2}\,\frac{\sin(x) \quad 2\cos(x)\sin(x)}{1 \quad \cos(x) \quad \cos(x)^2} \quad \frac{2}{3}\,\frac{\sin(x)}{1 \quad \frac{1}{3}\,(\ 1 \quad 2\cos(x))^2}$$

$$\frac{\sin(x)\,(\cos(x) \quad 1)}{1 \quad \cos(x) \quad \cos(x)^2}$$

$$0$$

Although one can repeatedly integrate a function, there is no shorthand for multiple interation as there is for multiple differentiation.

```
f:=x^2-sin(2*x);
int(%,x);
int(%,x);
int(%,x);
int(%,x);
int(f,x,x,x,x);
```

This only integrates *once*.

$$f := x^2 \quad \sin(2\,x)$$

$$\frac{1}{3}\,x^3 \quad \frac{1}{2}\cos(2\,x)$$

$$\frac{1}{12}\,x^4 \quad \frac{1}{4}\sin(2\,x)$$

$$\frac{1}{60}\,x^5 \quad \frac{1}{8}\cos(2\,x)$$

$$\frac{1}{360}\,x^6 \quad \frac{1}{16}\sin(2\,x)$$

$$\frac{1}{3}\,x^3 \quad \frac{1}{2}\cos(2\,x)$$

Maple will also do definite integrals. It will give an exact answer if it can.
Note the way you specify the range of integration: **x = 0 . . 1** etc

```
int(x^4,x=0..1);
int(sin(x)^2,x=0..Pi);
```

$$\frac{1}{5}$$

$$\frac{1}{2}$$

The calculation mentioned above that Archimedes used to calculate Pi (*the quadrature of the circle*) is equivalent to the integral:

```
int(sqrt(1-x^2),x=-1..1);
```

$$\frac{1}{2}$$

Maple will (sometimes) handle integrals over infinite ranges as well as integrals over ranges where the function goes off to infinity.

```
int(exp(-x),x=0..infinity);
int(exp(x),x=0..infinity);
int(tan(x),x=0..Pi/2);
int(tan(x),x=0..Pi);
```

$$1$$

$$\textit{undefined}$$

Even if it can't work out exactly what the answer is, you can ask it to do a *numerical* integration by using the **evalf** function. It does this using a differnt method from working out an indefinite integral.

```
k:=int(cos(sqrt(1+x^2)),x=0..1);
```



```
evalf(k);
```

$$k := \int_0^1 \cos\left(\sqrt{1 \quad x^2}\right)\, dx$$

$$0.4074043129$$

As in the differentiation case, Maple will write out the formula for the derivative if you ask for **Int** (with a capital letter).

```
Int(cos(x)^2,x);
Int(cos(x)^2,x=0..Pi);
evalf(%);
Int(cos(x)^5,x)=int(cos(x)^5,x);
```

$$\int \cos(x)^2\, dx$$

$$\int_0 \cos(x)^2\, dx$$

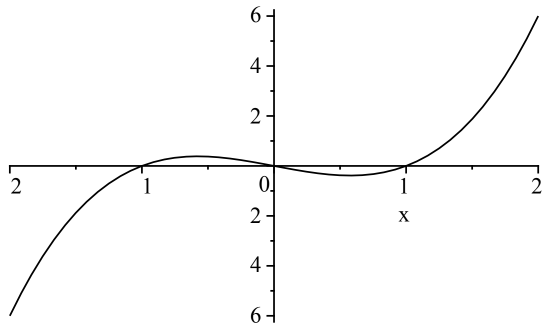
$$1.570796327$$

$$\int \cos(x)^5\, dx = \frac{1}{5}\cos(x)^4\sin(x) \quad \frac{4}{15}\cos(x)^2\sin(x) \quad \frac{8}{15}\sin(x)$$

Plotting

Maple will plot the graph of a function y = *an expresssion involving* x on a given interval which you specify (as in the definite integrals above) by (say) **x = 0 . . 1**.

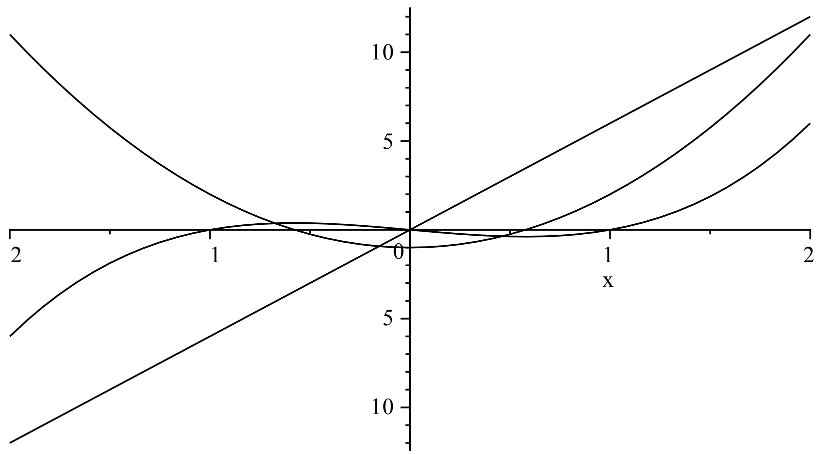
```
plot(x*(x^2-1),x=-2..2);
```



Maple will plot several functions on the same axes. Put the expressions to plot into a list (with [ ] brackets round them) or a set (with { } brackets around them).

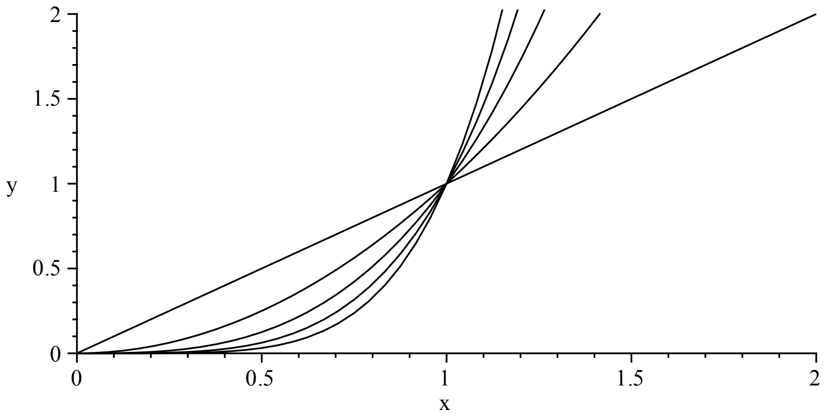
```
f:=x^3-x;
plot([f,diff(f,x),diff(f,x,x)],x=-2..2);
```

$$f := x^3 \quad x$$



Maple will choose the vertical axis so that all the graphs fit in. If you don't want this, you can specify the vertical range.

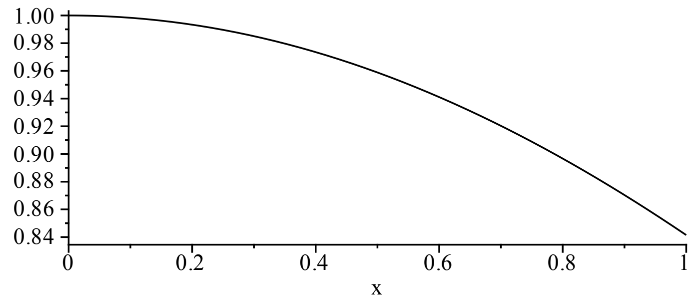
```
plot([x,x^2,x^3,x^4,x^5],x=0..2,y=0..2);
```



Maple will plot *expressions*. If you have defined a function *f* you can turn it into an expression by **f(x)**.

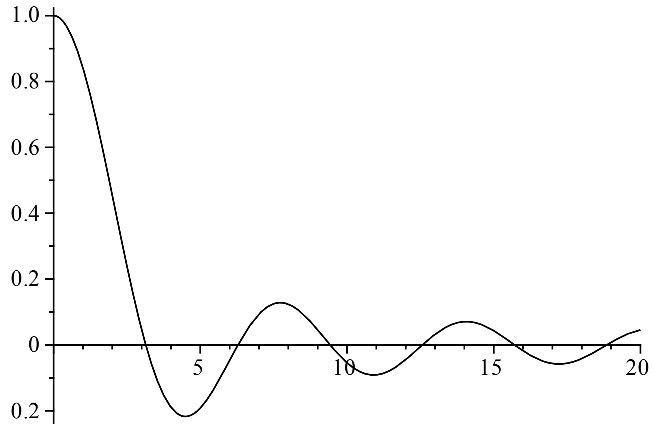
```
f:=x->sin(x)/x;
plot(f(x),x=0..1);
```

$$f := x \rightarrow \frac{\sin(x)}{x}$$



If you want, you can plot the function directly -- but then you musn't mention *x* at all.

```
plot(f,x=0..20);
Error, (in plot) invalid plotting of procedures, perhaps you mean plot(f, 0 ..
20)
plot(f,0..20);
```



One can also specify the colours of the various graphs, or choose to plot them with dots, crosses, ...
You can find out about this using the help facilities on **plot** which you can get by typing in **?plot**.

Solving

Maple will try and solve equations. You have to give it the equation and tell it what to solve for. If there is only one variable in the equation, it will solve for that without being told. If you give it an expression instead of an equation, it will assume you mean *expression* = 0.

```
solve(7*x=22,x);
solve(7*x=22);
solve(7*x-22);
solve(all_my_problems);
```

$$\frac{22}{7}$$

$$\frac{22}{7}$$

$$\frac{22}{7}$$

$$0$$

Maple will solve equations which have more than one solution.

```
solve(a*x^2+b*x+c=0,x);
solve(x^3-2*x^2-5*x+1=0,x);
evalf(%,4);
```

$$\frac{1}{2}\,\frac{b \quad \sqrt{b^2 \quad 4\,a\,c}}{a},\quad \frac{1}{2}\,\frac{b \quad \sqrt{b^2 \quad 4\,a\,c}}{a}$$

$$\frac{1}{6}\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3} \quad \frac{38}{3\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}} \quad \frac{2}{3},\ \frac{1}{12}\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1}$$

$$^{/3} \quad \frac{19}{3\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}} \quad \frac{2}{3} \quad \frac{1}{2}\,\mathrm{I}\,\sqrt{3}\left(\frac{1}{6}\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}\right.$$

$$\left.\frac{38}{3\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}}\right),\ \frac{1}{12}\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}$$

$$\frac{19}{3\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}} \quad \frac{2}{3} \quad \frac{1}{2}\,\mathrm{I}\,\sqrt{3}\left(\frac{1}{6}\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}\right.$$

$$\left.\frac{38}{3\left(316 \quad 12\,\mathrm{I}\,\sqrt{2355}\right)^{1/3}}\right)$$

$$3.390 \quad 0.0003\,\mathrm{I},\ 1.576 \quad 0.0007660\,\mathrm{I},\ 0.1871 \quad 0.0009660\,\mathrm{I}$$

A short cut if you only want to see the decimal expansion is to use the function **fsolve.**
Usually, **fsolve** will only give the real roots of the equation. There are ways of getting complex roots out of it. If you want to know what they are then you can consult the **Help** facilities for **fsolve**.

```
fsolve(x^3-2*x^2-5*x+1=0,x);
```


$$1.575773473,\ 0.1872837251,\ 3.388489748$$

```
fsolve(sin(x),x=3..4);
```

$$3.141592654$$

Maple will solve simultaneous equations
You have to enter the equations as a "set" (with { } round them and commas between).
If you want to solve for several variables, you have to enter these as a set too. If you leave out the variables you want to solve for, Maple will assume you want all of them.

```
solve({y=x^2-4,y=-2*x-2},{x,y});
solve({y=x^2-4,y=-2*x-2});
evalf(%);
```

$$\{y = \quad 2\,RootOf(2\,\_Z \quad 2 \quad \_Z^2, label = \_L2) \quad 2,\ x = RootOf(2\,\_Z \quad 2 \quad \_Z^2, label = \_L2)\}$$

$$\{x = RootOf(2\,\_Z \quad 2 \quad \_Z^2, label = \_L4),\ y = \quad 2\,RootOf(2\,\_Z \quad 2 \quad \_Z^2, label = \_L4) \quad 2\}$$

$$\{y = \quad 3.464101615,\ x = 0.7320508076\}$$

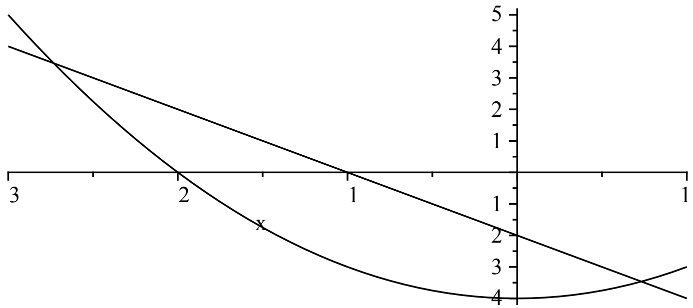
and of course, you can use **fsolve** here too.

```
fsolve({y=x^2-4,y=-2*x-2},{x,y});
```

$$\{y = \quad 3.464101615,\ x = 0.7320508076\}$$

You can then use the plot facility to see the intesection of the two curves. Then you see that we've missed one of the solutions

```
plot({x^2-4,-2*x-2},x=-3..1);
```



We can find the missing one by specifying ranges

```
fsolve({y=x^2-4,y=-2*x-2},{x=-3..0,y=0..5});
```

$$\{x = \quad 2.732050808,\ y = 3.464101615\}$$

If there is more than one solution you can pick out the one you want using [1] or [2] or ...

```
f:=x^2+3*x+1;
sol:=solve(f=0,x);
evalf(sol[1]);
```

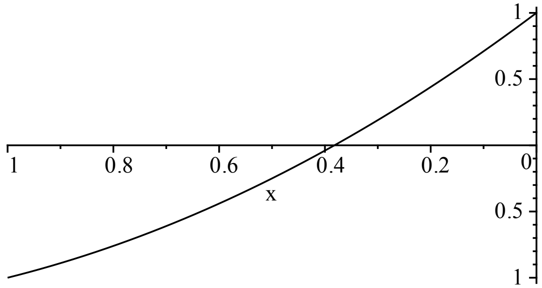
$$f := x^2 \quad 3\,x \quad 1$$

$$sol := \frac{1}{2}\sqrt{5} \quad \frac{3}{2}, \quad \frac{3}{2} \quad \frac{1}{2}\sqrt{5}$$

0.381966012

and then you can draw the graph to see where it is

```
plot(x^2+3*x+1,x=-1..0);
```



As an illustration of how this can be used, we will calculate the equation of the tangent to a curve $y = f(x)$ at some point x_0 say.
Recall that if the tangent is $y = mx + c$ the gradient m is the derivative at $x = x_0$. Then we have to choose the constant c so that the line goes through the point $(x_0, f(x_0))$

```
f:=x->sin(x);
x0:=0.6;

fd:=diff(f(x),x);
m:=subs(x=x0,fd);
c0:=solve(f(x0)=m*x0+c,c);
y=m*x+c0;
plot([f(x),m*x+c0],x=0..1,y=0..1);
```

$$f := x \rightarrow \sin(x)$$

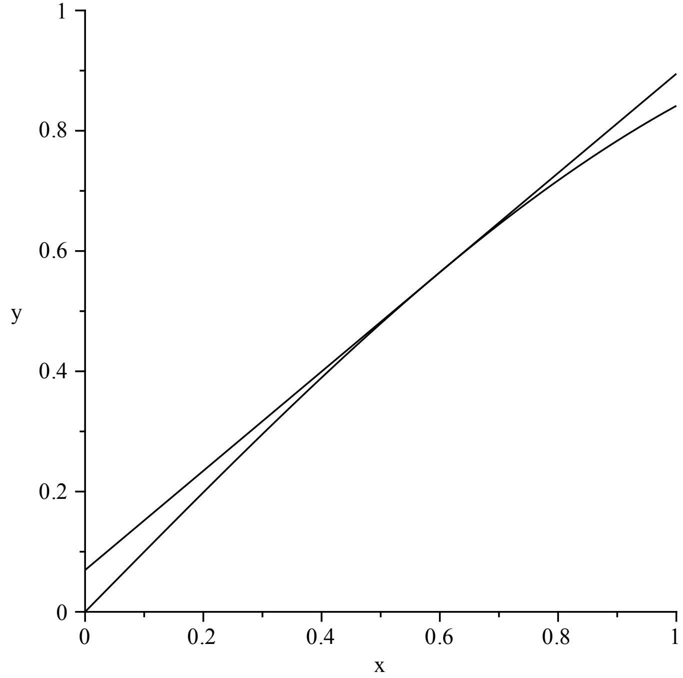
$$x0 := 0.6$$

$$fd := \cos(x)$$

$$m := \cos(0.6)$$

$$c0 := 0.06944110450$$

$$y = 0.8253356149\,x \quad 0.06944110450$$



You could now go back and alter the function f and the point x_0 and run the same bit of code to calculate the equation of the tangent to anything!

As a further illustration, we consider the problem of finding a tangent to a circle from a point outside the circle.

The circle (well, semi-circle, actually) can be specified by $y = \sqrt{1 \quad x^2}$ and we'll find a tangent to it from the point (say) (0, 2).
We first use the same method as above to calculate the equation of the tangent to the curve at a *variable* point x_0
We'll begin by unassigning all our variables and then use a similar bit of code to that used above.

```
restart;
f:=x->sqrt(1-x^2);

fd:=diff(f(x),x);
m:=subs(x=x0,fd);
c0:=solve(f(x0)=m*x0+c,c);
y=m*x+c0;
```

$$f := x \rightarrow \sqrt{1 \quad x^2}$$

$$fd := \frac{x}{\sqrt{1 \quad x^2}}$$

$$m := \frac{x0}{\sqrt{1 \quad x0^2}}$$

$$c0 := \frac{1}{\sqrt{1 \quad x0^2}}$$

$$y = \frac{x0\,x}{\sqrt{1 \quad x0^2}} \qquad \frac{1}{\sqrt{1 \quad x0^2}}$$

Then vary x_0 until the tangent goes through the point (0,2).
We'll make x_1 the value of x_0 when this happens.
Since this produces two answers, we'll choose one of them.
The gradient m_1 of the tangent is then the value of m at this point and the intercept c_1 on the y-axis is the value of c_0 at this point.
So we can find the equation of the tangent: $\boldsymbol{y} = m_1 \boldsymbol{x} + c_1$.

```
x1:=solve(subs(x=0,y=2,y=m*x+c0),x0)[1];
m1:=subs(x0=x1,m);
c1:=subs(x0=x1,c0);
y=m1*x+c1;
```

$$x1 := \frac{1}{2}\sqrt{3}$$

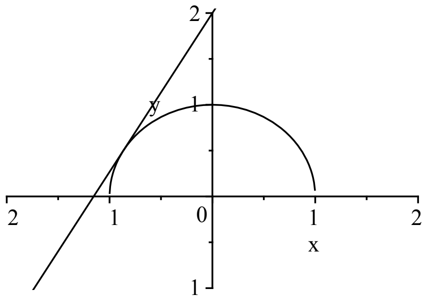
$$m1 := \frac{1}{2}\sqrt{4}\,\sqrt{3}$$

$$c1 := \sqrt{4}$$

$$y = \frac{1}{2}\sqrt{4}\,\sqrt{3}\,x \qquad \sqrt{4}$$

We'll plot the answer to see if it really works.

```
plot([f(x),m1*x+c1],x=-2..2,y=-1..2);
```



Asking Maple for the *second* solution above would give *another* tangent to the circle.

You could now replace $x = 0$, $y = 2$ by any other point and work out the equation of the tangents going through that point.

Looping

There are several ways to get Maple to perform what is called a "loop".
The first is a **for-loop**. You put what you want done between the **do** and the **end do**.

```
for i from 1 to 5 do
x:=i;
end do;
```

$$x := 1$$

$$x := 2$$

$$x := 3$$

$$x := 4$$

$$x := 5$$

It is often better to stop Maple printing out everything it does. You can do this by putting a **:** (a colon) after the loop instead of **;** (a semi-colon). Putting a **:** instead of a **;** after *any* command will stop Maple from printing it out as it executes the command.
However, if you do want it to print something about what is going on, you can ask for it with a **print** command.

```
for i from 1 to 5 do
x:=i;
print(x);
end do:
```

1

2

3

4

5

If you want to include some text, you can include it in "back quotes" (between the left-hand shift and the z).
Single words (which Maple will interpret as variables) can get away without quotes, but more than one word can't. See below for the effect of the usual ".

```
print(one_word);
print(`two words`);
print("two words");
```

one_word

two words

"two words"

```
print(two words);
Error, missing operator or `;`
```

There is another forms of **for-loop**: one in which we get the variable to increase itself by more than one between implementing the things in the loop:

```
for i from 1 to 15 by 3 do
x:=i;
```

```
print(`the value of x is `,x);
end do:
```

the value of x is , 1

the value of x is , 4

the value of x is , 7

the value of x is , 10

the value of x is , 13

As an illustration of what to do with a loop, we calculate the sum of an *Arithmetic Progression* (AP) to several terms e.g. 3, 5, 7, 9, ... We will suppress printing in the loop.
Suppressing printing has the effect of speeding things up, since it takes Maple much longer to print than to calculate.

```
a:=1;d:=2;
term:=a:
total:=0:
for i from 1 to 100 do
total:=total+term;
term:=term+d;
end do:
total;
```

$$a := 1$$

$$d := 2$$

$$10000$$

In a similar way, we can calculate the sum of a *Geometric progression* (a GP) like 3, 6, 12, 24, ...

```
a:=3;r:=2;
term:=a:
total:=0:
for i from 1 to 20 do
total:=total+term;
term:=term*r;
end do:
total;
```

$$a := 3$$

$$r := 2$$

$$3145725$$

As an illustration of another kind of loop we answer the question of how many terms of a GP we need to take before the sum is > (say) 10000 ?
We use a **while-loop** which will be implemented until the "boolean expression" in the first line of the loop becomes False. A *boolean expression*, named after the English mathematician *George Boole* (1815-1864) who was one of the first to apply mathematial techniques to logic, is something which takes the values either **True** or **False**.

```
a:=3;
r:=1.1;

term:=a:total:=0:
count:=0:
while total < 10000 do
total:=total+term;
term:=term*r;
```



```
count:=count+1;
end do:
count;
```

$$a := 3$$

$$r := 1.1$$

$$61$$

If clauses

We now show how Maple can make a choice of several different things to do. This branching is controlled by something called an **if-clause**. We put what we want Maple to do between the **then** and the **end if**.
Here is an example.

```
a:=90;
if a>60 then print(`a is bigger than 60 `);end if;
```

$a := 90$

a is bigger than 60

In the above, if the *boolean expression* (between the **if** and the **then**) is *False*, then nothing gets done. We can alter that:

```
a:=50;
if a > 60 then print(`a is bigger than 60 `);
else print(`a is less than or equal to 60 `);end if;
```

$a := 50$

a is less than or equal to 60

You can put in lots of other alternatives using **elif** (which stands for *else if*).

```
a:=50;
if a > 60 then print(`a is bigger than 60`);
elif a >= 40 then print(`a is greater than or equal to 40 but
less than or equal to 60 `);
else print(`a is less than 40`);end if;
```

$a := 50$

a is greater than or equal to 40 but less than or equal to 60

Lists and Sets

A list in Maple is an ordered set and is written with [ ].

It is often convenient to put results of calculations into such a list. The *n*th element of a list *L* (say) can then be referred to later by **L[n]**.
You can treat the elements of a list as variables and assign to them.

```
A:=[1,2,3];
A[3];
A[2]:=55;
A;
```

$A := [1, 2, 3]$

3

$A_2 := 55$

[1, 55, 3]

You can't assign to an element that isn't there!

```
A[5]:=22;
```



```
Error, out of bound assignment to a list
```

The elements of a list are **op**(*A*) which stands for the *operands* of *A*.

To add extra elements:

```
L:=[];
for i from 1 to 100 do L:=[op(L),i^2];end do:
L;
```

$L := [\ ]$

[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 2401, 2500, 2601, 2704, 2809, 2916, 3025, 3136, 3249, 3364, 3481, 3600, 3721, 3844, 3969, 4096, 4225, 4356, 4489, 4624, 4761, 4900, 5041, 5184, 5329, 5476, 5625, 5776, 5929, 6084, 6241, 6400, 6561, 6724, 6889, 7056, 7225, 7396, 7569, 7744, 7921, 8100, 8281, 8464, 8649, 8836, 9025, 9216, 9409, 9604, 9801, 10000]

The number of elements in a list is **nops** (= number of operands).

```
nops(L);
```

100

As above one can use a *for* loop to put elements into a list. We could have done the same thing using the **seq** function:

```
M:=[seq(n^2,n=1..100)];
```

M := [1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400, 441, 484, 529, 576, 625, 676, 729, 784, 841, 900, 961, 1024, 1089, 1156, 1225, 1296, 1369, 1444, 1521, 1600, 1681, 1764, 1849, 1936, 2025, 2116, 2209, 2304, 2401, 2500, 2601, 2704, 2809, 2916, 3025, 3136, 3249, 3364, 3481, 3600, 3721, 3844, 3969, 4096, 4225, 4356, 4489, 4624, 4761, 4900, 5041, 5184, 5329, 5476, 5625, 5776, 5929, 6084, 6241, 6400, 6561, 6724, 6889, 7056, 7225, 7396, 7569, 7744, 7921, 8100, 8281, 8464, 8649, 8836, 9025, 9216, 9409, 9604, 9801, 10000]

The elements of a list do not need to be all the same kind and they can even be other lists:

```
N:=[x^2,[1,2],[]];
```

N := [169, [1, 2], [ ]]

You can sort lists:

```
sort([1,6,-4,10,5,7]);
```

[4, 1, 5, 6, 7, 10]

You can do quite useful things with lists. For example, here are all the primes < 1000

```
L:=[]:
```

```
for i from 1 to 1000 do if isprime(i) then L:=[op(L),i];end if;
end do:
L;
```

[2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67, 71, 73, 79, 83, 89, 97, 101, 103, 107, 109, 113, 127, 131, 137, 139, 149, 151, 157, 163, 167, 173, 179, 181, 191, 193, 197, 199, 211, 223, 227, 229, 233, 239, 241, 251, 257, 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 439, 443, 449, 457, 461, 463, 467, 479, 487, 491, 499, 503, 509, 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883, 887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997]

There are 168 of them!

```
nops(L);
```

168

Maple can also deal with sets, which it puts in { }. The elements are not in any particular order, and if an element is "repeated" it will be left out.

```
S:={5,3,6,8};
T:={1,2,2,3};
```

$S := \{3, 5, 6, 8\}$

$T := \{1, 2, 3\}$

You can add an element to a set using **union**:

```
S:=S union {13};
```

$S := \{3, 5, 6, 8, 13\}$

We can get Maple to loop over only certain specified values. We may list these values either as a list:

```
for i in [2,8,5,3] do
x:=i;
print(`the value of x is `,x);
end do:
```

the value of x is , 2

the value of x is , 8

the value of x is , 5

the value of x is , 3

or as a set. If Maple uses a set it will usually put it into order before implementing the commands.

```
for i in {2,8,5,3} do
x:=i;
print(`the value of x is `,x);
end do:
```

the value of x is , 2

the value of x is , 3

the value of x is , 5

the value of x is , 8

You can find out more with the Help command: **?list**

Procedures

We saw earlier how to define a function using an assignment like: **f := x -> x^2;** This is in fact shorthand for using a construction known as a **procedure**. We can get the same effect with:

```
f:=proc(x) x^2; end proc;
f(20);
```

f := **proc**(x) x^2 **end proc**

400

Procedures can act like functions and return a value (like x^2 in the above example) but can implement functions which are more complicated than just evaluating a formula. For example, we can adapt the code we wrote above to define a procedure which returns the number of terms of a Geometric Progression needs for its sum to go past some given value n.
The value returned is the *last thing mentioned* before the **end proc**, or you can put in **return** if you wish. Note that variables which are only needed inside the procedure get *declared to be local*, so that if the same variable names had been used somewhere else, these will not be changed by the procedure.

```
howmanyterms:=proc(x)
local term,total,count;
term:=a:total:=0:
count:=0:
while total<x do
total:=total+term;
term:=term*r;
count:=count+1;
od:
return count;
end proc;
```

howmanyterms := **proc**(x)

local *term*, *total*, *count*;

term := *a*;

total := 0;

count := 0;

while *total* *x* **do** *total* := *total* *term*; *term* := *term* * *r*; *count* := 1 *count* **end do**;

return *count*

end proc

Notice that Maple will *pretty-print* the procedure, indenting the code to indicate where the procedure or loops start and finish.
To call the procedure, we tell Maple what the values of a and r are and then apply the procedure to a number.

```
a:=3;r:=2;
howmanyterms(10000);
```

$a := 3$

$r := 2$

12

We can use an if-clause to define a function.

```
f:=proc(x) if x < 0 then x^2; else x+1; end if;end proc;
f(-1);
f(3);
```

f := **proc**(*x*) **if** *x*   0 **then** *x*^2 **else** *x*   1 **end if end proc**

1

4

We can plot this function, but it is necessary to be a bit careful plotting functions defined by procedures, otherwise Maple gets unhappy.
You can either plot them without mentioning the variables at all, by: **plot(f , -1 . . 1);** or you can do it by putting in some single quotes: **' '**.
If the variable *x* had been assigned to you would have to put that in quotes too.

```
plot(f,-1..1);
plot('f(x)',x=-1..1);
```

We count the primes up to a real number *n*. When we plot this we get a kind of "step function".

```
countprimes:=proc(n)
local count,i;
count:=0;
for i from 1 to n do
if isprime(i) then count:=count+1;end if;
end do;
count;
end proc;
```

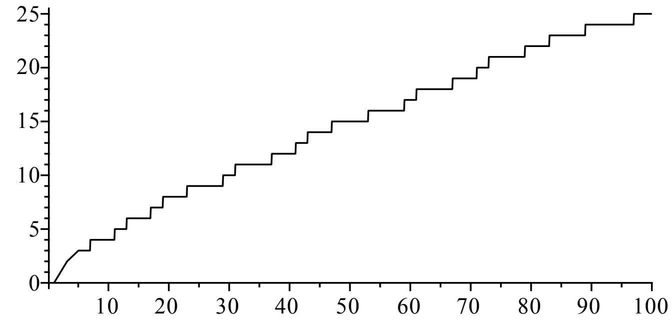
countprimes := **proc**(*n*)

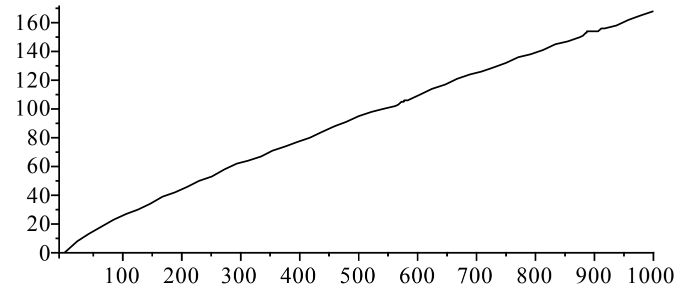
local *count*, *i*;

count := 0; **for** *i* **to** *n* **do if** *isprime*(*i*) **then** *count* := *count*   1 **end if end do**; *count*

end proc

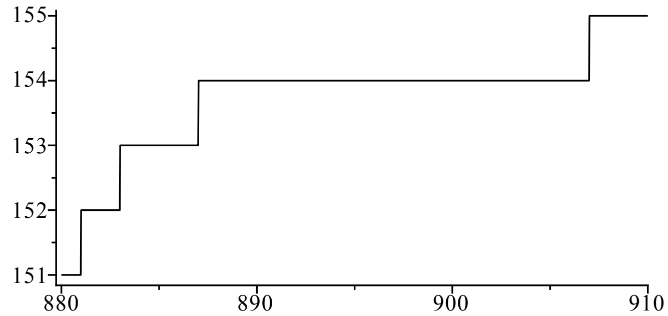
```
plot(countprimes,1..100);
plot(countprimes,1..1000);
```



There is a "flat bit" near 890. In fact a sequence of 19 consecutive non-primes from 888 to 906.

```
plot(countprimes,880..910);
```

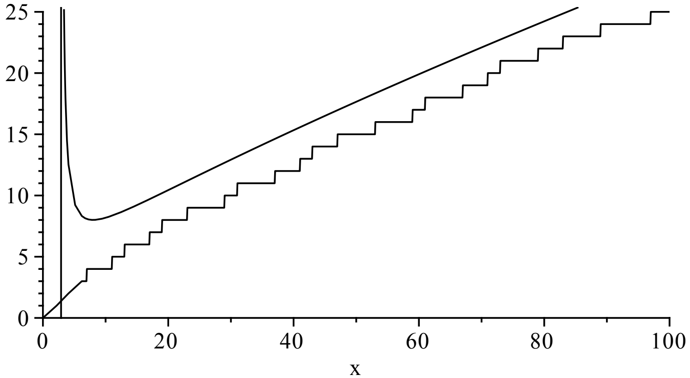


The French mathematician *Adrien-Marie Legendre* (1752 to 1833) approximated the growth of the number of primes with the function:

```
L:=x/(log(x)-1.08);
```

$$L := \frac{x}{\ln(x) - 1.08}$$

```
plot(['countprimes(x)',L],x=0..100,0..25);
```

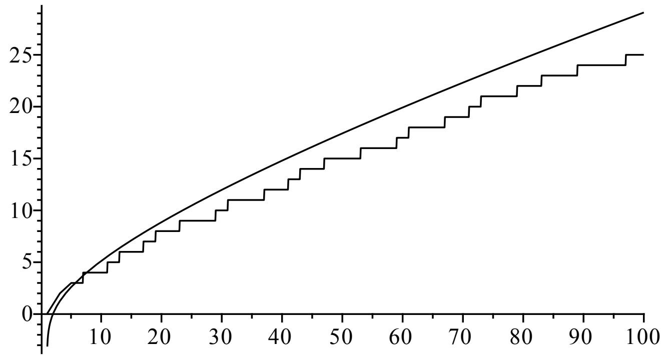


The German mathematician *Carl Friedrich Gauss* (1777 to 1855) used a different approximation: the *logarithmic integral*:

```
logint:=proc(x) int(1/log(t),t=2..x); end proc;
```

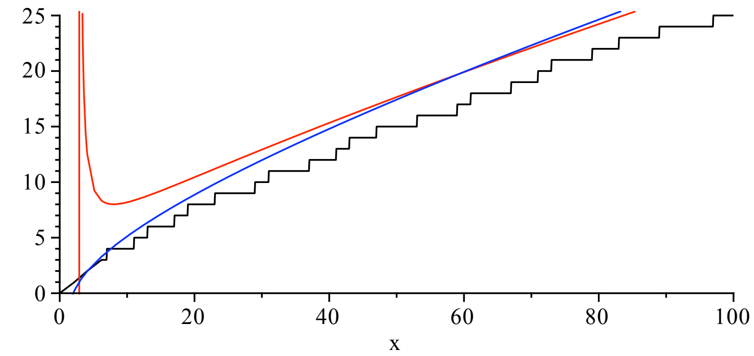
$$logint := \mathbf{proc}(x)\ \ int(1/\log(t), t=2..x)\ \ \mathbf{end\ proc}$$

```
plot([countprimes,logint],1..100);
```



You can see these different approximations together (we'll specify the colours since otherwise Maple will draw one of them in yellow which does not print well):

```
plot(['countprimes(x)',L,'logint(x)'],x=0..100,0..25,colour=
[black,red,blue]);
```

More Procedures

If you want to make a procedure do something to a variable you have already defined outside the procedure, you have to declare it as **global**.
Note that if you don't pass any parameters to a procedure, you still have to out in the brackets.
If you don't want to return anything from a procedure, finish it with **return**; or **return( )**;.

```
addonetoN:=proc()
global N;
N:=N+1;
return;
end proc;
```

addonetoN := **proc**( ) **global** *N*; *N* := *N* 1; **return** **end proc**

```
N:=1;
addonetoN();
N;
```

$N := 1$

2

We'll add an element to the end of a list. In this case we don't want to return anything (though we *do* want to alter something!) so we put **return;** at the end of the procedure.

```
addon:=proc(x)
global A;
A:=[op(A),x];
return;
end proc;
```

addon := **proc**(*x*) **global** *A*; *A* := [*op*(*A*), *x*]; **return** **end proc**

Then apply this

```
A:=[1,2,3];
addon(0);
A;
```

$A := [1, 2, 3]$

$[1, 2, 3, 0]$

We could do something similar *without* using a global variable. Notice that in this case the variable *A* does not change.

```
addend:=proc(A,x)
return [op(A),x];
end proc;
```

addend := **proc**(*A*, *x*) **return** [*op*(*A*), *x*] **end proc**

```
A:=[1,2,3];
B:=addend(A,0);
A;
```

$A := [1, 2, 3]$

$B := [1, 2, 3, 0]$

$[1, 2, 3]$

One thing to be careful of is that you cannot use the parameters passed to the procedure as local variables and assign to them.


```
test:=proc(n) n:=n+1; return n end proc;
```

test := **proc**(*n*) *n* := *n* 1; **return** *n* **end proc**

```
test(9);
Error, (in test) illegal use of a formal parameter
```

Reversing

We can use Maple to find the solution to the following problem

Find a four digit number which is multiplied by 4 when its digits are reversed.

We start by defining a procedure which turns the digits of a number into a list.

Note that we test each procedure as we write it

```
digits:=proc(n)
local ans,m,d;
m:=n;ans:=[];
while m<>0 do
d:=m mod 10;m:=(m-d)/10;ans:=[d,op(ans)];
end do;
return ans;
end proc;
```

digits := **proc**(*n*)

local *ans*, *m*, *d*;

m := *n*;

ans := [ ];

while *m* 0 **do**

d := *mod*(*m*, 10); *m* := 1 / 10 * *m* 1 / 10 * *d*; *ans* := [*d*, *op*(*ans*)]

end do;

return *ans*

end proc

```
K:=digits(12345008);
```

$K := [1, 2, 3, 4, 5, 0, 0, 8]$

It's easy to write a list in the opposite order:

```
reverselist:=proc(L)
local i,M;
M:=[];
for i from 1 to nops(L) do
M:=[L[i],op(M)];
end do;
return M;
end proc;
```

reverselist := **proc**(*L*)

local *i*, *M*;

M := [ ]; **for** *i* **to** *nops*(*L*) **do** *M* := [*L*[*i*], *op*(*M*)] **end do**; **return** *M*

```
end proc
```

```
reverselist(K);
```

[8, 0, 0, 5, 4, 3, 2, 1]

Now we have to get a number back from its list of digits

```
buildit:=proc(L)
local ans,i;
ans:=0;
for i from 1 to nops(L) do
ans:=10*ans+L[i];
end do;
ans;
end proc;
```

buildit := **proc**(*L*)

local *ans*, *i*;

ans := 0; **for** *i* **to** *nops*(*L*) **do** *ans* := 10 * *ans* *L*[*i*] **end do**; *ans*

end proc

```
buildit(K);
```

12345008

Put the ingredients together:

```
reversenum:=proc(n)
buildit(reverselist(digits(n)));
end proc;
```

reversenum := **proc**(*n*) *buildit*(*reverselist*(*digits*(*n*))) **end proc**

```
reversenum(78531);
```

13587

Now we can look for our four digit number

```
for n from 1000 to 9999 do
if reversenum(n)=4*n then print(n); end if
end do:
```

2178

Do the same for a 5 digit number

```
for n from 20000 to 25000 do
if reversenum(n)=4*n then print(n); end if
end do:
```

21978

and even for a 6 digit number

```
for n from 200000 to 250000 do
if reversenum(n)=4*n then print(n); end if
end do:
```

219978

So it looks as if we might have a theorem!

```
219999978*4;
```

879999912

An old question asks if one can always make a number "palindromic" by adding it to its reverse


```
n:=1790;
count:=0:
while reversenum(n)<>n do
n:=n+reversenum(n);
print(n);
count:=count+1;
end do:
print(`Palindromic in `,count,` steps`);
```

n := 1790

2761

4433

7777

Palindromic in , 3, *steps*

Numbers to try are 89 or 296 or ... . A number NOT to try is 196

```
n:=196;
count:=0:
while reversenum(n)<>n do
n:=n+reversenum(n);
print(n);
count:=count+1;
end do:
print(`Palindromic in `,count,` steps`);
```

n := 196

887

1675

7436

13783

52514

94039

187088

1067869

10755470

18211171

35322452

60744805

111589511

227574622

454050344

897100798

1794102596

8746117567

16403234045

70446464506

130992928913

450822227944

900544455998

1800098901007

8801197801088

17602285712176

84724043932847

159547977975595

755127757721546

```
Warning, computation interrupted
```


Pythagorean triples

We can use Maple to search for solutions to equations with integer solutions. Such equations are called *Diophantine* after the Greek mathematician *Diophantus of Alexandria* (200 to 284 AD).

For example, looking for Pythagorean triples satisfying $a^2 \quad b^2 = c^2$ we can use the Maple function **type(**x**, integer)** to check whether a number has an integer square root. We take $y \geq x$ since otherwise we will get each pair (x, y) twice.

```
for x from 1 to 20 do
  for y from x to 20 do
   if type(sqrt(x^2+y^2),integer) then
     print(x,y,sqrt(x^2+y^2))
   fi;
  od
od:
```

3, 4, 5

5, 12, 13

6, 8, 10

8, 15, 17

9, 12, 15

12, 16, 20

15, 20, 25

Here are all solutions up to $x = 100$ and $y = 100$.
We'll leave out those which are multiples of others we have found.
We do this by insisting that x and y have no factor bigger than 1 in common.
We arrange this using the **igcd** (= *integer greatest common divisor* or *highest common factor*) function.
Notice how we combine the two conditions with an **and**.

```
L:=[]:
for x from 1 to 100 do
  for y from x to 100 do
    if type(sqrt(x^2+y^2),integer) and igcd(x,y) = 1 then
      L:=[op(L),[x,y,sqrt(x^2+y^2)]];
    fi;
  od:
od:
L;
```

[[3, 4, 5], [5, 12, 13], [7, 24, 25], [8, 15, 17], [9, 40, 41], [11, 60, 61], [12, 35, 37], [13, 84, 85], [16, 63, 65], [20, 21, 29], [20, 99, 101], [28, 45, 53], [33, 56, 65], [36, 77, 85], [39, 80, 89], [48, 55, 73], [60, 91, 109], [65, 72, 97]]

Some other equations

Similarly, we can look for solutions of other equations like $a^2 \quad 2\,b^2 = c^2$ or $2\,a^2 \quad 3\,b^2 = c^2$ or ... Sometimes we don't find any and then we could try and prove *mathematically* that no such solution exists!

```
for a from 1 to 100 do
for b from 1 to 100 do
c:=sqrt(2*a^2+3*b^2);
if type(c,integer) then print(a,b,c); end if;
end do
end do:
```

Another important equation we can explore in this way is *Pell's equation*, named (probably incorrectly) after the English mathematician *John Pell* (1611 to 1685) but in fact studied by the Indian mathematician *Brahmagupta* (598 to 660) much earlier.
It asks for a solution (x, y) to the equation $n\,x^2 \quad 1 = y^2$ where n is a fixed integer.

```
n:=7:
for x from 1 to 2000 do
y:=sqrt(n*x^2+1);
if type(y,integer) then print(x,y); end if;
end do:
```

$$3, 8$$

$$48, 127$$

$$765, 2024$$

```
n:=13:
for x from 1 to 2000 do
y:=sqrt(n*x^2+1);
if type(y,integer) then print(x,y); end if;
end do:
```

$$180, 649$$

A mysterious sequence

The following is an old problem on which a lot of computing time has been spent.

Start with a number. If it is even then divide it by 2, if it is odd multiply it by 3 and add 1.

What happens to the sequence generated in this way ?

Experiment suggests that any sequence eventually "crashes down" to 1 (and then it will cycle 1, 4, 2, 1, 4, 2, ... indefinitely). The code below investigates, firstly how many terms it takes the sequence to reach 1 (this is the variable **count**) and also what is the maximum value that the sequence reaches on its way there (this is the variable **biggest**).

We investigate it for sequences starting at 1, 2, ... , 200.

Note that the expression **a mod 2** gives the remainder when we divide *a* by 2 and so is 0 if *a* is even.

We'll print out the numbers as lists to save space.

```
L1:=[]: L2:=[]:
```



```
for i from 1 to 100 do
a:=i;
count:=0:
biggest:=0:
while a>1 do
if a mod 2=0 then a:=a/2; else a:=3*a+1; end if;
count:=count+1;
if a>biggest then biggest:=a;end if;
od:
L1:=[op(L1),count];
L2:=[op(L2),biggest];
end do:
L1; L2;
```

[0, 1, 7, 2, 5, 8, 16, 3, 19, 6, 14, 9, 9, 17, 17, 4, 12, 20, 20, 7, 7, 15, 15, 10, 23, 10, 111, 18, 18, 18, 106, 5, 26, 13, 13, 21, 21, 21, 34, 8, 109, 8, 29, 16, 16, 16, 104, 11, 24, 24, 24, 11, 11, 112, 112, 19, 32, 19, 32, 19, 19, 107, 107, 6, 27, 27, 27, 14, 14, 14, 102, 22, 115, 22, 14, 22, 22, 35, 35, 9, 22, 110, 110, 9, 9, 30, 30, 17, 30, 17, 92, 17, 17, 105, 105, 12, 118, 25, 25, 25]

[0, 1, 16, 2, 16, 16, 52, 4, 52, 16, 52, 16, 40, 52, 160, 8, 52, 52, 88, 16, 64, 52, 160, 16, 88, 40, 9232, 52, 88, 160, 9232, 16, 100, 52, 160, 52, 112, 88, 304, 20, 9232, 64, 196, 52, 136, 160, 9232, 24, 148, 88, 232, 40, 160, 9232, 9232, 52, 196, 88, 304, 160, 184, 9232, 9232, 32, 196, 100, 304, 52, 208, 160, 9232, 52, 9232, 112, 340, 88, 232, 304, 808, 40, 244, 9232, 9232, 64, 256, 196, 592, 52, 304, 136, 9232, 160, 280, 9232, 9232, 48, 9232, 148, 448, 88]

Things to notice about this output are the fact that successive integers often take the same number of repetitions to reach 1 and the rather curious numbers which occur several times as the largest value attained. They include 52, 88, 160 and most mysterious of all, 9232. You can spend more time playing with this sequence, but be warned, a lot of effort has been spent on it already, and nobody even knows if the sequence eventually gets back to 1 for every integer.

Continued fractions

In an earlier exercise in Spreadsheets we looked at *Continued fractions*:

You can think of calculating the decimal expansion of a (positive) real number as the result of implementing the algorithm:

(*) Make a note of the integer part of the number. Subtract this from the number.
This gives a number x in the range [0,1). If $x \neq 0$ then:

** Multiply x by 10 **

This (perhaps) gives a number ≥ 1.
Now repeat the loop from (*).

We can replace the step at ** ... ** by anything else that makes x bigger.
In particular, if we put in:

** Take the reciprocal $1/x$ of x **

then the sequence of integers we get is the *Continued fraction expansion* of x.

We'll use Maple to calculate the first n terms in this expansion.

To round down a real number x Maple uses **floor(x)**.

```
cf:=proc(r,n)
local ans,s,i;
ans:=[];s:=r;
for i from 1 to n do
ans:=[op(ans),floor(s)];
s:=s-floor(s);
if s<>0 then s:=1/s;end if;
end do;
ans;
end proc;
```

cf := **proc**(r, n)

local *ans*, *s*, *i*;

ans := [ ];

s := *r*;

for *i* **to** *n* **do**

ans := [*op*(*ans*), floor(*s*)]; *s* := *s*   floor(*s*); **if** *s*   0 **then** *s* := 1 / *s* **end if**

end do;

ans

end proc

The continued fraction expansion of a rational number (a fraction) vanishes after a certain number of steps.

```
cf(12346/56789,10);
```

[0, 4, 1, 1, 2, 189, 1, 1, 6, 0]

For irrational numbers we need to work with a decimal expansion and then we must tell Maple what accuracy to work to.

If we want it to work to 20 significant figures, we enter: **Digits:=20;** (Note the capital letter!) It will stay at that until you **restart;** or reassign **Digits**.

```
Digits:=50:
cf(Pi,30);
```

[3, 7, 15, 1, 292, 1, 1, 1, 2, 1, 3, 1, 14, 2, 1, 1, 2, 2, 2, 2, 1, 84, 2, 1, 1, 15, 3, 13, 1, 4]

```
cf(exp(1),40);
```

[2, 1, 2, 1, 1, 4, 1, 1, 6, 1, 1, 8, 1, 1, 10, 1, 1, 12, 1, 1, 14, 1, 1, 16, 1, 1, 18, 1, 1, 20, 1, 1, 22, 1, 1, 24, 1, 1, 26, 1]

Continued fractions of square roots of integers repeat themselves:

```
cf(sqrt(117),40);
```

[10, 1, 4, 2, 4, 1, 20, 1, 4, 2, 4, 1, 20, 1, 4, 2, 4, 1, 20, 1, 4, 2, 4, 1, 20, 1, 4, 2, 4, 1, 20, 1, 4, 2, 4, 1, 20, 1, 4, 2]

Maple has "built-in" continued fractions which you can find out about with:

```
?convert/confrac
```

Sums of two squares

```
restart;
```

Asking which integers can be written as a sum of one or two squares goes back to *Diophantus of Alexandria*. The answer was given by the Dutch mathematician *Albert Girard* in 1625 and by *Fermat* a little later. The first proof was given by *Euler* in 1749.

```
S:={};
for i from 0 to 100 do
for j from i to 100 do
c:=i^2+j^2;
if c<=100 then S:=S union {c}; end if;
end do;end do:
S;
```

$S := \{\ \}$

{0, 1, 2, 4, 5, 8, 9, 10, 13, 16, 17, 18, 20, 25, 26, 29, 32, 34, 36, 37, 40, 41, 45, 49, 50, 52, 53, 58, 61, 64, 65, 68, 72, 73, 74, 80, 81, 82, 85, 89, 90, 97, 98, 100}

Let's see the factorisation of all these numbers.

```
T:=[]:
for i from 1 to nops(S) do T:=[op(T),ifactor(S[i])];end do:
T;
```

$[0, 1, (2), (2)^2, (5), (2)^3, (3)^2, (2)\,(5), (13), (2)^4, (17), (2)\,(3)^2, (2)^2\,(5), (5)^2, (2)\,(13), (29), (2)^5, (2)\,(17), (2)^2\,(3)^2, (37), (2)^3\,(5), (41), (3)^2\,(5), (7)^2, (2)\,(5)^2, (2)^2\,(13), (53), (2)\,(29), (61), (2)^6, (5)\,(13), (2)^2\,(17), (2)^3\,(3)^2, (73), (2)\,(37), (2)^4\,(5),$

$(3)^4, (2)\ (41), (5)\ (17), (89), (2)\ (3)^2\ (5), (97), (2)\ (7)^2, (2)^2\ (5)^2]$

Maple will let you apply a function to every member of a list or set.

```
ifactor(S);
```

$\{0, 1, (2)^3\ (3)^2, (2)^3, (3)^2, (2)^4\ (5), (2)\ (5), (2)^2\ (5), (2)\ (3)^2, (2)^3\ (5), (3)^2\ (5),$
$(2)^2\ (3)^2, (2)^2, (97), (2)\ (7)^2, (3)^4, (2)\ (41), (5)\ (17), (2)\ (3)^2\ (5), (2), (5), (13),$
$(17), (41), (29), (37), (53), (2)^4, (5)^2, (2)\ (13), (2)^5, (2)\ (17), (7)^2, (2)^2\ (13),$
$(2)\ (29), (2)^6, (5)\ (13), (2)^2\ (17), (2)\ (37), (61), (73), (2)\ (5)^2, (89), (2)^2\ (5)^2\}$

The answer is still not obvious.
In fact the numbers which can be written as a sum of two squares are those in which all the primes of the form $4k + 3$ are present as *even* powers in the factorisation.

Fermat also gave a formula for the number of different ways a number could be written as a sum of two squares.

Looking for prime numbers

Recall that a prime number is an integer which is not exactly divisible by any smaller integer except ± 1.
Maple tests for primeness using the function **isprime** which returns the answer *True* or *False*.

For example, we may look for the next prime after (say) 1234567
(Actually, Maple has a function **nextprime** which would do this for us, but let's not spoil the fun.)

```
a:=1234567;
while not isprime(a) do a:=a+1;end do:
a;
```

$a := 1234567$

1234577

We may count the number of primes in any given range. The German mathematician *Gauss* (1777 – 1855) was interested in how the primes were distributed and when he had any free time, he would spent 15 minutes calculating the primes in a "chiliad" (a range of a 1000 numbers). By the end of his life, it is reckoned that he had counted all the primes up to about two million.

```
count:=0:
for i from 15000 to 16000 do
if isprime(i) then count:=count+1;fi;
end do:
count;
```

108

There are some other interesting places to look for primes. The mathematician *Euler* (1707– 1783) discovered that the formula $x^2 - x + 41$ is particularly prime-rich. In fact it produces primes for *every* integer from 1 to 40 (but not, of course, for $x = 41$) and for lots of others as well.

```
howmanyprimes:=proc(n)
local count,x;
count:=0:
for x from 1 to n do
if isprime(x^2-x+41) then count:=count+1;end if;
end do:
count;
end proc;

howmanyprimes(40);
howmanyprimes(400);
```

howmanyprimes := **proc**(*n*)
  local *count*, *x*;
  count := 0;
  for *x* **to** *n* **do if** *isprime*(*x*^2 *x* 41) **then** *count* := *count* 1 **end if end do**;
  count
end proc

40

270

For many years mathematicians have tried to find big primes. The French mathematician *Fermat* (1601 – 1665) best known for his so-called Last Theorem, investigated primes in the sequence 2^n 1.

```
for n from 1 to 50 do if isprime(2^n+1) then print(n,2^n+1);end
if;end do:
```

1, 3

2, 5

4, 17

8, 257

16, 65537

You should note that the values of n which give primes are all of the form 2^m, but that $n = 32$ does not give a prime.
Euler was the first to show (150 years after Fermat guessed that 2^{32} 1 would be prime) that it is composite.
You can use the Maple function **ifactor** (= integer factorise) to verify this.
(In fact nobody knows if the formula 2^{2^m} 1 produces any other primes after $m = 4$, though a lot of effort has gone into looking for them.)

```
a:=2^32+1; ifactor(a);
```

$a := 4294967297$

$(641)\ (6700417)$

One of Fermat's correspondents was the mathematician *Mersenne* (1588 – 1648). He too investigated primes and looked at numbers of the form 2^n 1.

```
for n from 1 to 150 do
m:=2^n-1;
if isprime(m) then print(n,m);end if;
end do:
```

2, 3

3, 7

5, 31

7, 127

13, 8191

17, 131071

19, 524287

31, 2147483647

61, 2305843009213693951

89, 618970019642690137449562111

107, 162259276829213363391578010288127

127, 170141183460469231731687303715884105727

In fact it is fairly easy to show that if n is not prime the neither is $2^n - 1$. For example, $2^{35} - 1$ is divisible by $2^5 - 1$ and by $2^7 - 1$.

```
(2^35-1)/(2^5-1);(2^35-1)/(2^7-1);
```

1108378657

270549121

Prime numbers of the form $2^n - 1$ are called Mersenne primes and they are almost always the largest primes known. This is because a French mathematician called *Lucas* (1842 – 1891) invented a test for such primes using the Fibonacci numbers. In 1876 he proved that the number $2^{127} - 1$ (see above) is prime. This was the largest known prime until people started using computers in the 1950's. At present 47 Mersenne primes are known. The most recent is $2^{42643801} - 1$ and was discovered on 12th April 2009 using GIMPS (the Great InterNet Mersenne Prime Search). The largest (and largest known prime) is the 45th to be discovered on August 23rd 2008 and is $2^{43112609} - 1$.

With a bit of effort, we can show that it has 12 978 189 decimal digits (and won a prize of $100 000 for being the first one found with more than 10 million digits).

```
evalf(43112609*log10(2));
```

$1.297818850\ 10^7$

This is (well) outside the range of Maple, but you can test the next after those listed above.

```
isprime(2^521-1);
```

true

The Linear Algebra package

A lot of clever stuff has been written for Maple and has been put into *Packages*.

To see what packages are available type in **?index,package**.
Here is one of them.

The *LinearAlgebra* package lets you work with Matrices (and vectors).
Note that all the commands in this package have capital letters.

```
restart;
with(LinearAlgebra);
```

[*&x*, *Add*, *Adjoint*, *BackwardSubstitute*, *BandMatrix*, *Basis*, *BezoutMatrix*, *BidiagonalForm*, *BilinearForm*, *CharacteristicMatrix*, *CharacteristicPolynomial*, *Column*, *ColumnDimension*, *ColumnOperation*, *ColumnSpace*, *CompanionMatrix*, *ConditionNumber*, *ConstantMatrix*, *ConstantVector*, *Copy*, *CreatePermutation*, *CrossProduct*, *DeleteColumn*, *DeleteRow*, *Determinant*, *Diagonal*, *DiagonalMatrix*, *Dimension*, *Dimensions*, *DotProduct*, *EigenConditionNumbers*, *Eigenvalues*, *Eigenvectors*, *Equal*, *ForwardSubstitute*, *FrobeniusForm*, *GaussianElimination*, *GenerateEquations*, *GenerateMatrix*, *GetResultDataType*, *GetResultShape*, *GivensRotationMatrix*, *GramSchmidt*, *HankelMatrix*, *HermiteForm*, *HermitianTranspose*, *HessenbergForm*, *HilbertMatrix*, *HouseholderMatrix*, *IdentityMatrix*, *IntersectionBasis*, *IsDefinite*, *IsOrthogonal*, *IsSimilar*, *IsUnitary*, *JordanBlockMatrix*, *JordanForm*, *LA_Main*, *LUDecomposition*, *LeastSquares*, *LinearSolve*, *Map*, *Map2*, *MatrixAdd*, *MatrixExponential*, *MatrixFunction*, *MatrixInverse*, *MatrixMatrixMultiply*, *MatrixNorm*, *MatrixPower*, *MatrixScalarMultiply*, *MatrixVectorMultiply*, *MinimalPolynomial*, *Minor*, *Modular*, *Multiply*, *NoUserValue*, *Norm*, *Normalize*, *NullSpace*, *OuterProductMatrix*, *Permanent*, *Pivot*, *PopovForm*, *QRDecomposition*, *RandomMatrix*, *RandomVector*, *Rank*, *RationalCanonicalForm*, *ReducedRowEchelonForm*, *Row*, *RowDimension*, *RowOperation*, *RowSpace*, *ScalarMatrix*, *ScalarMultiply*, *ScalarVector*, *SchurForm*, *SingularValues*, *SmithForm*, *SubMatrix*, *SubVector*, *SumBasis*, *SylvesterMatrix*, *ToeplitzMatrix*, *Trace*, *Transpose*, *TridiagonalForm*, *UnitVector*, *VandermondeMatrix*, *VectorAdd*, *VectorAngle*, *VectorMatrixMultiply*, *VectorNorm*, *VectorScalarMultiply*, *ZeroMatrix*, *ZeroVector*, *Zip*]

The above lets you use all the functions it lists. If you don't want to see this list, put **:** instead of **;** when you enter **with(...)**.

If you ever use **restart;** you will have to read the package in again.

A *matrix* is a "box of numbers". There are several ways to enter matrices. You tell Maple the number of rows and columns (or just how many rows if you want a square one).
The second parameter is a list of lists. Maple will put in 0 if you don't tell it what the entry is.

```
A:=Matrix(2,[[a,b],[c,d]]);
B:=Matrix(2,3,[[a,b,c],[d,e,f]]);
C:=Matrix(3,2,[[a,b],[c]]);
Z:=Matrix(2,1);
```

$$A := \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$B := \begin{bmatrix} a & b & c \\ d & e & f \end{bmatrix}$$

$$C := \begin{bmatrix} a & b \\ c & 0 \\ 0 & 0 \end{bmatrix}$$

$$Z := \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

You can enter a matrix by *rows*: written < a | b | c > or by *columns*: < a , b , c > and then rows of columns or columns of rows.

There is something called the matrix palette on the **View** menu which can help,

```
A:=<<a|b|c>,<d|e|f>,<g|h|i>>;
B:=<<a,b,c>|<d,e,f>|<g,h,i>>;
```

$$A := \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}$$

$$B := \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

You can initialise the entries of a matrix using a double **for**-loop or you can use the following:

```
M := Matrix(3,5,(i,j) -> i+2*j);
```

$$M := \begin{bmatrix} 3 & 5 & 7 & 9 & 11 \\ 4 & 6 & 8 & 10 & 12 \\ 5 & 7 & 9 & 11 & 13 \end{bmatrix}$$

You can even get a matrix with "unassigned variables" for all the entries.

```
M := Matrix(3,(i, j) -> m[i,j]);
```

$$M := \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix}$$

You can add or subtract matrices of the same size and can multiply them by a number (or a variable) using *.

```
A:=<<a|b>,<c|d>>;
```



```
B:=<<e|f>,<g|h>>;
C:=<<p|q>,<r|s>>;
A+B;
A-B;
3*C;
```

$$A := \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$B := \begin{bmatrix} e & f \\ g & h \end{bmatrix}$$

$$C := \begin{bmatrix} p & q \\ r & s \end{bmatrix}$$

$$\begin{bmatrix} a \quad e & b \quad f \\ c \quad g & d \quad h \end{bmatrix}$$

$$\begin{bmatrix} a \quad e & b \quad f \\ c \quad g & d \quad h \end{bmatrix}$$

$$\begin{bmatrix} 3\,p & 3\,q \\ 3\,r & 3\,s \end{bmatrix}$$

You can multiply together matrices of compatible shapes by **A.B**; and you can take powers of matrices by (for example) **A^3;** or **A^(-1);** (giving the *inverse* of the matrix).

```
A.B;A^3;A^(-1);
```

$$\begin{bmatrix} a\,e \quad b\,g & a\,f \quad b\,h \\ c\,e \quad d\,g & c\,f \quad d\,h \end{bmatrix}$$

$$\begin{bmatrix} (a^2 \quad b\,c)\,a \quad (a\,b \quad b\,d)\,c & (a^2 \quad b\,c)\,b \quad (a\,b \quad b\,d)\,d \\ (c\,a \quad d\,c)\,a \quad (b\,c \quad d^2)\,c & (c\,a \quad d\,c)\,b \quad (b\,c \quad d^2)\,d \end{bmatrix}$$

$$\begin{bmatrix} \frac{d}{a\,d \quad b\,c} & \frac{b}{a\,d \quad b\,c} \\ \frac{c}{a\,d \quad b\,c} & \frac{a}{a\,d \quad b\,c} \end{bmatrix}$$

Multiplication of square matrices is *associative*: $A.(B.C) = (A.B).C$ and *distributive* $A.(B + C) = A.B + A.C$ but not (in general) *commutative* $A.B \neq B.A$.

```
(A.B).C-A.(B.C);
```

$$[[(a\,e \quad b\,g)\,p \quad (a\,f \quad b\,h)\,r \quad a\,(e\,p \quad f\,r) \quad b\,(g\,p \quad h\,r), (a\,e \quad b\,g)\,q \quad (a\,f \quad b\,h)\,s$$
$$a\,(e\,q \quad f\,s) \quad b\,(g\,q \quad h\,s)],$$
$$[(c\,e \quad d\,g)\,p \quad (c\,f \quad d\,h)\,r \quad c\,(e\,p \quad f\,r) \quad d\,(g\,p \quad h\,r), (c\,e \quad d\,g)\,q \quad (c\,f \quad d\,h)\,s$$
$$c\,(e\,q \quad f\,s) \quad d\,(g\,q \quad h\,s)]]$$

```
simplify(%);
```

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

```
A.(B+C)-(A.B+A.C);
```

$$[[a\,(e \quad p) \quad b\,(g \quad r) \quad a\,e \quad b\,g \quad a\,p \quad b\,r, a\,(f \quad q) \quad b\,(h \quad s) \quad a\,f \quad b\,h \quad a\,q \quad b\,s],$$
$$[c\,(e \quad p) \quad d\,(g \quad r) \quad c\,e \quad d\,g \quad c\,p \quad d\,r, c\,(f \quad q) \quad d\,(h \quad s) \quad c\,f \quad d\,h \quad c\,q \quad d\,s]]$$

```
simplify(%);
```

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

```
A.B-B.A;
```

$$\begin{bmatrix} b\,g \quad c\,f & a\,f \quad b\,h \quad e\,b \quad f\,d \\ c\,e \quad d\,g \quad g\,a \quad h\,c & c\,f \quad b\,g \end{bmatrix}$$

```
simplify(%);
```

$$\begin{bmatrix} b\,g \quad c\,f & a\,f \quad b\,h \quad e\,b \quad f\,d \\ c\,e \quad d\,g \quad g\,a \quad h\,c & c\,f \quad b\,g \end{bmatrix}$$

Here are some useful matrices: RandomMatrix produces a matrix with entries in the range -99 .. 99.

```
IdentityMatrix(4);
ZeroMatrix(2,3);
RandomMatrix(3,2);
```

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

$$\begin{bmatrix} 34 & 21 \\ 62 & 56 \\ 90 & 8 \end{bmatrix}$$

The determinant of a matrix is a combination of the entries of a *square* matrix (in rather a complicated way!) which has the property that it is 0 if the matrix does not have an inverse.

```
M:=<<a|b>,<c|d>>;  Determinant(M);
```

$$M := \begin{bmatrix} a & b \\ c & d \end{bmatrix}$$

$$a\,d \quad b\,c$$


```
N := Matrix(3,(i, j) -> n[i,j]):  Determinant(N);
```

$$n_{1,1}\,n_{2,2}\,n_{3,3} \quad n_{1,1}\,n_{2,3}\,n_{3,2} \quad n_{2,1}\,n_{3,2}\,n_{1,3} \quad n_{2,1}\,n_{1,2}\,n_{3,3} \quad n_{3,1}\,n_{1,2}\,n_{2,3} \quad n_{3,1}\,n_{2,2}\,n_{1,3}$$

We can use Maple to demonstrate a theorem discovered by the English mathematician *Arthur Cayley* and the Irish mathematician *William Hamilton*.

First we take a "general matrix".

```
n:=2;  A:=Matrix(n,(i,j)->a[i,j]);
```

$$n := 2$$

$$A := \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}$$

```
Id:=IdentityMatrix(n,n);
```

$$Id := \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

Then we take the determinant of the matrix *A* - *xI* where *x* is an unassigned variable. This is a polynomial in *x* called the *characteristic polynomial*.

Then the *Cayley-Hamilton theorem* says that the matrix *A* satisfies its characteristic polynomial. That is, if we substitute *A* for *x* in this polynomial, we get the zero matrix.

```
p:=Determinant(A-x*Id);
```

$$p := a_{1,1}\,a_{2,2} \quad a_{1,1}\,x \quad x\,a_{2,2} \quad x^2 \quad a_{1,2}\,a_{2,1}$$

```
p:=collect(p,x);
```

$$p := x^2 \quad (\ a_{1,1} \quad a_{2,2})\,x \quad a_{1,1}\,a_{2,2} \quad a_{1,2}\,a_{2,1}$$

```
Q:=sum(coeff(p,x,k)*A^k,k=0..n);
```

$$Q := a_{1,1}\,a_{2,2} \quad a_{1,2}\,a_{2,1} \quad (\ a_{1,1} \quad a_{2,2}) \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix} \quad \begin{bmatrix} a_{1,1} & a_{1,2} \\ a_{2,1} & a_{2,2} \end{bmatrix}^2$$

```
simplify(Q);
```

$$\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

Sudoku

In Sudoku you are given a partly filled 9 × 9 grid and have to fill in the rest with integers 1 to 9 subject to the conditions that no two entries in any row, column or 3 × 3 subsquare are equal.

The Maple program given here does *not* do it in the same way as an intelligent human, but uses a technique called "backtracking" to make a series of guesses and then if they do not work, goes back and guesses again.

We write a procedure to print the matrix split up into its 9 sub-squares.

```
pprint := proc(M)
local i,j,A,m,n;

 for i from 0 to 2 do
   A:=[Matrix(3),Matrix(3),Matrix(3)];
     for j from 0 to 2 do
      for m from 1 to 3 do
       for n from 1 to 3 do
        if M[3*i+m,3*j+n] =0 then A[j+1][m,n]:=` `;
        else A[j+1][m,n]:=M[3*i+m,3*j+n];end if;
      end do;end do;end do;
   print(A[1],A[2],A[3]);
 end do;
end proc:
```

We calculate which numbers can be put into a given "hole" in the matrix.

```
choice := proc(M, r, c)
 local i,j,ans,r1,c1;

  ans := {1,2,3,4,5,6,7,8,9};
  for i to 9 do
     ans := ans minus {M[r,i]};
     ans := ans minus {M[i,c]}
  end do;
  r1 := iquo(r-1,3); c1 := iquo(c-1,3);
  for i from 3*r1+1 to 3*r1+3 do
    for j from 3*c1+1 to 3*c1+3 do
      ans := ans minus {M[i,j]}
    end do
  end do;
  return ans
end proc:
```

We check to see if we have filled in all the "holes". If we have not, we look for the hole with the fewest number of possibilities and try one of these. We repeat the process until either we have finished or we are stuck. In the latter case we go back and try the next possibility.
The program uses a technique called *recursion* where the procedure calls itself.

```
checkit := proc(M)
 global gotit,soln;
 local i,j,k,N,holes,nextone,ch,nextchoice;

 if not gotit then
   holes:=0;
   nextone:=[0,0,10];
   N:=Matrix(9);
   for i from 1 to 9 do
     for j from 1 to 9 do
         N[i,j]:=M[i,j];
         if M[i,j]=0 then
           holes:=holes+1;
           ch:=choice(M,i,j);
           if nops(ch)<nextone[3] then nextone:=[i,j,nops(ch)];
end if;
         end if;
       end do;
     end do;
     if holes=0 then
       gotit:=true; soln:=M;
     elif nextone[3]<>10 then
       nextchoice:=choice(M,nextone[1],nextone[2]);
       for k in nextchoice do
         N[nextone[1],nextone[2]]:=k;
         checkit(N);
       end do;
     end if;
   end if;
end proc:
```


We feed in a matrix with 0's for the spaces.

```
M:=Matrix(9,
[
[0,0,0,0,0,8,0,5,4],
[0,4,0,0,0,9,0,7,1],
[0,0,0,0,5,0,0,0,0],
[0,0,0,2,0,0,0,6,7],
[0,0,8,0,0,0,9,0,0],
[9,3,0,0,0,4,0,0,0],
[0,0,0,0,3,0,0,0,0],
[5,7,0,6,0,0,0,8,0],
[1,2,0,7,0,0,0,0,0]
]):
```

The computers in the microlab take less that 2 seconds to find the solution!

```
pprint(M);print(` `);
gotit:=false:
checkit(M);
if gotit then pprint(soln); else print(`NO SOLUTION`);end if;
```

$$\begin{bmatrix} & & \\ & 4 & \\ & & \end{bmatrix}, \begin{bmatrix} & & 8 \\ & & 9 \\ & 5 & \end{bmatrix}, \begin{bmatrix} & 5 & 4 \\ & 7 & 1 \\ & & \end{bmatrix}$$

$$\begin{bmatrix} & & \\ & & 8 \\ 9 & 3 & \end{bmatrix}, \begin{bmatrix} 2 & & \\ & & \\ & & 4 \end{bmatrix}, \begin{bmatrix} & 6 & 7 \\ 9 & & \\ & & \end{bmatrix}$$

$$\begin{bmatrix} & & \\ 5 & 7 & \\ 1 & 2 & \end{bmatrix}, \begin{bmatrix} & 3 & \\ 6 & & \\ 7 & & \end{bmatrix}, \begin{bmatrix} & & \\ & 8 & \\ & & \end{bmatrix}$$

$$\begin{bmatrix} 3 & 9 & 6 \\ 8 & 4 & 5 \\ 7 & 1 & 2 \end{bmatrix}, \begin{bmatrix} 1 & 7 & 8 \\ 3 & 2 & 9 \\ 4 & 5 & 6 \end{bmatrix}, \begin{bmatrix} 2 & 5 & 4 \\ 6 & 7 & 1 \\ 3 & 9 & 8 \end{bmatrix}$$

$$\begin{bmatrix} 4 & 5 & 1 \\ 2 & 6 & 8 \\ 9 & 3 & 7 \end{bmatrix}, \begin{bmatrix} 2 & 9 & 3 \\ 5 & 1 & 7 \\ 8 & 6 & 4 \end{bmatrix}, \begin{bmatrix} 8 & 6 & 7 \\ 9 & 4 & 3 \\ 5 & 1 & 2 \end{bmatrix}$$

$$\begin{bmatrix} 6 & 8 & 4 \\ 5 & 7 & 3 \\ 1 & 2 & 9 \end{bmatrix}, \begin{bmatrix} 9 & 3 & 1 \\ 6 & 4 & 2 \\ 7 & 8 & 5 \end{bmatrix}, \begin{bmatrix} 7 & 2 & 5 \\ 1 & 8 & 9 \\ 4 & 3 & 6 \end{bmatrix}$$

You may change one of the entries in the original matrix and verify that no solution exists.

Summary 1

Maple will work like an ordinary calculator. Type in your sum, put a semi-colon after each calculation and then press the *Return* key. You can put more than one calculation on a line, but you must put a **;** after each one.

You can move on to a new line using *Shift-Return*.

Use the usual symbols +, – for addition and subtraction. Use * and / for multiplication and division. Use ^ for *to the power of* and remember that the computer will stick strictly to the correct order when doing arithmetic. Use *round* brackets to alter this order if you want.

If it can Maple will produce exact answers containing fractions or square roots of fractions.

If you want the answer as a *decimal* then use **evalf**(*expression*, [*number of figures*]) which evaluates the expression as a floating point number. (The second argument is optional.)

Use **%** for the last expression that Maple calculated and **%%** for the one before.

Maple recognises the usual functions like **sqrt**, **sin**, **cos**, **tan**, ... , **arcsin**, **arctan**, ... as well as **exp**, **log** = **ln**, **log10**, ... and lots of other functions you've never heard of.

To use them, put ( ) around what you evaluate.

The number π lives in Maple as **Pi**.

Maple has functions your calculator can't handle. For example, the function **ifactor** (for *integer factorise*) can be applied to an integer (or even a fraction) to write it as a product of prime factors and the function **isprime** tells you if an integer is a prime number or not.

The function **factorial** (or **n!**) calculates $1 \times 2 \times 3 \times 4 \times ... \times n$.

expand(*expr*) will "multiply out" an arithmetic expression, a polynomial expression, a ratio of polynomials or a trigonometric expression.

factor will try to factorise a polynomial with integer (or fractions) as coefficients.

sort(*expr*) and **collect**(*expr, x*) will sort the terms of a polynomial and collect terms in x^n together. The coefficients of a polynomial can be obtained using **coeff**(*p, x, n*)

simplify(*expr*) will attempt to simplify an expression. It won't always succeed or necessarily produce what you expect.

You can see the help files for any command by typing **?command** or by highlighting the command and using the **Help** menu.

Summary 2

a:=1; assigns the value **1** to the "store" or "variable" *a*. Lots of things, including numbers and expressions, can be stored in this way.

If you store an unassigned variable *x* (or an expression containing *x*) in a variable *a* then altering *x* will also alter *a*. If the variable *x* already had something assigned to it *before* assigning it to *a*, then *a* will *not* be affected by changing *x*.

a:='a'; unassigns this variable — and makes it back into a proper variable again.

restart; unassigns *all* variables.

subs(*x* = *a*, *y* = *b*, ..., *expr*) substitutes *a* for *x*, *b* for *y*, ... in the expression.

f:= x -> *expr* assigns to the variable *f* the "rule" or "function" that maps *x* to the expression. You can evaluate it at a point *a* say, by **f(a);**

Note that the *function f* is different from the *expression f*(*x*) (which is the function evaluated at the "point" *x*).

diff(*expr*, **x**) differentiates the expression with respect to the variable *x*. You cannot differentiate (or integrate) with respect to a variable which has had something assigned to it. Any other variables in the expression will be treated as constants when differentiating.

You can differentiate twice using **diff**(*expr*, **x,x**) or **diff**(*expr*, **x$2**).

Diff (with a capital letter) returns the *formula* for the derivative, but does not do the differentiation.

To put in comments when the cursor is at a Maple prompt > either use the **Insert text** item from the **Insert menu**, or the keyboard shortcut **Control-T** or click on the **T** on the Tool bar.

When you have finished, start a new *Execution group* (what Maple calls the group enclosed by a bracket at the left) by using the item **Execution group** in the **Insert menu**, by using one of the keyboard shortcuts **Control-J** or **Control-K** or click on the **[>** on the Tool bar.

You can use the same method to get a new execution group anywhere in your worksheet and then, if you wish, you can use this to insert some explanatory text.

Summary 3

plot(*expr*, **x = a..b**) plots the graph of y = the expression, for values of x between a and b.

If you leave out the range: **plot**(*expr*, **x)** then Maple takes it to be $-10 \le x \le 10$.

plot([*expr*1, *expr*2], **x = a..b**, **c..d**) or **plot**([*expr*1, *expr*2], **x = a..b**, **y=c..d**) plots the two graphs on the same axes and uses the specified ranges for the (horizontal) x axis and the vertical axis.

int(*expr*, **x**) calculates the *indefinite* integral wrt x (if it can — integration can be difficult).

int(*expr*, **x = a..b**) calculates the *definite* integral of the expression from a to b. Use **evalf** to get a numerical answer if you need one. You can even make a or b **infinity** or **–infinity** if you want.

Int (capital letter) returns the *formula* for the integral.

If a *function* has been assigned to f, you cannot differentiate, integrate (or plot) f without turning it into an *expression* by, for example, putting it in as (say) **f(x)**. As in, for example

f:= x->x^2; diff(f(x),x);

solve(*equation*, **x**) tries to solve the equation for x. The variable x had better be unassigned.

If you just want to see the floating point answer, you can use **fsolve**(*equation*, **x**).

To get Maple to find solutions lying in a range $a \le x \le b$ you can use **fsolve**(*equation*, **x=a..b**).

If there are several solutions, you can assign them to a list by, for example, **s := solve**(*eqn*, **x**) and then ask for **s[1]**, **s[2]**, etc. to get the first, second, ...

You can solve *simultaneous equations* for (say) x and y using **solve**({*eqn1*, *eqn2*}, **{x, y}**).

Similarly, **fsolve**({*eqn1*, *eqn2*}, **{x, y}**) or **fsolve**({*eqn1*, *eqn2*}, **{x=a..b, y=c..d}**) will give the numerical answers.

The brackets { } indicate that Maple is treating these as *sets*.

Summary 4

You can make Maple repeat a calculation a given number of times using a *for-loop*.

for i from 1 to 100 do ... end do; will go through the instructions between **do** and **end do** 100 times.

If you want to stop Maple printing everything it does, finish the loop with a colon **:** rather than a semi-colon **;**

Using a colon instead of a semicolon stops it printing other stuff too.

You can then ask it to print *some* of what is going on using **print**(*something, something else*).

It will print the things you ask for (with commas between if there is more than one thing) with a new line each time it implements the print statement.

If you want to include text, you have to put in *back-quotes* **\`** **...** **\`** (to get these use the key at the top left of the keyboard).

Examples of other forms of a for-loop are:

for i from 1 to 100 by 3 do ... end do; which increases the value of i by 3 each time it goes through the loop.

You can use: **for i from 10 to 1 by –1 do ... end do;** to decrease i.

for i in [1, 2, 5, 10] do ... end do; makes i take the values in the list successively.

Another form of loop is a *while-loop*. This has the form:

while *boolean expression* **do ... end do;** where the *boolean expression* evaluates to give either *True* or *False*. Maple will repeat the instructions between **do** and **end do** until the boolean expression becomes *False*. If you make a mistake, it might continue for ever, but you can stop it by clicking on **Stop** on the menu-bar.

You can use a boolean expression to choose between alternatives using an *if-clause*.

if *boolean expression* **then ... end if;** will make Maple do what is between **then** and **end if** provided the boolean expression evaluates to *True*.

Alternative forms are: **if** *boolean expression* **then ... else ... end fi;** or the more elaborate:

if *boolean expression* **then ... elif** *another boolean expression* **then ... else ... end if;**

You can put in lots of other **elif** (= else if) bits if you want.

Summary 5

A *list* is an ordered set of elements (or *operands*). **L := [a, b, c, d];**

It lives between **[ ]** and has commas between its elements.

The elements can be numbers, expressions, variables, other lists, ...

The empty list is **[ ]**. You can get at an element of a list (if there is one) by asking for it by its *index* (or number): **L[3]; L[3 .. 5];**

You can assign to elements of a list in the same way as you can to any variable: **L[3] := 4;**

The number of elements in a list *L* is **nops(L)** (which stands for *number of operands*).

A *sequence* is like the contents of a list (without the brackets).

You can make a sequence with (say) **seq(***expression*, **n = 1 .. 20)**; and make this into a list by putting it between **[ ]**. For example **[seq(0, n=1..10)];** produces a sequence of 10 zeros.

The sequence of elements of a list *L* is **op(L)** (which stands for the *operands of L*).

You can add to the elements of a list using **L := [op(L),** *new elements***];**

You can sort a list *L* with **sort(L);**

Sets are like lists, but unordered and with no repetitions. They live inside { }.

You can deal with them using the operators **union**, **intersection** and **minus**.

You can use a for-loop to sum a sequence, but Maple has a special function to do this. **sum(***expression*, **n = 1 .. 20)** sums the terms obtained by substituting *n* = 1, 2, 3, ... , 20 in the expression. You can sometimes get Maple to give the general formula for a sum and can sometimes even sum for **n = 1 .. infinity**.

Summary 6

A **procedure** can be used to define more complicated functions than the ones given using an assignment **f := x –>** *a formula involving x*.

For example, **f := x –> x^2;** is really shorthand for **f := proc(x) x^2; end proc;**

The last expression in the procedure, immediately before **end proc**, is the value that the procedure returns. The parameters inside the brackets are described as being "passed to the procedure". Even if there aren't any such parameters to pass, you still have to put in the brackets!

If the procedure involves a complicated calculation with assignment to extra variables, these can be declared as **local** variables and will not then affect the values of variables used outside the procedure which happen to have the same names. If you don't declare them as local variables, Maple will rather crossly tell you that it has done the job for you.

You cannot use the parameters that are passed to the procedure as variables which can be assigned to.

If you *do* want the procedure to access variables you have already used, you can declare them as **global**. and if you change them inside the procedure, the values ouside will be altered.

To plot a function defined by a procedure, do not mention the variable. For example to plot a function defined by a procedure *fn*:

plot(fn, -1 ..1);

If you do need to mention the variable (for example if you want to simultaneously plot a function given by an expression *ex*, say) then you can enclose things in single quotes.

plot(['fn(x)', ex], x = 0 .. 1);

Summary 7

A lot of clever stuff has been written for Maple and has been put into Packages.
To see what packages are available type in **?index,package**.
To use the commands in (say) the **LinearAlgebra** package enter **with(LinearAlgebra):**
(If you put a **;** you get a list of all the available commands.)
If you ever use **restart;** you will have to read the package in again.

The **LinearAlgebra** package lets you work with *Matrices* (and vectors). Note that all the commands in this package have capital letters.
A:=Matrix(2, [[a, b], [c, d]]); produces a square matrix with rows taken from the list of lists in the second parameter.
B:=Matrix(2, 3, [[a, b, c], [d, e, f]]); makes a matrix with 2 rows and 3 columns. If you don't tell it what the entries are it fills up with zeros.
You can make a matrix of (say) 3 rows by
A := < <a | b | c> , <d | e | f> , <g | h | i> >;
or of 3 columns by **B := < <a , b , c> | <d , e , f> | <g , h , i> >;**
There is something called the matrix palette (on the **View** menu) which can help.

You can make a matrix whose entries are given by some formula by using a double **for**-loop, or by using (say) **C := Matrix(3, 5 ,(i, j) -> i+2*j);**
You can even get a matrix with "undefined" entries by **X := Matrix(3, 3, (i, j) -> x[i, j]);**
You can add and subtract matrices of the same shape. You can multiply a matrix by a constant (or a variable) by **a*M;** You can multiply together matrices of compatible shapes by **A.B;** and you can take powers of matrices by (for example) **A^3;** or **A^(-1);** (giving the *inverse* of the matrix).
Multiplication of square matrices is *associative*: $A.(B.C) = (A.B).C$ and distributive $A.(B + C) = A.B + A.C$ but not (in general) commutative $A.B \neq B.A$.

Some interesting matrices are **IdentityMatrix(m, n); ZeroMatrix(m, n); RandomMatrix(m, n);**

The determinant of a square matrix is a real number which is a complicated combination of the entries of a matrix. It has the property that if the matrix is not invertible than the determinant is 0. Maple calls it **Determinant(M);**
It also satisfies $\det(A.B) = \det(A).\det(B)$.