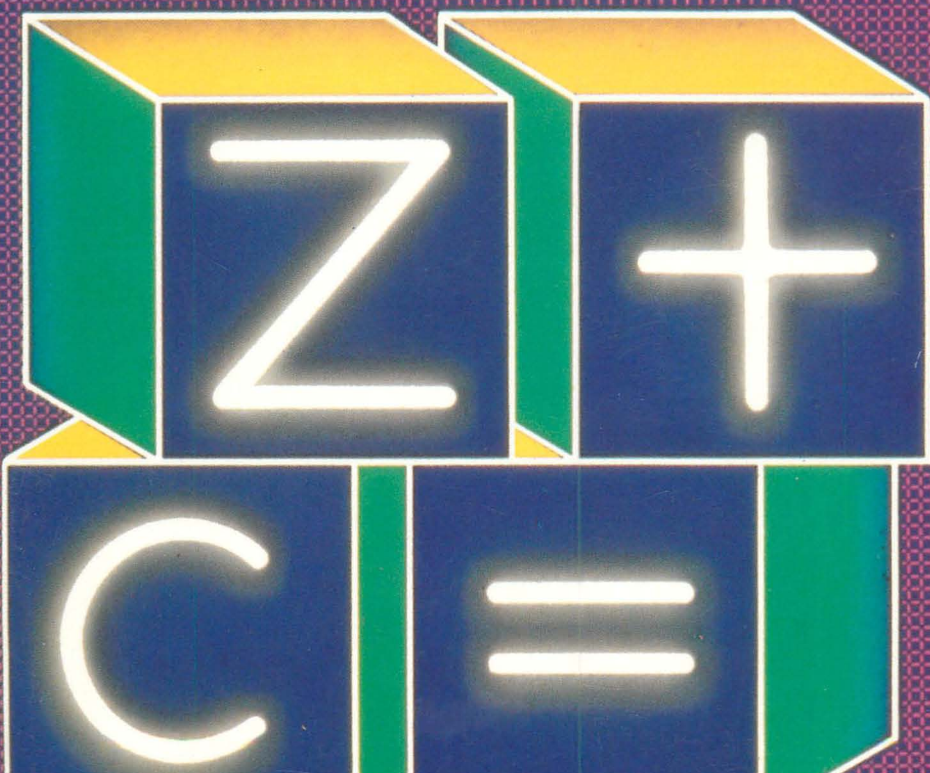
CENTURY
COMMUNICATIONS

ROBERT CARTER

MATHS TUTOR FOR THE COMMODORE 64



MATHS TUTOR

FOR THE COMMODORE 64

MATHS TUTOR
FOR THE COMMODORE 64

Robert Carter

CENTURY COMMUNICATIONS
LONDON

First published in Great Britain in 1984
by Century Communications Ltd,
12–13 Greek Street,
London W1V 5LE

ISBN 0 7126 0612 2

This book is also available in a software pack
containing a cassette of the programs.

Typeset by Avocet Marketing Services, Aylesbury
Reproduced, printed and bound in Great Britain by
Hazell Watson & Viney Limited,
Member of the BPCC Group,
Aylesbury, Bucks

CONTENTS

Preface		ix
1	Introduction to the computer	1
2	Numbers	10
	The Real line	
	Variables and Constants	
	Equations	
3	Operators	26
	Powers	
	Roots	
4	Inequalities	44
	Functions	
	Graphs	
5	Number system	62
	Logarithms	
	Indices	
6	Angles	74
	Sin, Cos and Tan	
	Inverse trig functions	
	The radian	
	Pythagoras' theorem	
	Cofunctions	
7	Polynomials	95
	Quadratic equations	
	Conics	

8	Polar coordinates Curve sketching Solution of triangles Trig equations	118
9	Complex numbers Argand diagram Hyperbolic functions Inverse hyperbolics	137
10	The factorial Series Progressions Convergence Bionomial expansion Fibonacci	159
11	Vectors Matrices Determinants	181
12	Differentiation Taylor & Maclaurin Differential equations	208
13	Integration	223
14	The Random Factor Permutations and combinations Probability Distributions	235

Appendices

1	User-defined graphics	250
2	A series for PI	253
3	Geometric shapes	255

To David and Susan and the Lump

PREFACE

This book is not an examination syllabus text, but will certainly be of use to people who find themselves dealing with O and A level maths, with TEC subjects, and those in their first year's study at University and Polytechnic in science and engineering.

I would hope also that a lumen or two of light might be shed on mathematics for the casual reader and micro-user by the book, and that those for whom the whole domain of mathematics is a mystery might be tempted to explore further.

Since it is customary to thank those who helped in the preparation of a book at this point, may I tender my gratitude to the following: Mr Steve Crocker, Mrs June Hall, Miss P. C. Gibson, and the gentlemen members of 2B Lime Grove.

Rob Carter
Shepherd's Bush
February 1984

Important Note

Wherever in a program listing the instruction PRINT"" appears, you should use the reverse heart shape control code. See page 9 for instructions.

1

INTRODUCTION TO THE COMPUTER

(This section is for people who are unfamiliar with Commodore BASIC. If you've never written a program before, then you will find this introduction necessary.)

First, plug in your computer and tune in your TV (the manual will give you details) so that you get a blank screen with writing at the top like this:

```
     **** COMMODORE 64 BASIC V2 ****
64K RAM SYSTEM 38911 BASIC BYTES FREE
READY.
```

Now, anything to be spelled out on the keyboard will appear on the screen.

Next, you must learn how to make the computer perform tasks. There are basically two things it will do: calculate, and display (for example, the result of a calculation). If you want it to calculate and then display the answer, you have to tell it to do both. The CBM 64 will handle most of the common mathematical calculations (you can guess this from looking at the manual, where you will find mathematical terms such as SIN, COS and TAN, and symbols like + and −).

The slash (/) is used to show division (as when you write fractions such as 2/3 or 1/2), and the star symbol (*) means 'multiply'. Computers don't use a small x to show multiplication because this can be confused with the letter of the alphabet, and computers have to be very precise. For the same reason, a zero is shown with a slash through it, so it is not confused with a capital O. Displaying answers is achieved by the instruction PRINT. So without more delay let's make the computer do a calculation.

Spell out the word PRINT - but make sure you spell it correctly or the computer will not understand! Use the shift key, as you would on a typewriter, to select the upper case letters. You'll see that the cursor now sits after the word you've just written.

Now, if you want the computer to PRINT something, you must further instruct it what to PRINT; and you must enclose whatever you want printed in inverted commas (or, as they're sometimes called, speech marks) (").

Suppose we want to get the computer to write:

I AM THE COMMODORE 64!

Then we have to assemble this instruction:

PRINT "I AM THE COMMODORE 64!"

The following keystrokes are necessary:

PRINT (You should already have this)
" - (SHIFT and 2 key)
I -
(space) - (You have to put this space yourself by pressing the SPACE bar - bottom of keyboard)

Then you spell out the rest of the sentence in the same way. Don't get mad if you make a mistake - keep cool and remember that you can erase anything by placing the cursor and pressing the INST DEL key. You can move the cursor left or right in the line you're assembling by using the CRSR keys (the arrows on the keys show you which way is which; use the SHIFT where necessary).

Now that we've assembled the command we can give it to the Commodore, and we do this by pressing the RETURN key (far right of the keyboard). This sends the command into the computer's electronic circuits to be processed. As soon as RETURN is pressed, the computer examines the command and sees the word PRINT. It knows it must print something, and when it sees the inverted commas (") it looks for the other set of inverted commas and knows that it has to print whatever it finds between them.

Press RETURN and see the effect. You will now have your message printed up on the screen and below it the message:

READY.

which just means 'OK, I've done what you asked, I'm ready for the next task!'

The only time you don't need a pair of inverted commas round the message you're printing is when it's a number (or a letter that you've told the computer is going to represent a number. But don't worry about that now; we'll come to it later).

Try this:

PRINT 3+5

(You will find the + sign on the top row.) When you've got the line ready, press RETURN, and the line will enter the computer's electronics to be executed (which means 'carried out', as in 'executing a command'). You should get

8

printed up on the screen, because 3+5=8: try this out:

PRINT"3+5"

See the difference?

So you can now instruct the computer to do small calculations and print up the answer. Now let's go on to the next step.

We already know about printing using PRINT. But try this:

PRINT TAB (25);"A"

This lets you put your message where you want it, and to tell your Commodore to do that you have to know the column number. The Commodore's screen has 25 lines in which to write, and 4Ø spaces in each line. So you have a screen with 1ØØØ (25 times 4Ø) squares where a letter or other symbol can be printed.

The lines are numbered from Ø to 24, and the columns are numbered from Ø to 39. When you write TAB, you are telling the Commodore that you are going to give it a column number, and the computer expects you to enclose the number in a pair of brackets (N) so it doesn't get confused. Furthermore, once you've given it the number you must use a semi-colon (;) before telling it what to print. So, the finished instruction looks like this:

```
PRINT TAB (5);"A"
```

which means: 'Print a letter A in the 6th column' (NB: the first column has number Ø so the sixth column has number 5.)

Now unlike people, who might notice errors but still be able to follow instructions, the computer is unforgivingly rigid about the form in which it expects instructions, and any departure from this will cause it to give up and write an error message to you. Also, the Commodore's no fool, and if you tell it to print a character in a column number greater than 39 then you will get your character printed in a row further down. Try this:

```
PRINT TAB (50);"X"
```

All this sounds complicated at first, but it's surprising how quickly you get used to it, and after a couple of hours of practice you'll be able to do anything this book asks of you.

Colours can be produced using the key marked CTRL (on the far left of the keyboard) along with the number keys. The keys 1 to 8 each call up a colour, and keys 9 and Ø switch the foreground and background colours into reverse and back again. (Using CTRL and the space bar means you can get blocks of colour.) CTRL stands for CONTROL, and there are other things connected with it which we will look at later. The control functions help to get the desired effects onto the screen.

The Commodore key is the one with the funny-looking symbol on it down at bottom left. This key is like a SHIFT key and you use it to get the graphic symbols on the sides of the keys. Hold down the SHIFT and Commodore key and you can get lower case letters, and the Commodore key alone allows you to get at the graphic symbol on the left side of a key. (SHIFT alone will give you the right hand graphic symbol.)

Now we must discover what a program is. We have made the computer carry out instructions in the form of commands, but a program is a *series* of commands. What is needed is a way of making the computer wait until a complete sequence of instructions is composed, and then execute them one at a time. If the commands are numbered before we write them, then the computer will store them in order, and when the command RUN is given, it will do them in sequence. In this way a complicated sequence of instructions called a PROGRAM is created.

When we enter the command RUN, we set the computer to examining the numbered program lines IN THE ORDER IN WHICH THEY ARE NUMBERED, and as it examines each instruction it decodes them and executes them.

So let's write a program!

Program 1: (press the RETURN key after each line.)

```
1 PRINT "I"
2 PRINT "HAVE"
3 PRINT "JUST"
4 PRINT "WRITTEN"
5 PRINT "A PROGRAM."
```

To execute, type RUN and press RETURN.

By RETURNing after each line, the program is assembled on the screen so that you can easily see your sequence of instructions.

Notice that each new PRINT instruction starts out on a new line (there was no placing instruction in your program to go with it), and how fast the Commodore runs through the sequence.

You may notice a potential problem with using the numbers 1,2,3,4, etc. What happens if you want to slip in a line between 2 and 3? You can't – the computer won't recognise fractional line numbers. Consequently lines are generally numbered in steps of 1Ø. Try this program:

Program 2

```
1Ø POKE 5328Ø,Ø
2Ø POKE 5328Ø,1
3Ø POKE 5328Ø,2
4Ø POKE 5328Ø,3
5Ø POKE 5328Ø,4
6Ø POKE 5328Ø,5
```

```
70 POKE 53280,6
80 POKE 53280,7
```

then type RUN and press RETURN.

Now try this variation:

Program 3

```
10 FOR N = 0 TO 7
20 POKE 53280,N
30 NEXT N
```

This gives exactly the same effect, but with less typing. To slow down the program execution in order to see what's going on, add these lines to program 3. Don't worry about the order of the lines, the computer will automatically put them in your program in the correct place.

```
25 GOSUB 50
40 END
50 FOR X = 1 TO 100: NEXT X
60 RETURN (Spell this out in letters - it doesn't mean hit
the Return key!)
```

Now type LIST (and return).

While the program was assembled each numbered line was sandwiched into the program at exactly the correct place. When RUN the program is slowed down by using two loops. The FOR - NEXT loop is a common 'control structure' in programming; it sends the computer round the loop a set number of times, so it will do some instructions several times over. The first FOR loop is from line 10 to line 30 and the instructions repeated are POKE 53280,N (which changes the colour of the border) and GOSUB 50 (which makes the program jump to line 50). The second FOR loop is in line 50 and has no instructions inside it so that it just goes round 100 times using up time. In this way you have taken enough time to see the colour the border has been set to.

The other control structure used here is the GOSUB - RETURN, which jumps the computer to the line number mentioned, and when it reaches the RETURN instruction jumps back to where it came from.

Now try this one: (Incidentally, when you enter a program on the CBM 64 you can leave out the spaces if you like – but it's better to get into the habit of writing your programs so they can be read easily.)

Program 4

```
1Ø FOR N = Ø TO 15
2Ø POKE 5328Ø,N
3Ø POKE 53281, N
4Ø FOR X = 1 TO 5ØØ NEXT X
5Ø NEXT N
```

This loop does much the same thing, except that the X FOR loop is left inside the N FOR loop (and there is no GOSUB – RETURN to jump in and out). But the other instructions here are new ones. The effect of running the program is to change the border and background colours. Since we already know that POKE 5328Ø,N is changing the border, POKE 53281,N is clearly changing the background colour. In fact, POKE puts a number (the one written after the comma) into a memory location (or address), so that

```
POKE 53281,5
```

'pokes' the value 5 into address 53281. These memory addresses are recognised by the computer because it sets some of them aside for special tasks and (so it seems) stores the border colour in location 5328Ø and background colour in location 53281.

The colours have codes, and those codes are as follows:

Ø BLACK	6 BLUE	12 GREY 2
1 WHITE	7 YELLOW	13 LIGHT GREEN
2 RED	8 ORANGE	14 LIGHT BLUE
3 CYAN	9 BROWN	15 GREY 3
4 PURPLE	1Ø LIGHT RED	
5 GREEN	11 GREY 1	

Furthermore, the CBM 64 screen itself is 'pokeable' because it's made up of 1,ØØØ character squares and each has its own memory location. These screen addresses start at 55296 (top left corner) and go to 56295 (bottom right corner), and are in the same sequence as lines of writing (i.e. from left to right, and line-by-line from the top). POKE any of these addresses with a colour code and the relevant square turns the correct colour, provided something's printed in it. Try program 5 to see how it works:

Program 5
```
10 FOR N = 55296 TO 56295
20 POKE N,255*RND(15)
30 NEXT N
```

In line 20 there is a function RND which selects a random number to poke into each screen address, thus ensuring that the screen becomes colourful.

An important function of the computer is the drawing of graphs. Unlike some computers the CBM 64 doesn't have PLOT and DRAW functions already in its BASIC, so a small program has to be prepared that allows us to plot dots and draw lines using the screen like graph paper. A small section of a program created to do a certain job is often referred to as a 'routine' (a small and therefore presumably minor routine is known as a 'subroutine'), so here's a routine that allows us to PLOT a spot:

Program 6
```
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Don't try to run this program on its own, but watch out for it in the programs that are to be found in this book. It will turn up whenever we want to draw a line, to help make up graphs and diagrams that illustrate the mathematics.

We can draw circles too; here is a program for that:

Program 7
```
2000 PI = 3.1415927
2010 FOR ANG = 0 TO 2*PI/35
```

```
2Ø2Ø X=INT(X1+1.1*R1*COS(ANG)+.5):
     Y=INT(Y1+R1*SIN(ANG)+.5)
2Ø3Ø GOSUB 1ØØØ
2Ø4Ø NEXT ANG
2Ø5Ø RETURN
```

Don't worry too much about how these routines work; just think of them as being a set of instructions tucked away in the program that allows the effect to be achieved, and something to call on when you want to build up screen graphs. The screen is low resolution - reasonably coarse - so that the lines and circles obtained are by no means smooth, but they serve the purpose of illustrating the mathematical principles.

One further point concerns a graphic that occurs frequently in our programs. It is a reversed heart-shape and unless you understand it it might cause you some puzzlement. The symbol is obtained by holding down the SHIFT and pressing down the CLR HOME key at top right, and if you enter it as a command you get a clear screen. To incorporate it into a program (as with other control graphics) you should hide it into a program (as with other control graphics) you should hide it in a PRINT command. Try it out like this:

Program 8

```
1Ø PRINT "♥"
2Ø FOR N = Ø TO 2Ø
3Ø PRINT "X"
4Ø NEXT N
5Ø FOR X = Ø TO 1ØØØ: NEXT X
6Ø PRINT "♥"
```

A little practice with these basic programs will make the machine's controls clear to you. Good luck with the rest of the book - and don't be afraid to experiment!

2

NUMBERS

1 The Real line

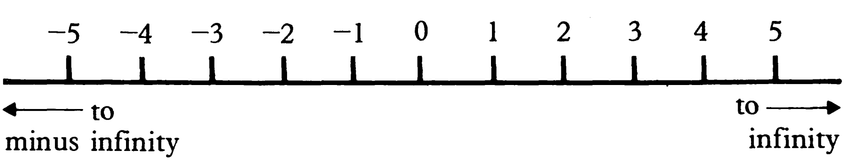
Since computers ultimately deal with numbers, we should begin with the number system.

Normally, we take numbers for granted, assuming that they have just been there all the time. But there *is* a number system, and it's beautifully logical. It starts with nothing, or rather, with *zero*.

There *is* a difference between zero and nothing. Zero is one less than one, whereas nothing is just … nothing. To demonstrate: this line is called the Real line:

Actually, this line is only a bit of the Real line. The whole thing wouldn't fit on this page, or indeed on any page. The Real line is infinitely long (from infinity to minus infinity), but we shall only bother with the middle bit of it.

At the absolute centre is zero, and at regular intervals to the right are the numbers one, two, three, … and so on; and to the left are minus one, minus two, minus three and so on, thus:



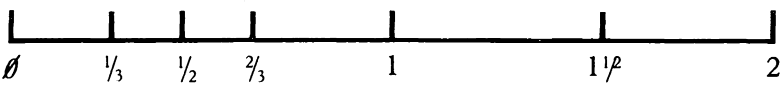
You can do adding and subtracting calculations on the Real line. *Step to the right for every one you want to add and one to the* *left for every one you want to subtract.* Take the sum 2+3. The answer is 5. Let the Real line prove it. Count from 2 three spaces to the right to add.

But what about 3–8? There are those who say it can't be done, but we know that by using the Real line we can get at the truth: 3–8=–5.

This is why the Real line needs to be infinitely long: numbers can be as large positive or negative as you like. And you can see exactly what is meant by zero being just one less than one.

Let's look at an even smaller stretch of the Real line, from zero to two. Now, both one and two are whole numbers, and are called INTEGERS. An integer is any whole number, and there are positive integers and negative integers, for example: 1, 2, –5, 2ØØØ, –37, 67,831 and an infinite number of others. Integers are strung out along the Real line like pearls on a necklace. But whole numbers, of course, are not the whole story.

Let's look at the Real line between zero and two:



Half way between Ø and 1 there's ½, and half way between 1 and 2 there's 1½. Also, the ⅓ and ⅔ positions are marked so you can see where they live.

You probably have a wooden model of the positive side of the Real line over several integers: a ruler. Some rulers have all sorts of subdivisions: quarters, tenths, eighths, twelfths, sixteenths, and you could magnify a portion and magnify that portion again, and again and so on until you were looking at hundredths, or thousandths or millionths. All these fractions are important to us, but how can we handle fractions on the Commodore?

There's no fraction key, no ½ or ¾, so when we want to talk fractions we have to convert them into *decimals*.

So that, ½ equals Ø.5
¼ equals Ø.25
¾ equals Ø.75

There's a calculator facility built into the computer so that you can instantly transform any fraction into a decimal with a single command. Just do:

PRINT 3/4   ENTER

and the computer will give you back .75 printed neatly on the screen. Try it out and prove that:

3/8 = .375
17/34 = .5
7/5 = 1.4

Or you could get adventurous, with 355/113=3.14159292.

Jargon in fractions includes calling them *vulgar* if the number on top is bigger than the number underneath, because that makes them bigger than one, like 7/5 or 1.4). The top number is the *numerator* and the bottom number is the *denominator*.

Some fractions can't be turned into decimals so easily. Try it with the *third*. Simple task for a computer like the CBM 64 with all the computing power it has you might suppose, dividing one by three. But no computer can do it, because the result is an endless number.

Numbers can be exact or unending, and exact ones end after a few decimal places, like 1.23456789 or 5.75 or 0.383838. Once you have them, you know all about them.

With unending ones you never know all about them because you can never write them down completely. The string of decimal places *never ends*. No computer could ever write ⅓ out in decimal form because it has an infinite number of decimal places. It goes something like this:

Ø.33333333333333333333333333333333...

and then one gets bored and stops. If you run it on the CBM 64 you'll get Ø.333333333 and then *it* gets bored.

Another unending number is ⅔; if you divide 2 by 3 the repeating decimal is 6. If you command your CBM 64 to

PRINT 2/3

it will give you back:

Ø.666666667

The ninth place is rounded off in the interest of accuracy. The CMB 64 will display numbers to nine decimal places. (That means it will list up to nine numbers after the decimal point. If we had a number as the result of some calculation that came to

Ø.1234567889 and the computer only displays to nine places, then we might read on its screen Ø.123456788. This is pretty close, but it is even closer if it displays the number Ø.123456789, which is only Ø.ØØØØØØØØØ 1 out from the real number instead of Ø.ØØØØØØØØØ 9 out. If you have a number in the first *non-displayed* decimal place greater than five, it is clearly more accurate to bump the last displayed decimal place up by one. And *that* is why 2/3=Ø.666666667 on the Commodore.

Equally, just cutting the first non-displayed number off and throwing it away if it is less than 5 (or *equal to* five), is getting the best accuracy. That's why 1/3 is shown as Ø.333333333. Cutting the last figure off is called *truncating*.

You may ask, why do we call it the Real line. The answer is that it's a line full of real numbers. There are *unreal* numbers too, but we will leave those until later.

2 Variables and Constants

If you try this:

PRINT 3/11

and you will get Ø.272727273. Here the repeat is a *pair* of numbers. If you try 22/7, you'll get 3.14285714. There are only seven decimal places in that one, and it's a good approximation to perhaps the most famous number of them all: PI.

Pronounced pie as in steak 'n' kidney, it's a Greek letter written π. On the Commodore, it's a SHIFTed ↑ (next to the RESTORE key) and the reason Commodore's engineers have gone to the trouble of putting it on a key is that it comes into lots of different areas of maths, and it saves the tedious keying-in of the number everytime you want to use it.

It is, in fact, irrational (meaning it can't be written as a fraction) and if you do

PRINT π

you will get 3.14159265. So the approximation 22/7 is not bad.

The approximation 22/7 was discovered by the Greek mathematician Archimedes (287–212BC). A 35 figure approximation to PI is 3.14159265358979323846264338327950288. Can you see where the Commodore rounds it off? PI is an example of an irrational number that goes on and on without any repetitions at all.

Earlier we suggested that if you were really adventurous you might like to try 355/113. Look back and see what it comes to. It is a very good approximation to PI just by dividing two numbers. It didn't escape the attention of Tsu Ch'ung Chi (430–501), a Chinese engineer who found it too. If he'd had a computer, he could have got it much quicker by using program 9.

Program 9 looks for an approximation by dividing integers by other integers; it never lets an integer go above a thousand, and it looks for PI to three decimal places. Try it and see if it will find both Archimedes' and Tsu's approximations.

Program 9 PI APPROXIMATIONS
(Takes 10 minutes so wait for it!)

```
1 REM PROGRAM 9
2 REM PI APPROXIMATIONS
10 PRINT "PROGRAM IS RUNNING-PLEASE WAIT
"
15 LET PI=3.1415927
20 FOR N=4 TO 1000
30 FOR D=INT(N/4) TO N/3
40 LET P=N/D
50 IF P>PI-0.0001 AND P<PI+0.0001 THEN P
RINT N;"    ";D,N/D :END
60 NEXT D
70 NEXT N
```

You might have guessed that N stood for numerator and D for denominator, and it would have been possible to have both N and D run from 1 to 1ØØØ. But that would have taken too long, so line 2Ø has some means of keeping the running time short by excluding obviously wildly wide-of-the-mark examples. We know that d is going to be between a quarter and a third of n. Line 5Ø selects values within the required accuracy range, i.e. between PI −Ø.ØØØ1 and PI +Ø.ØØØ1.

Program 9 is rather daft from a mathematical point of view because it relies upon knowing PI to find approximations, but it does serve to show just how clever old Tsu must have been. We have shown in a program taking about 1Ø minutes to run that his approximation is the *best* that could be got using two integers less than a thousand!

This is all very well, but what *is* PI? We know it's value pretty well, and lots of its history, but what's all the fuss about? To begin with, it is a *fundamental constant*. It's so important in maths it gets into almost everything, and a couple of good examples are circles. The reason the ancient Greeks got to know about PI is because they were architects, and that meant they had to be interested in geometry.

If a piece of string is laid in a circle and the circle is one yard across, then the piece of string is PI yards long. In other words, the diameter of any circle is related to the distance round the circle (the circumference) by the constant PI.

PI is called a constant because its value never changes. There are many constants in maths and the sciences: the speed of light, the constant that converts inches to centimetres, an interesting device called 'e' that you'll be discovering later, lots of them. But PI is probably the king of them all.

Consider also: if you have a circular table top and the distance from the centre to the edge is one yard, then the area of the table top will be PI square yards. The distance from centre to edge (radius) of a circle is related to the area by the constant PI. We can write equations for both these results:

$C=\pi\star d$ (where C is the circumference and d is the diameter)

$A=\pi\star r\star r$ (where A is the area, and r is the radius)

If you draw an accurate circle and a diameter and radius on it, the diameter is twice as long as the radius, or in equations: $d=2\star r$

Let's put the computer to work on circles.

Program 10 CIRCLES & PI

```
2 REM CIRCLES & PI
10 PRINT""
20 INPUT"INPUT RADIUS IN CM. (2 TO 10) "
;RADIUS
22 REM DRAW RADIUS ARM
```

```
25 COL=1:CHAR=62
30 FOR X=25 TO 25+RADIUS:Y=12:GOSUB 1000
:NEXT X
40 PRINT"RADIUS=";RADIUS;"CM"
50 REM DRAW CIRCLE
60 X1=25:Y1=12:R1=RADIUS:COL=2:CHAR=81
70 GOSUB 2000
80 PRINT"CIRCUMFERENCE=";2*RADIUS*PI
998 END
999 REM PLOT POINT SUBROUTINE
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
1999 REM CIRCLE PLOTTING SUBROUTINE
2000 PI=3.1415927
2010 FOR ANG=0 TO 2*PI STEP PI/(R1*3.5)
2020 X=INT(X1+1.1*R1*COS(ANG)+.5):Y=INT(
Y1+R1*SIN(ANG)+.5)
2030 GOSUB 1000
2040 NEXT ANG
2050 RETURN
```

When run, the program stops asking for a radius length in centimetres, instead drawing a scale model of the circle (which is a bit disjointed because of the screen blocks) and giving you its vital statistics.

If you want to use radius lengths greater than 1Ø it will not all fit, but if you just want to work out the vital statistics alone try program 10a:

Program 10a CIRCLE STATISTICS

```
1 REM PROGRAM 10A
2 REM CIRCLE STATISTICS
10 PRINT ""
20 INPUT "RADIUS ";RADIUS
25 PI=3.1415927
```

```
30 PRINT
40 PRINT "RADIUS=";RADIUS
50 PRINT "DIAMETER=";RADIUS*2;" UNITS"
60 PRINT "CIRCUMFERENCE=";PI*RADIUS*2; "
UNITS"
70 PRINT "AREA IS ";PI*RADIUS*RADIUS
```

The beauty of this is that you can get radius lengths of any size and any units (inches, centimetres, miles, chains, rods, poles... anything you like).

So a constant is a number that doesn't change, and a variable is one that does. In an equation like C=PI*d, PI is a constant – you know its value and that value never changes no matter how you use it – and you have other quantities (d and C) which are variables. So if you're talking about a big circle both C and d will be bigger than the C and d in a small circle. The PI however is always the same.

In the computer, variables are specified using a letter, or a word made of letters and numbers, e.g. R, C2, Diameter, Area 7. This is partly because having a name for a variable gives you a clue to what it's a variable of: if a variable for the radius of a circle is called R, the program will be easier to read than if we called it C or f.

In ordinary, non-computing maths, variables are usually just a single letter; i.e. the three sides of a triangle might be called a,b and c. When ordinary letters run out Greek letters are often used, especially for angles. *Theta* is a favourite. It looks like a circle with a bar in it: θ.

Constants are often found in exercise books, converting one set of units into another, like the English units into metric:

1 metre = 3.28 ft
1 litre = 33.8 fluid ounces
1 mile = 1.609 Kilometres

Those numbers are constants. They relate Imperial measure to metric measure and vice-versa, so that if you ask how many feet in 1Ø metres the equation would be:

1Ø metres =3.28*1Ø ft
= 32.8 ft

The 3.28 stays in there all through until you do the multiplication.

If the computer did not have a PI key, that would not stop you using PI to work out the areas of circles and so on. It would mean that you'd have to find the value of PI and then you could *define* it to the CBM 64 with a LET instruction. (See line 25 of program 10a, for example.)

LET instructions help you to get the information into the computer, and sometimes they tell the computer to perform a calculation. (In Commodore BASIC you don't actually have to write LET; the equation is enough for it to understand what you mean.)

All along we've been using the computer's built-in calculating facility. We just tell it to PRINT 5+4 and it returns 9. LET does the same thing only *invisibly*.

If you were to write:

```
1Ø LET a=5
2Ø LET b=4
3Ø LET c=a+b
4Ø PRINT c
```

(Remember, LET is optional; why not try this program without it?)

it would give you 9. But the computer only prints it because of line 4Ø. So what if you delete line 4Ø and run it again? It still does something. It sets a variable called a equal to five, it sets a variable called b equal to four and it adds the values of a and b and sets c equal to the sum, 9. In other words, the computer has been thinking about it all without writing anything.

And almost all BASIC computing is like that: there are instructions that make the computer *think* and other instructions that make it *show* you what it has thought. In this book, we're most concerned with the thinking part, because as programs 1Ø and 1Øa showed, if you want pretty displays you've got to work longer on your program. Program 1Ø gives red circles, but 1Øa is far more powerful as a calculating aid.

3 Equations

Algebra is one of history's most powerful ideas. It was algebra that built the pyramids and St Paul's Cathedral; it was algebra that put mankind on the moon.

The dictionary says that the word 'algebra' comes from the Arabic words *al*, meaning *the*, and *jabr*, meaning *to fix* (more or less). So algebra is *the fixer*.

Most people have a hazy notion that algebra is arithmetic with letters instead of numbers, but that idea is misleading. An algebraic equation is like a set of weighing scales: you put something in the right hand pan and some weights in the left hand pan, and when the two weights are equal you have your answer, and between them is an equal sign, meaning that what is on the left equals what is on the right. Algebra is concerned with tinkering around with equations to get a useful result, and has rules that allow you to alter equations to suit yourself.

Earlier, we had a couple of equations for the circle. Remember $C=\pi^\star d$ relates the distance round a circle to the distance across it. We could write:

Circumference=$\pi^\star$diameter

but it's better shorthand to write:

$C=\pi^\star d$

We know the value of PI – it's 3.14159 etc. ... we only need to know the diameter and we can calculate the circumference.

If we can find the right hand total, we can say that *since it is an equation the left hand side must equal the right hand side*, and we have our answer.

We try to get equations into the form whereby the thing we want to find or calculate is on its own on the left hand side, and all the other constants and variables are on the right hand side. But what if, in the example above, we knew the circumference and wanted to calculate the diameter? Well that is why we have to be able to *manipulate* equations. So let's have a go at manipulating $C=\pi^\star d$.

We want to know the diameter d, and we already know the circumference, C. PI, of course, is a constant and the computer remembers its value even if we don't. So let's start off with the understanding that we need to get d on its own on the left hand side. (We'll abbreviate right hand side to RHS and left hand side to LHS.)

Several Rules of Algebraic Manipulation

RULE 1 You can make a left hand side into a right hand side and vice-versa. That is, you can swap sides.

(so we can make $C=\pi*d$ into $\pi*d=C$, just swapping RHS and LHS.)

RULE 2 You can do anything to one side so long as you do it to the other side as well.

This means that you can *add* anything to both sides
subtract anything from both sides
multiply both sides by anything
divide both sides by anything.

BUT TAKE CARE – *if you do it to one side, you must do exactly the same to the other side*. And you've got to look carefully at the equation, bearing in mind what your aim is, and choose a manipulation that helps you achieve your aim.

In the example you have now got $\pi*d=C$, so that the aim of getting d alone on the left is only one step away. PI times d on the left could be made into d alone by dividing the LHS by PI. (Think about that carefully. $\pi*d/\pi$ is really the same as d on its own, because it's taking d and multiplying it by PI and then dividing it by PI, and the two operations cancel one another out.) But you must do the same to the RHS too. Giving:

$$\pi*d/\pi=C/\pi$$

and because the LHS is really the same as d alone, you can write:

$$d=C/\pi$$

And d is now alone on the LHS.

Now let's step back and have a look at exactly what has been achieved. What does $d=C/\pi$ *mean*?

It contains the same information as the equation you started out with: the same constant PI, and the same two variables d and C. It's just been manipulated so that if you know the circumference you can calculate the diameter.

So let's try it out. Back to the Commodore:

Program 11 MASSAGE

```
1 REM PROGRAM 11
2 REM MASSAGE
10 PRINT ""
20 INPUT "ENTER DIAMETER ";DIAM
30 PI=3.1415927
40 LET CIRC=PI*DIAM
50 PRINT
60 PRINT"CIRCUMFERENCE IS ";CIRC
70 PRINT
80 INPUT"ENTER CIRCUMFERENCE ";CIRC
90 LET DIAM=CIRC/PI
100 PRINT
110 PRINT "DIAMETER IS ";DIAM
120 END
```

Line 2Ø asks you to choose any diameter. Line 4Ø uses the original equation to calculate the circumference, and line 6Ø prints it out. Then line 8Ø asks for a circumference, and you should read the one just calculated and whose value is printed at the top of the screen. If you enter that one, line 9Ø will do the reverse of what line 4Ø did, and you should get back to the diameter you started with. That way we prove that line 9Ø equation (the one we manipulated) is really doing what we want it to do.

Run it a few times, but remember the diameter you used and compare it with the diameter it gives you back.

If you do it often enough you will find that sometimes the CBM 64 gives you back a diameter which is not *exactly* equal to the one you started with. Can you see why this should be? It's not that our equation is a bit inaccurate, it's that the Commodore can't use the actual value of PI, but only the one rounded off to nine decimal places. If you look at the error involved, it is very small.

Before we leave the subject we really ought to have a look at a few more of these manipulations. Stick to rules one and two. Remember that adding and subtracting are opposites, and multiplying and dividing are opposites. Remember also that any number divided by itself is 1 (π/π for example is 1, and you can try it with other numbers) and any number minus itself is zero ($\pi-\pi=\emptyset$)

Try a few of these with a pencil and paper: If x=a+b+c, get b as the subject of the equation (*Subject* just means the one on its own on the LHS.) This is it in full:

x is the subject of the equation at present; let's swap sides:
a+b+c=x
If we now subtract a from both sides we get,
a−a+b+c=x−a
and a−a is zero, so we have,
b+c=x−a
If we now do the same subtracting b from both sides,
b+c−c=x−a−c
and since c−c=Ø, we have,
b=x−a−c

And that's our result.

In order to get into the language of maths, it is a good idea to *read* equations when you see them. Pronounce them to yourself just like you might read a line of poetry to yourself. If you, for example, read:

$$x=\frac{(2*a)+((5*a*b)/c)}{(c-d)}$$

"x equals two times a...
plus...
five times a times b over c...
all over c minus d"

Notice that the equation is read in several lines in order to break down the parts of the right hand side so that it means something unique. The real equation itself is broken down that way too, by using brackets. The brackets are like little parcels of numbers and variables that must each be worked out as a group. This makes the equation a lot easier to work out. If we represent the first bracket in the above equation by a letter A, the second one by B and the third by C, it would simplify matters quite a lot:

If A=(2*a)
and B=((5*a*b))/c)
and C=(c−d)

then we could substitute the letters for the brackets and write:

$$x=\frac{A+B}{C}$$

We'll be coming back to brackets, because it is really important to get the brackets entered correctly. Let's prove it. Enter this in your machine:

```
1Ø LET a=2:LET b=3:LET c=4
2Ø PRINT a*b+c
```

You ought to get back 1Ø. Then add,

```
3Ø PRINT (a*b)+c
4Ø PRINT a*(b+c)
```

And you'll find line 3Ø gives you 1Ø as before, but line 4Ø gives you back 14.

Same variables, same order, same multiply and add signs, different result. If you *evaluate* (work out) the two *expressions* (combination of variables) you will see why. The one in line 3Ø is (two times three) that is six...plus four...which is *ten*. The line 4Ø expression is two times (three plus four), or two times seven, which is *fourteen*.

Line 2Ø gave 1Ø as a result instead of 14 because the computer has an inbuilt order for working out expressions. It will evaluate the multiplys and divides *before* it evaluates the adds and subtracts. Unless of course you tell it you want it to work out an expression in a certain way by using brackets as we did in line 4Ø). This idea of some operations being carried out before others is called *priority*. We will come back to that.

For the time being however, we must look at FOR loops again, because they *automate* program operations, making programs easier to use.

There are four instructions in BASIC used with FOR loops: FOR, TO, STEP and NEXT.

The purpose of a FOR loop is to stop the program going in one line from top to bottom and make it loop back over itself several times. This is quite simple to illustrate on the machine itself, and here's a simple example:

Program 12 SIMPLE FOR LOOP

```
1 REM PROG 12
2 REM SIMPLE LOOP
10 FOR N=1 TO 10
20 PRINT "LOOP NO.";N
30 NEXT N
```

Run it and see what happens. Line 10 tells you how many times to go round the loop. You can change the values that the loop variable, n, starts and ends at. (Experiment with different numbers other than 1 and 10.)

Line 20 is the part of the program that is being looped through. It happens to contain a PRINT instruction to help you see what's happening, but it doesn't have to. Line 30 is the NEXT instruction, and serves to send the program-watching computer back to line 10 where the FOUR instruction is to be found, and make n take its next value (in this case two). But what if you'd wanted to count the number down from ten to one, instead of from one to ten?

You just use STEP. The number associated with STEP tells the computer how many to add to or subtract from n each time it goes around the loop. So that if you make line 10 of our program 12 into:

```
10 FOR n=10 TO 1 STEP -1
```

and you will find the numbers do indeed run from 10 to 1.

You should experiment now, to familiarise yourself with the function.

Use different values of STEP variable. Get a feel for how the FOR loop works. Try using small decimals like 1.5, or expressions like 2*5/3.

Notice that for a FOR loop to work, the number before the TO must be smaller than that after the TO when the STEP variable is positive, and it must be bigger number first if the STEP variable is negative.

It will be a lot easier using FOR loops to take the drudgery out of some of the more complex equations we're going to get into later.

Now for *nesting*. With this technique you can fit FOR loops one inside another, like this:

Program 13   RANK AND FILE

```
1 REM PROGRAM 13
2 REM RANK AND FILE
9 PRINT ""
10 CHAR=88:COL=7
20 FOR X=0 TO 39 STEP 2
30 FOR Y=0 TO 24 STEP 2
40 GOSUB 1000
50 NEXT Y
60 NEXT X
70 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Notice that the y loop is *entirely* inside the x loop. If you swapped lines 5ø and 6ø into:

5ø NEXT x
6ø NEXT y

it wouldn't work. Experiment and prove it to yourself.

Now go back to Program 13 in its original form, run it and count the *club* graphics made in each vertical line. That tells you how many times it goes round each y loop for each x loop. If you count the number of dots horizontally you will find the number of times it passes round the x loop. And it does it all in five seconds, which is much faster than you could specify each plot yourself.

3

OPERATORS

1 Operators

Maths is based on the number system and four basic operators, and everything comes down to them in the end.

The famous four are:

Add +
Subtract −
Multiply * (or × if you are not into computers)
Divide / (or ÷ sometimes)

We have seen that 'add' and 'subtract' are opposites. (This must be so because if you take any number and add any other number to it, then subtract the same number, you get back to the first number.)

The same is true of the other two operators. There are two *pairs* of operators then, addition/subtraction, and multiplication/division.

The relationship between add and subtract is that one is the *inverse* of the other – meaning that if you do one then the other you get back to where you started. 'Multiply' and 'divide' are also inverses of one another. But are there other relationships?

Multiplication is just a way of adding lots of times repeatedly. We could say five fours are twenty, or five times four equals twenty, or five+five+five+five; that is, five added together four *times*. (That's where the word *times* comes from when we say five times four.)

Now, if we can reduce all mathematics to four operators, and we can reduce those four operators to two pairs of inverse operations, and we know that of those pairs multiplication is just a kind of repeated addition, then we could say that all mathematics is just a matter of adding.

Ask yourself this:

Is a+b the same as b+a?

or, in symbols,

Is a+b=b+a true?

The answer is, generally, yes, although (believe it or not) there are some quantities that this can't be said of. But they're not ordinary numbers like we're used to and we can leave that until later.

What about this, then,

Is a*b=b*a?

Yes, again, generally speaking the answer is yes, so it doesn't matter which way round we write the sum or product. (*Sum* means numbers added together, *product* means numbers multiplied together.)

But now consider this:

Is a–b=b–a true?

Try it out with a few numbers for a and b. You will see that it would not be possible to replace a–b with b–a in an equation, because they're not equal. Similarly, division is one of those peculiar inverse operations, and you can't replace a/b with b/a. Try it with any pair of numbers you like. (With 5 and 2, a/b=2.5 and b/a=Ø.4).

In fact, division can be a problem to computer users, because it often causes the crash of a program with the report: ?DIVISION BY ZERO ERROR.

The computer will only go so far along the real line. In fact the biggest number the CBM 64 will handle is around 1ØØØØØØØØØ ØØØØØØØØØØØØØØØØØØØØØØØØØØØØØ. Not bad, you might think, but it is easy to exceed that limit with calculations involving probability, for example (which we'll look at later on).

Consider this: take the number one, and divide it (on your computer) by one. Command it to,

PRINT 1/1

and it will return 1. Then decrease the denominator by a factor of ten,

```
PRINT 1/Ø.1
```

and you get 1Ø returned.

Now try,

```
PRINT 1/Ø.Ø1
```

and you have 1ØØ. You can see the denominator getting smaller and smaller and the result getting bigger and bigger, and if you were to carry on reducing the denominator by a factor of ten (in practice slipping in an extra Ø after the decimal point each time), you will find that on commanding your Commodore to

```
PRINT 1/Ø.ØØØØØØØØ1
```

it will give you back 1E+Ø9.

Eventually, you will find that the computer will stop giving you back numbers like 1Ø or 1ØØ or 1ØØ or even 1000000 (one million) and it will automatically give answers in a condensed way. 1E+Ø9 is an example of that. This is called *scientific notation.* It saves cramp when writing out a string of zeros, or eyestrain when trying to read it. The number quoted above with thirty-eight zeros is a good example. Far better to write 'one followed by thirty-eight zeros', and better still to abbreviate that to '1E+38'. The point is (for now) that as you make a denominator smaller and smaller the result gets bigger and bigger. And if you try to make the denominator zero you will be in trouble, because the result will be too big for the CBM 64 to handle.

So, we have to avoid any situation where we might divide by zero when programming.

SUMMARY

1. There are four mathematical operators to remember: +,−,*,/
2. a+b=b+a and a*b=b*a
3. But a/b does not =b/a and a−b does not =b−a
4. There is no way of handling a fraction whose denominator is zero.
5. Very large numbers are written using *scientific notation.*

It might be useful to add a note about abbreviation as used by mathematicians and in mathematical texts. You will see products written without any multiplication operator between them. So that, to use an earlier example,

$C=2*\pi*R$ becomes $C=2\pi R$

Whenever the operator is absent you can be sure that you are meant to multiply the variables together, and you would read it 'C equals two pi r'.

Remember the formula for the area of a circle?

$A=\pi*R*R$

That was how we wrote it. But mathematicians (famous for their shorthand) have the habit of leaving out the multiplication operators, and they collect all the similar factors together and write a number above it to remind themselves how many they've collected. And they end up with:

$A=\pi R^2$

And they read it 'A equals pi r squared'. Which brings us to *powers*.

2 Powers

This is a shorthand for multiplying. As we can write $A=B+B+B+B$ as $A=4*B$, so the shorthand for $A=B*B*B*B$ is $A=B^4$.

But we have to write the four above and to the right of the B. This is inconvenient for computer users, and to get around it, computer designers came up with a symbol that says it all:

↑ (found on the ↑ key)

It is an 'up arrow' and is easy to remember because it gives you the clue by pointing up to where the little number might have been. So we would write, now,

A=B↑4

which is read as 'A equals B *to the power* four'. Remember that: 'to the power'

Here's a few for you to try on the computer. Enter,

PRINT 2↑3

You will get back 8 as your result, and this is so because 2↑3 really only means 2*2*2 (two times two times two). Try program 14.

Program 14 POWERS OF TWO

```
1 REM PROGRAM 14
2 REM POWERS OF TWO
10 FOR P=0 TO 10
20 PRINT 2^P
30 NEXT P
```

Try replacing line 2Ø with: 2ØPRINT "2 to the power";p;" equals "; 2↑p

This gives a better view of what's going on. Two to the power zero is one, two to the power one is two, two to the power two is four and so on ...

If a FOR loop is incorporated to draw a graph as it does its calculating, the computer will show you how the results rise dramatically.

```
1 REM PROGRAM 14A
2 REM GRAPH OF POWERS OF TWO
10 PRINT ""
12 POKE53281,0:REM BLACK BACKGROUND
20 FOR P=0 TO 8
30 PRINT "2^";P;" IS ";2^P
35 FOR Y=0 TO INT((P^2)/3)
40 COL=P+1:CHAR=102:X=P*2+20
50 GOSUB 1000
60 NEXT Y
70 NEXT P
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This illustrates the way powers of numbers soon take off, growing very large very quickly.

Now go back to simple program 14, and, in line 2Ø instead of putting 2↑p, replace the 2 with 3. Run it. The results are growing even faster now, because you are looking at powers of three. Now try it with 1Ø instead, so that you are looking at powers of ten:

```
2Ø PRINT 1Ø↑p
```

This is an interesting result because scientific notation has reappeared, because the numbers have grown as big as 10^9 and that's where the computer stops writing the whole number out.

Try,

```
1Ø FOR P=Ø TO 1Ø
2Ø PRINT 9↑p
```

The last value reached is in scientific notation. If you look closely at the form of it, you will see that it's in two parts. It might be difficult to read it, but in fact the 3.48678441 is just an ordinary decimal number and the E+Ø9 part means 'times ten to the power nine'. 'Ten to the power nine' is just a one with nine zeros after it:

1,ØØØ,ØØØ,ØØØ

Unlike computers, people often write big numbers with commas in so they can keep track better. Don't do that on your computer; it won't understand.)

So ... we can write the whole thing out as:

3.48678441*1ØØØØØØØØØ

or,

348678441Ø

or, in English,

3,486,784,41Ø

or even more in English,

Three thousand million (or Billion if you are American), four hundred and eighty six million seven hundred and eighty four thousand, four hundred and ten.

What a saving in effort 3.48678441EØ9 turned out to be!

Don't worry about losing some precision by quoting numbers to a limited number of decimal places; the number is accurate to a millionth of a percent, and scientists rarely manage that sort of precision in their data anway.

So far we have considered powers which are positive integers. What about other sorts of number? Try raising 9 to the power 1.5 by entering,

```
PRINT 9↑1.5
```

and you get 27. Which is surprising since we have defined powers as repeated multiplications, and how can you multiply nine by itself one and a half times?

3 More about powers

To illustrate powers let's look at *area*. Earlier, we talked about the *circle*, its *area* and its *circumference*, and we had:

C=2*PI*R (where R is the radius)

How do you make an equation for the *perimeter* of a square? First you need a variable for the perimeter, and P seems a reasonable choice. Then you have to decide what you want it *in terms of*. Suppose we want it in terms of the length of one side, which we can call s.

Then, we have to think what makes a square a square and not a triangle or a rectangle: it has four sides, and all the sides are equal, and each corner is a right angle.

From these things we can say that,

P=s+s+s+s

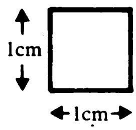
or

P=4*s

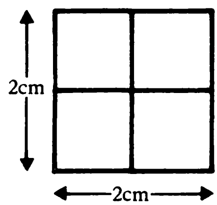
And there is a fromula for the perimeter of a square.

What about the *area* of a square?

Take the square which has sides one centimetre long.



To make a larger square by stacking 1cm squares together, we need four:



And the next biggest square is going to have nine 1cm squares in it.

Now, if our first square, one centimetre on a side, has one unit of area, then a square of side *two* cm has an area of *four* units. And a square of side *three* cm has an area of nine units.

Suppose we have six 1cm squares stacked so that there are two rows of three. Then the area of the rectangle (oblong) you get is six units, and the length of the rectangle is 3cm and its height is 2cm. *To get the areas we're just multiplying the length by the height.*

When you look at the general formula for a rectangle, it can be written:

A=L*H (Rectangle)

Where A is the area and L is the length and H is height. The squares are a special sort of rectangle though; because we know they have a length and a height which is equal, we can write:

A=s*s (Square)

Where s is the length of a side. And we can write:

A=s↑2

And that's why any number raised to the power two is called a *square*. Program 15 will generate squares. You can enter the length of a side and it will tell you the area. It also handles units to your taste. But you can enter any kind of unit you like, so if you enter 'eggs' for the unit of length, you will get 'square eggs' for the area. Better to stick to metres and inches etc, I think.

Program 15 Squares

```
1 REM PROGRAM 15
2 REM SQUARES
10 PRINT "":INPUT "ENTER LENGTH OF EACH
SIDE ";SIDE
20 INPUT "ENTER UNIT OF LENGTH ";UNIT$
30 PRINT
40 PRINT SIDE;" ";UNIT$;" SQUARE"
50 LET AREA=SIDE^2
60 PRINT"HAS AREA ";AREA;" SQUARE ";UNIT
$
70 FOR T=0 TO 5000:NEXT T:REM WAIT!!!
80 GOTO 10
```

Think of cubes. Cubes are three-dimensional squares, with length, height and depth all equal.

The *volume* (the block of space inside the object) of a cube is going to be the product of height, length and depth, which for any 3D oblong is:

V=L*H*D

where V is the volume, L the length, H the height and D the depth. For all cubes, we said that L=H=D, and we can call it s (for side), so that:

V=s*s*s

or,

V=s↑3

And we can write program 16 to calculate cubes. As with the case of squares, any number raised to the power *three* is called a *cube*.

Program 16 CUBES

```
1 REM PROGRAM 16
2 REM CUBES
10 PRINT"":INPUT "ENTER THE LENGTH OF A
SIDE ";SIDE
20 INPUT "ENTER THE UNIT OF LENGTH ";UNI
T$
30 PRINT
40 PRINT SIDE;" ";UNIT$;" CUBE"
50 LET VOL=SIDE^3
60 PRINT "HAS VOLUME ";VOL; "CUBIC ";UNI
T$
70 FOR T=0 TO 5000:NEXT T
80 GOTO 10
```

Now, if a 3D square is a cube, what's a 3D circle? A *sphere*. The formula for the volume of a sphere is:

V=(4*π*R↑3)/3 (Where R is the radius)

or, in mathematical form,

$$V=\frac{4\pi R^3}{3}$$

The volumes of all *solids*, as 3D shapes are called, are measured in *cubic* units. They don't have to be cubes.

In what we've been doing, we've used the general instruction:

PRINT X↑Y

and we've seen that X can be any kind of positive number. But what if X is a *negative* number?

Program 17 ARGUMENT DEMO

```
1 REM PROGRAM 17
2 REM INVALID ARGUMENT DEMO
10 FOR N=5 TO -5 STEP -1
20 PRINT N^2
30 NEXT N
```

This works with the positive values of n, and it also works with negative value, such as −2*−2=4. The Commodore can handle it, but some computers cannot. A negative number can be squared or raised to any power. If the power to which you are raising a negative number is *even* (2,4,6, etc) then the result is positive, but if the power is *odd* then the result is negative. This is so because pairs of negatives cancel one another out, but when you have an odd number of items in the product, there's one left over after all pairs have cancelled out and it makes the result negative.

Which brings us back to peculiar kinds of Y in X↑Y. We could have numbers other than positive integers. Let's take positive fractions. What is the meaning of X↑(1/2) for example.

Try this and don't worry about the rounding errors.

Program 18 ROOTS

```
1 REM PROGRAM 18
2 REM ROOTS
10 FOR N=1 TO 10
20 LET S=N^2
30 LET R=S^(1/2)
40 PRINT N;TAB(10);S;TAB(25);R
50 NEXT N
```

This program gives some insight into the meaning of a fractional power. Line 20 gives a variable S the value of N squared. Then s is raised to the power of (1/2) in the next line, and raising a number to the power two and then raising it to a half brings you back to the number you started with. We have done one operation and then done its *inverse*. We can make a general rule of it: X↑Y has an inverse X↑(1/Y). Try it out yourself with program 18, making the 2 in line 20 and the 2 in line 30 into 3, or 5, or whatever.

The name of the inverse process to raising to a power is called *taking a root*. And the power (1/2) is called a *square root*, and the power (1/3) is called a *cube root*.

For instance, with the area of a square, suppose you had a square and you knew its area was 100 sq cm. Then you could calculate the length of its side, because we already have that,

A=s↑2

and inversely,

s=A↑(1/2)

(Remember that you can do the same process to both sides of an equation.) So that in our case,

s=100↑(1/2)

Check it on the computer with,

```
PRINT 100↑0.5
```

The answer is 10, and you can check it by raising 10 to the power two.

Suppose we take a look at our original example now: 9↑1.5.

When you instruct your CBM 64 to PRINT 9↑1.5 it will return 27. Can you see why?

It makes it much easier if we write 1.5 as 3/2 because the two processes at work here are made more visible. (1.5 is equal to 3/2, or three halves.)

We are simultaneously *cubing* 9 and then taking its *square root*. That is, raising it to the power 3 and then raising that to the power (1/2).

If you are not clear on that, try the following:

```
PRINT 9↑(3/2)
```

then,

```
10 PRINT 9↑3
20 PRINT 729↑(1/2)
```

and you can see the secret laid bare.

This subject is often found in maths books under the heading INDICES (pronounced In –di – sees) because the power number is called an INDEX and the (Latin) plural is *indices*.

Think of decimal indices as fractions: the numerator raises to a power and the denominator takes a root. Program 19 runs through a few possibilities.

Program 19 FRACTIONAL INDICES

```
1 REM PROGRAM 19
2 REM FRACTIONAL INDICES
10 PRINT"":INPUT"ENTER X ";X
20 INPUT "ENTER A ";A
```

```
30 INPUT "ENTER B ";B
40 LET Y=A/B
50 LET R=X^Y
60 PRINT L,R
70 FOR T=0 TO 5000:NEXT T:GOTO 10
```

And you can choose the numbers involved as you like.

There remains one major category of indices we've not looked at yet: *negative indices.*

What does 2↑(−3) mean?

We can show its value is Ø.125, which is 1/8. Which is interesting because 2↑3 is 8. In general:

X↑(−Y)=1/(X↑Y)

X to the power minus Y equals one over X to the power Y.

This program demonstrates the rule:

Program 20 NEGATIVE INDICES

```
1 REM PROGRAM 20
2 REM NEGATIVE INDICES
10 PRINT"":INPUT"ENTER X ";X
20 INPUT "ENTER Y ";Y
30 LET L=X^(-Y)
40 LET R=1/(X^Y)
50 PRINT L,R
60 FOR T=0 TO 5000:NEXT T:GOTO 10
```

If you like you can automate the program:

Program 20a AUTONEGATIVE INDICES

```
1 REM PROGRAM 20A
2 REM AUTONEGATIVE INDICES
10 LET X=RND(1)*100
20 LET Y=RND(1)*10
30 LET L=X^(-Y)
40 LET R=1/(X^Y)
50 PRINT L,R
60 FOR T=0 TO 1000:NEXT T:GOTO 10
```

And the pairs of numbers you generate are identical.

4 Scientific Notation

We're now in a position to take a full look at the idea of scientific notation:

Program 21 POWERS OF TEN (COMPLETE)

```
1 REM PROGRAM 21
2 REM POWERS OF TEN (COMPLETE)
10 FOR P=-10 TO 10
20 PRINT 10^P;TAB(20);P
30 NEXT P
```

This is an extension of Program 14, and throws up some interesting points. It shows that scientific notation can deal with very small numbers as well as very large ones. That's what the + and − signs are all about.

The first point to note is that 1Ø↑Ø=1 (in fact any number raised to the power Ø is defined as 1).

The second point to note is that 1Ø↑1 is 1Ø. (In fact any number raised to the power 1 is itself: X↑1=X.) Perhaps you could write a small program to demonstrate those two points.

The next point to note is that when the number of zeros gets beyond eight, scientific notation is used and we get numbers like 1E+Ø9 (which reads '1 times ten to the power of nine'). Also, as the numbers go smaller than Ø.Ø1 we get scientific notation coming in with numbers such as 1E−6, which is read '1 times ten to the minus six' or zero point zero zero zero zero zero one, six zeros in all, one of them before the decimal point and five of them after it.

(The letter E stands for *Exponent*. This is just another word for index. There is a word for the decimal part of a scientific notation number too: *mantissa*. So that in the number 6.397E−5, the exponent is −5 and the mantissa is 6.397.)

This is the way the computer does it, but in a science or maths book very large or small numbers are written this way:

2.35×10^{6} (2,350,000)
1.42×10^{-4} (0.000142)

The Commodore also has a special function for giving square roots, called SQR. You must enclose the number following SQR is in brackets. Test it out with this program:

Program 22 SQUARE ROOTS

```
1 REM PROGRAM 22
2 REM SQUARE ROOTS
10 LET X=INT(RND(1)*100)
20 LET Y=X^0.5
30 LET Z=SQR(X)
40 PRINT Y;TAB(20);Z
50 FOR T=0 TO 1000:NEXT T:GOTO 10
```

All we are doing here is comparing the square roots generated by the two different expressions in lines 2Ø and 3Ø. The two columns should be identical.

SQR is not happy about negative numbers, and in your use of this function you must be careful not to try to use it on one. Can you see why this restriction should be?

Try to think in terms of the square and the square root as being inverses of one another. If you start with a number that is a square of another number, say 1ØØ which is the square of 1Ø, then the 1Ø is the square root of 1ØØ. OK, so far so good. But consider this: 1Ø is the number which when multiplied by itself gives 1ØØ, *but there is another way of multiplying a number by itself to get 100*:

−1Ø*−1Ø=1ØØ (because the negatives form a self-cancelling pair)

so that the answer to SQR (1ØØ) is really a double answer, 1Ø and also −1Ø. (This is the doorway to a whole branch of mathematics called *complex numbers*, and we can investigate that later.)

So, since (−1Ø)↑2 is actually 1ØØ, what number when squared gives −1ØØ? Without starting a discussion on complex numbers, there isn't any such number. The very act of multiplying a number by itself is creating a self-cancelling pair so that minus signs don't have a chance.

Program 23 is intended to introduce you to the SQR function, drawing a graph of the square roots of the integers.

Program 23 ROOT GRAPH

```
1 REM PROGRAM 23
2 REM SQUARE ROOTS GRAPH
10 PRINT ""
12 POKE 53281,0:REM BLACK BACKGROUND
15 CHAR=98:COL=1:Y=0
20 FOR X=0 TO 39:GOSUB 1000:NEXT X
25 X=0:CHAR=97
30 FOR Y=0 TO 24:GOSUB 1000:NEXT Y
35 CHAR=81  :COL=2
40 FOR X=0 TO 39:Y=INT(4*SQR(X)+.5):GOSU
B 1000:NEXT X
50 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

As you can see the graph rises quickly at first and then more slowly as the numbers get bigger – just the opposite of what happened when we graphed out the powers of two in Program 14a. This is because powers and roots are inverses.

The next program draws out the graphs of 2nd, 3rd and 4th roots, and you can see that as the Y in X↑(1/Y) increases, the result is an increasingly flat graph.

Program 24 ASSORTED ROOTS

```
1 REM PROGRAM 24
2 REM ASSORTED ROOTS
10 PRINT""
12 POKE 53281,0
15 CHAR=98:COL=1:Y=0
20 FOR X=0 TO 39:GOSUB 1000:NEXT X
25 X=0:CHAR=97
30 FOR Y=0 TO 24:GOSUB 1000:NEXT Y
35 CHAR=81  :COL=2
```

```
37 FOR N=2 TO 4:COL=N
40 FOR X=0 TO 39 STEP 2:Y=INT(4*X^(1/N)+
.5):GOSUB 1000:NEXT X
50 IF N=2 THEN PRINT TAB(5);"SQUARE ROOT
"
60 IF N=3 THEN PRINT TAB(5);"CUBE ROOT"
70 IF N>3 THEN PRINT TAB(4);"";N;" TH RO
OT"
75 FOR T=0 TO 2000:NEXT T:REM DELAY
80 NEXT N
90 FOR T=0 TO 5000:NEXT T:REM DELAY BEFO
RE FINISH
100 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

It is when you see a program like this one (slow though it is) that you realize what a saving in time is achieved by computers. If you could take the time to do manually what program 24 just did for you (if you could find sixth-root tables) you would see what I mean.

Take a look on the back of a 1984 pound note and see Sir Isaac Newton.

I'll let you into a secret about Isaac. He was, you might know, perhaps the greatest scientist who ever lived (I believe he was number one, and Isaac Asimov agrees with me!) but did you know that although he laid the foundations of optics, dynamics and mathematics during the course of his life, he failed a scholarship examination in 1663 (when he was 21 years old) due to 'woeful inadequacy in geometry'!

Another mathematical symbol

The square root sign (in a maths text, not on your computer is:

$\sqrt{x}$ (meaning square root of x)

and is very common. Often it is used over a whole expression like:

$\sqrt{ab-bc}$ (meaning root of (ab−bc))

And if you want different kinds of roots, a small number is hung over the beginning of the root sign to indicate the kind you want. So that:

$\sqrt[3]{y}$ (means cube root of y)

and,

$\sqrt[6]{z}$ (means sixth root of z)

Powers and roots are not as exotic as they may seem, and in any case while squares and square roots are quite common, cubes and cube roots less common by far, and the other kinds rarely met with.

Have you begun to get used to equations now? An equation is like a weighing scale that's in balance; one side equals the other side. But there are *non-equations* too, that are *not* in balance, where one side is *not* equal to the other. They are called *inequalities*, and they will be examined in the next chapter.

4

INEQUALITIES

1 Inequalities

Equalities (otherwise known as equations) are shown by an equals sign, so that

a=b

means a equals b.

And it is possible to have these five inequalities:

>	meaning greater than	(>)
<	meaning less than	(<)
>=	meaning greater than or equal to	($\geqslant$)
>=	meaning less than or equal to	($\leqslant$)
<>	meaning not equal to	($\neq$)

The symbols in brackets are the symbols found in maths books and there is a subtle difference in meaning between a computer-type inequality and a maths-type inequality.

The use to which an inequality is put is the important thing. An equals sign (=) is used in two ways: to define a relationship, as with the computer's LET statement LET a=b+c defines the variable a in terms of b and c) or with an IF statement:

IF a=1Ø THEN PRINT "X"

meaning, 'when the variable a has taken the value 1Ø, print "X"'.

Inequalities, however, are only used in the second sense, and cannot be used in a defining way. Try LET x<1Ø and it won't work. But you can use the inequality as part of an IF statement to provide a way of sorting out variables that take on different values. Try this program:

Program 25 SORT OUT

```
1 REM PROGRAM 25
2 REM SORT OUT
10 FOR N=0 TO 21
20 LET X=RND(1)-0.5
30 IF X<0 THEN PRINT X TAB(20);"NEGATIVE
"
40 IF X>0 THEN PRINT X TAB(20);"POSITIVE
"
50 IF X=0 THEN PRINT X TAB(20);"ZERO"
60 NEXT N
```

This program will go round the loop 22 times and generate a random number. The number (stored in the variable x) will be between $\frac{1}{2}$ and $-\frac{1}{2}$, then it will be tested by lines 30, 40 and 50 in turn. The number must be either positive or negative or zero, so each time there will be a number printed, and because the program has sorted the numbers out, the correct category will be printed alongside. This is the basis of all routines that make a distinction between numbers.

This is the way inequalities are used on the computer. It is possible, however, to use inequalities in a defining way with ordinary maths.

For example we can write:

$x<3$ (meaning x is less than three)

or,

$y>x$ (meaning y is greater than x)

These sorts of definitions are not precise in the same way that equations are precise; they only set the limit of the value that a variable can have. They do not tell you its exact value. But they can be used to define a range of values as well. So that we can say, for example,

$3<x<10$

meaning 'three is less than x and x is less than 10' or, if you like, x is between 3 and 10. Or we could have,

$$3 \leqslant x \leqslant 10$$

meaning x is greater than or equal to three, and less than or equal to 10.

We are using maths symbols here rather than computer language because inequalities can't be used on the computer to calculate anything. But on paper inequalities can be manipulated like equations. It is possible to write an inequality,

$$a>b$$

and add the same to both sides and it will remain true.

$$a+x>b+x$$

You can also replace the letters with numbers. For instance, you can let a be three and b be two. Then the inequality

$$a>b$$

is true (because three is greater than two) and if you choose any value for x, say five, then the inequality:

$$a+5>b+5$$

is also going to be true.

You can test the following rules that govern inequalities, listed in a table for you:

INEQUALITIES

If $a>b$	then	$a+x>b+x$	
If $a>b$	then	$a{*}x>b{*}x$	(when $x>\text{Ø}$)
and if $a>b$	then	$a{*}x<b{*}x$	(when $x<\text{Ø}$)
If $a>b$	then	$1/a<1/b$	
If $\text{Ø}<a<b$	then	$a\uparrow x<b\uparrow x$	(when $x>\text{Ø}$)
and if $\text{Ø}<a<b$	then	$a\uparrow x>b\uparrow x$	(when $x<\text{Ø}$)

The subject of inequalities is straightforward – just common sense in fact – but it gets very involved after a while. It is simply that if a number is compared to another number, it must be greater than, less than, or equal to it.

So that the opposite of $\geqslant$ is $<$, and the opposite of $\leqslant$ is $>$, because if it's not one it must be the other.

Here's a good way of testing it out:

Program 26 LOGICAL SORT OUT

```
1 REM PROGRAM 26
2 REM LOGICAL SORT OUT
10 PRINT "":REM CLEAR SCREEN
20 FOR N=0 TO 21
30 LET X=INT(RND(1)*20)
40 IF X>=10 THEN PRINT ;"*",X
50 NEXT N
```

Here you will get a printout of a star (asterisk) and the value of the randomly-chosen variable x *only if it is greater than or equal to ten.*

The use of the OR command may intrigue you. There is another one:

AND

and you can use them to mean much the same as you would in English to chain together a series of conditions.

Another exclusive pair is of course = and <> because two numbers are either equal or not equal.

2 ABS and SGN

Now it is time to introduce two more functions: ABS and SGN. They are BASIC commands – *functions* is the correct term.

ABS is a function just like INT was a function. Put INT in front of a number (or variable) and it will delete the decimal places and give you back a decimal-less number known as an INTeger. Remember that integer means 'whole number'.

Try:

PRINT INT(8.4)

and you get back 8. If you do this with negative numbers you will get the next lowest number, to that

PRINT INT(−6.4)

gives you back −7.

We can investigate the ABS function in the same way. Try this:

PRINT ABS(4.7)

and

PRINT (ABS Ø)

and,

PRINT (ABS 78639)

It doesn't do anything to those numbers. What use is a transparent function? But not *all* numbers go through untouched. Try,

PRINT ABS (−57)

Now we have found a number that is altered – because it was negative. The ABS function finds the *absolute value* of a number. In other words, it looks to see if the number is negative, and if it is, it makes it positive. ABS is a function that makes its use apparent as you are programming, for example to stop values being used that will print negative lines or column numbers. Also, remembering the restrictions on the square root function, we don't want a square root of a negative number, so it may be better to put in SQR ABS x in sòme circumstances.

SGN is even less used. It stands for SIGNUM, which is Latin for sign. It makes itself 1 if the number you apply it to is positive, and Ø if it is Ø, and −1 if it is negative. The best way to get a feel for it is by running Program 27.

Program 27 SIGNUM

```
1 REM PROGRAM 27
2 REM SIGNUM
10 PRINT "":REM CLEAR SCREEN
20 PRINT TAB(6);"NUMBER";TAB(16);"SIGNUM
"
30 FOR N=1 TO 21
40 LET X=INT((RND(1)-0.5)*10)
50 PRINT TAB(8);X TAB(19);SGN(X)
60 NEXT N
```

Where the number generated by the function in line 4Ø has a positive value, SGN has returned a 1, and where the number is negative we get −1. Where it is zero, SGN(x) is zero.

Now you have been introduced to the concepts we will examine in the rest of this chapter.

3 Functions

So far we have considered several functions. Look at this expression:

x=ABS(y)

If you plug in different values for the variable y, the equation will yield different values of x according to some rule, in this case that of the ABS function.

Another example of a function is:

x=SGN(y)

and again, once you know the rule, you have another mathematical tool.

We have also looked at these functions:

x=INT(y)

and

x=SQR(y)

and

x=π*y

Here is a list of all the functions we will discover:

PI	LOG	SIN
SQR	EXP	COS
ABS	ATN	TAN
SGN		

Column one should not pose any problems; if it does, you should re-read the book up to here to refresh your memory.

LOG is a function called 'natural logarithm', which is used by mathematicians to shorten calculations. EXP means 'exponential', and is the inverse of LOG so that understanding one will assist in understanding the other.

The other functions are used in geometry, important when dealing with circles and triangles, and will be easy to understand. SIN abbreviates 'sine', COS 'cosine', and TAN 'tangent' but the abbreviations are so common that they can be used when 'speaking' an equation involving one of them to yourself, so you might say,

x=COS y+TAN y

as 'x equals cos y plus tan y'.

ATN is short for 'arc tangent', but mathematicians write $\tan^{-1}$, some computers handle all three; the CBM 64 recognises only the last:

ASN	arc sine	$\sin^{-1}$
ACS	arc cosine	$\cos^{-1}$
ATN	arc tangent	$\tan^{1}$

The 'to the minus one' indices are there to show that these functions are the inverse of SIN, COS and TAN.

What is a function?
If we write an expression like y=SQR(x), we say that *y is a function of x*. We could write a *general* expression:

y=f(x)

which reads '*y equals f of x*' and means *y equals a function of x*, and that can be any function we care to put in, so that we could have y=ABS(x), or y=SGN(x) or, as we had above, y=SQR(x).

Why call it a *function*? Performing a function is simply doing a job of some kind. In maths, functions do their work on numbers. You put a number in and get a (sometimes) different number out. Think back over the ABS and INT and SGN functions and think what they did to the numbers to which they were applied.

Let's use a couple of functions you haven't studied and change numbers with them. Try this out, without worrying about what's going on:

Program 28 EXPONENTIALS

```
1 REM PROGRAM 28
2 REM EXPONENTIALS
10 PRINT ""
20 FOR X=1 TO 21
30 LET Y=EXP(X)
40 PRINT X,Y
50 NEXT X
```

You will get a column of x values (running from 1 to 21) and a second column of y values running from 2.7182818 to 1.31881573E+Ø9.

The speed with which that function grows may remind you of the powers of two function we considered earlier. Try replacing line Ø in program 28 with,

Ø LET y=2↑x

and see how that goes. That function will take off fast too, but not quite as fast as the one for y=EXP(x). Let's compare them directly:

Program 28a EXP INVESTIGATION

```
1 REM PROGRAM 28A
2 REM EXP INVESTIGATION
10 PRINT ""
20 FOR X=1 TO 21
30 LET Y=EXP(X)
40 LET Z=2^X
50 PRINT X,Y,Z
60 NEXT X
```

We can make a direct comparison now, and it seems that the exponential function is taking off faster than 2 to the x.

Now edit out the 2 in line 4Ø so that you're letting z=3↑x. What do you get? Then edit 2.5 into line 4Ø instead of 2 or 3. Try a few other values, and see if you can choose a number by trial and error that mimics the EXP function exactly.

It might be possible, and then again it might not. It should be fun to try to zero in on the number, but you will find that the number you are looking for has been staring you in the face all the time. When x=1, line 3Ø generates y=EXP(1) and prints it at the head of the second column. This number is 2.7182818. Try using that number in line 4Ø, writing: 4Ø LETZ =2.7182818↑x.

Now the two columns are almost identical, implying that the function in line 3Ø and the function in line 4Ø are almost the same. If we can produce an identical effect on all numbers then it must be an identical function, and in fact EXP, known as the *exponential function*, is defined as this curious number 2.7182818 to the power x. Like PI it is irrational, an infinitely-long decimal. The letter we use for the exponential number is (of course) 'e'. Like PI, because e is irrational, we can't put a precise value on it, but e=2.7182818 is good enough for our purposes. The mathematical definition of the exponential function is that it is the function whose rate of change equals itself.

An interesting thing about this function is that you can get at it using a *series*, a concept which we will discover later.

4 Introducing Graphs

A picture is worth a thousand words in mathematics. It is often much easier to see what is going on when you can have a picture of it, and a graph is nothing more than a mathematical picture.

We've used graphs before, notably in Program 14a and in the quite complicated program 24. We looked at the graph plotted by the functions for square roots, cube roots, fourth roots and so forth. A graph amounts to scales, like a ruler, or the Real line: a line marked off at equal intervals with a suitable scale. You could draw out a section of the Real line and do something with it. Suppose you had the data to represent on a graph, and you wanted to make some sort of comparison between the figures, then you could plot them along the real line.

For example, if we had the populations in millions of the twelve most populated countries in the world in 1973:

1. China	782
2. India	584
3. USSR	252
4. USA	214

5.	Indonesia	128
6.	Japan	108
7.	Brazil	100
8.	Bangladesh	74
9.	West Germany	62
10.	Nigeria	61
11.	Pakistan	59
12.	Great Britain	56

We could choose a real line running from zero to say 800, make a note on it to remember that the 800 marked means 800 *million* mark it off in equal sections and plot our data.

And we would get:

(all other countries in this space) Population (millions)

Why bother to draw the Real line to zero? Why not end it at 55 million, just below the last value? Because it would distort the data in this particular case: populations actually start at zero, and if you originated the graph at 55 million the graph would be telling you that 55 million is a minimum population.)

Plotting data along the Real line is not very graphic. It would be much more informative to have a two-dimensional display instead of a one-dimensional display. Here are programs that do both:

Program 29 POPULATIONS 1-D

```
1 REM PROGRAM 29
2 REM POPULATIONS
10 PRINT "":REM CLEAR SCREEN
20 POKE 53281,0:REM BLACK BACKGROUND
30 COL=1:CHAR=114
35 REM NOW DRAW LINE
40 FOR X=0 TO 39:Y=12:GOSUB 1000:NEXT X
```

```
50 FOR N=1 TO 12
60 READ X
70 CHAR=30:COL=3:Y=11:GOSUB 1000
80 NEXT N
90 DATA 5,9,28,35,1,23,18,38,18,33,5,29
100 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This reproduces the Real line with the points all along it. It's worth examining the operation of program 29, because it introduces a fresh computing concept. Line 90 holds the data (our twelve populations) and the computer is told that it's the data by the command DATA. So the program draws a line, then goes round the FOR loop, READing the data points each time, and plotting them along the line. The READ command tells the micro to pick out the DATA values, and we have used the variable name x to take the data.

Now try this.

Program 29a POPULATIONS 2-D

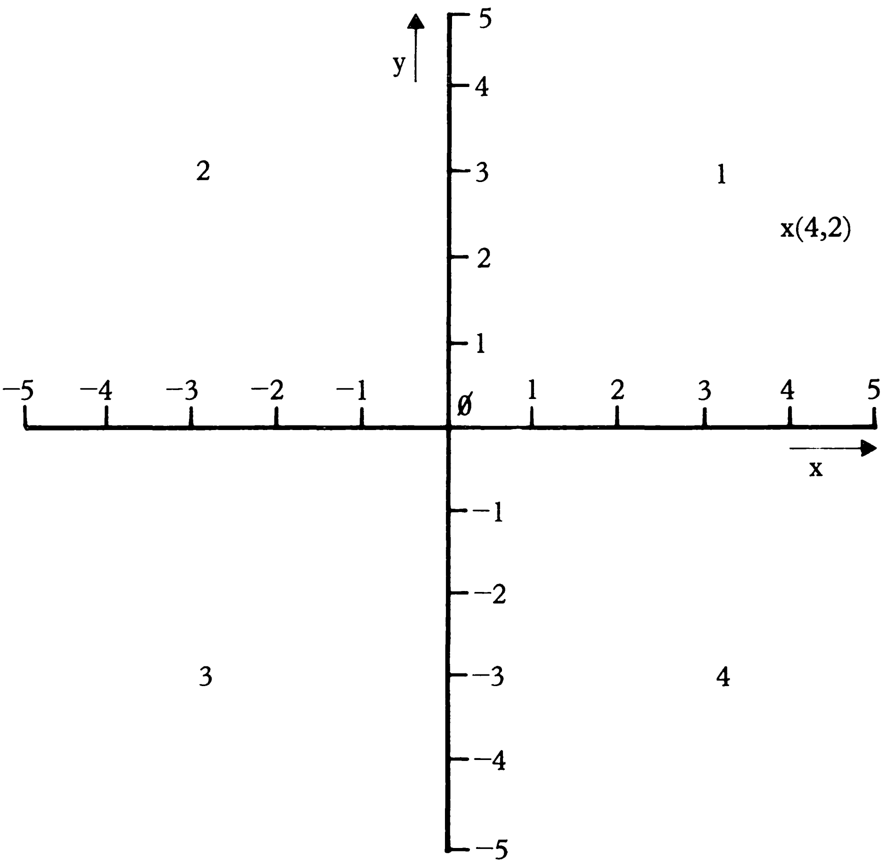
```
1 REM PROGRAM 29A
2 REM POPULATIONS
10 PRINT ""
20 CHAR=81:COL=1
30 FOR X=12 TO 34 STEP 2
40 READ Y
50 GOSUB 1000
60 NEXT X
70 DATA 24,16,12,9,6,4,3,2,1,0,0,0
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
```

```
1020 POKE 55296+ADR,COL
1030 RETURN
```

If we have a list of values connected to another list of values, then we ought to be able to make a graph out of it. If we have a function, or any expression involving a couple of variables and a few constants, we can get a graph out of it.

The convention to drawing graphs, established more than 300 years ago by Descartes, a French philosopher and geometry buff, is called the Cartesian System or Convention.

We normally use two dimensional graphs because they fit on sheets of paper best, and we say that the across-the-page direction is the 'x direction', and we call the up-and-down-the-page direction the 'y direction'. We take a couple of Real lines and lay them at right angles (90°) so that they cross at zero on each of them:



The lines are called *axes* - singular *axis*, plural *axes*, pronounced ak-seez. The place where they cross is called the *origin*. Obviously, we have four areas of paper created this way, and because four in latin is *quattuor*, we call each part a *quadrant*.

The quadrants are numbered anticlockwise starting at top right: 1,2,3,4 as shown, and any point in the first quadrant has a positive x value *and* positive y value. Any point in quadrant two will have a negative x value but a positive y value. Any point in quadrant three will have both x value and y value negative, and quadrant four has a positive x value but negative y value. We can summarise this in a table:

Quadrant	X value	Y value
1	+	+
2	−	+
3	−	−
4	+	−

To underline exactly what is meant by 'any point in the first quadrant', a point to consider is marked in quadrant one with a cross. Next to it is:

(4,2)

And that is the name of the point. We say the little cross is at the point (4,2), and so that there's no confusion we *always* specify a point by quoting the along-the-x-axis distance first, then the up-the-y-axis distance second, so that (4,2) means 'along four and up two'.

So all we have to do to get quadrant-one-type graphs, where x and y values are both positive, is to treat the computer screen as quadrant one, with the origin (point Ø,Ø) at the bottom left hand corner. We can move the origin to the centre of the screen if we want a four-quadrant representation.

Now to show a function on a graph. Try this first, so that we have the basics on the screen:

Program 30 AXES WITH CURVE

```
1 REM PROGRAM 30
2 REM AXIS WITH CURVE
10 PRINT "":REM CLEAR SCREEN
```

```
20 POKE 53281,0:REM BLACK BACKGROUND
30 FOR Y=0 TO 24:CHAR=66:COL=1:X=0:GOSUB
 1000:NEXT Y:REM DRAW Y AXIS
40 FOR X=0 TO 39:CHAR=82:COL=1:Y=0:GOSUB
 1000:NEXT X:REM AND THE X AXIS
50 FOR N=0 TO 13
60 Y=INT(N^2/8):X=0:COL=2:CHAR=81:GOSUB
1000:REM PLOT POINT
70 NEXT N
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

$y=x^2$

and it uses the values,

Y	X
∅	∅
1	1
4	2
9	3
16	4
25	5
36	6
49	7
64	8
81	9
1∅∅	1∅
121	11
144	12
169	13

The FOR loop stops at 13 because 14^2 is 196 and a Y value of 196 would not fit on the screen. What can be done about the fact it's squashed up into a portion of the screen near the Y axis?

Try this:

Program 30a EXPONENTIAL CURVE

```
1 REM PROGRAM 30A
2 REM EXPONENTIAL CURVE
10 PRINT "":REM CLEAR SCREEN
20 POKE 53281,0:REM BLACK BACKGROUND
30 FOR Y=0 TO 24:CHAR=97:COL=1:X=0:GOSUB
 1000:NEXT Y:REM DRAW Y AXIS
40 FOR X=0 TO 36:CHAR=64:COL=1:Y=0:GOSUB
 1000:NEXT X:REM AND THE X AXIS
50 FOR N=0 TO 7 STEP 0.2
60 Y=INT(EXP(N)/8)+1:X=INT(N*5)+1:COL=2:
CHAR=81:GOSUB 1000:REM PLOT POINT
70 NEXT N
80 FOR T=0 TO 5000:NEXT T
90 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

And there we have it, a perfectly reasonable representation of the equation,

y=EXP(x)

or, as mathematicians write it,

$y=e^x$

Now we've done exponential curves, but what about the humble straight line? Try this:

Program 30b THE HUMBLE STRAIGHT LINE

```
1 REM PROGRAM 30B
2 REM THE HUMBLE STRAIGHT LINE
10 PRINT "":REM CLEAR SCREEN
20 POKE 53281,0:REM BLACK BACKGROUND
30 FOR Y=0 TO 20:CHAR=97:COL=1:X=0:GOSUB
```

```
 1000:NEXT Y:REM DRAW Y AXIS
40 FOR X=0 TO 36:CHAR=64:COL=1:Y=0:GOSUB
 1000:NEXT X:REM AND THE X AXIS
50 FOR X=1 TO 35
60 Y=12:COL=2:CHAR=81:GOSUB 1000:REM PLO
T POINT
70 NEXT X
75 REM LABEL AXIS X & Y
76 COL=1:CHAR=25:X=2:Y=19:GOSUB 1000
77 COL=1:CHAR=24:X=35:Y=2:GOSUB 1000
80 FOR T=0 TO 5000:NEXT T
90 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This is a representation of the line whose equation is,

y=(a constant)

and the letters k or c are used to represent a constant, so that,

y=k

A special case is y=x. Can you see what this equation will look like on a graph? If not, make up two columns, one for the x values and the other for the y values and write them in, like this:

X	Y
Ø	Ø
1	1
2	2
3	3

etc

How are these values generated? By choosing our X values and using the equation to give the Y values. The equation this time happens to be y=x so that the X and Y columns are identical.

Can you convert this into a graph? When x is Ø, y is Ø so the curve goes through the point Ø,Ø, i.e. the origin. Also, when x is

1, y=1, too, so the curve goes through the point 1,1. Also, when x=2, y=2, so the curve goes through the point 2,2. And so on. You can automate it by writing y=x in line 60 of program 30b. Run it and see the result. It's a line that goes up from the origin at 45°, but it is a *straight* line.

There is a general equation for straight lines, in fact, and perhaps we should take a look at it:

$$y=mx+c \text{ (or in full, } y=(m*x)+x)$$

where m and c are both constants. Here is a program that lets you investigate the way that m and c affect the line, and also shows you the four-quadrant representation of a graph.

Program 31 THE STRAIGHT LINE

```
1 REM PROGRAM 31
2 REM THE STRAIGHT LINE
10 PRINT "":REM CLEAR SCREEN
15 INPUT "      ENTER SLOPE ";M
17 INPUT "      ENTER INTERCEPT ";C
20 POKE 53281,0:REM BLACK BACKGROUND
30 FOR Y=0 TO 20:CHAR=66:COL=1:X=20:GOSU
B 1000:NEXT Y:REM DRAW Y AXIS
40 FOR X=0 TO 39:CHAR=67:COL=1:Y=10:GOSU
B 1000:NEXT X:REM AND THE X AXIS
50 FOR N=-20 TO 19
55 LET Y=INT(M*N+C)
60 X=N+20:Y=Y+10:COL=2:CHAR=81:GOSUB 100
0:REM PLOT POINT
70 NEXT N
75 REM LABEL AXIS X & Y
76 COL=1:CHAR=25:X=19:Y=20:GOSUB 1000
77 COL=1:CHAR=24:X=37:Y=11:GOSUB 1000
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>20 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

first the program will ask you for a *slope*, and you should answer with a number between -3 and +3 (experiment with numbers outside this range later if you like). You will find that if you give m a negative value the slope is down from left to right, and if m has a positive value the line slopes up from left to right. This is what mathematicians mean by *positive* and *negative slopes*. And you can also see that the constant m is a measure of the slope of the line.

The other input value is for the *intercept*, which simply means that since the line must cross the y axis at some point, the value of the constant c will fix it. If you enter zero for the intercept, the line will cross the y axis at zero (i.e. through the origin), and you can try shifting the intercept up and down with other values. Try a family of lines with slope 1 and different intercepts.

Slope 1 means you have a line going up at 45°. Slope Ø is horizontal. If you experiment with bigger values the line gets steeper until it exceeds the tv picture's ability to distinguish it from the y axis. If the line were truly vertical its slope would be infinity. All this comes from the general equation of the family of straight lines:

y=m*x+c

This goes for all straight lines which can be drawn across Cartesian coordinates, and is not something specific to the computer.

There are families of other sorts of curve, for example families of parabolas and exponential curves, each with its general equation. The whole subject of 2-D graphs is known as *plane analytical geometry*, and we shall be returning to it later.

5

NUMBER SYSTEMS

1 Number Systems

It is generally believed that we use a decimal counting system because we have a total of ten fingers and thumbs. It seems quite reasonable to count on the fingers, so that when you get to the end of ten you start again with a fresh handful. The Latin for ten is *decem* (*Decem*ber is the *twelfth* month, but it was the tenth month in the Roman calendar). The word 'decimal' means a number system based on ten.

Well, if you compile a list of all one digit numbers you will get:

Ø 1, 2, 3, 4, 5, 6, 7, 8, 9

When you have reached the end of the ten symbols, you go on to 1Ø – in other words you start doubling up the digits, and when you reach 99 you have to start tripling them up, and so on. But there are systems which do not use ten as a base. The use of numbers to other bases has found its greatest application in the world of computers and microelectronics. Here is some more Latin:

Unus	One
Binus	Twofold

And now for a bit of Greek:

Hex	Six

Binus stands for *binary* and means a number system that has *only two types of number*, zero and one.

Now, consider decimal again. If I were to write the number,

4394

how would you know that it meant four thousand three hundred and ninety four? The position of the number digit within the whole number is important. It is not true to say that 4394 is the same as 3494 or 9443, because the positions of the digits is different. In 4394, we know that the digit on the right, 4, means 4, but the 9 is really 9 times 1Ø or 9Ø, the 3 is really 3 times 1ØØ which is 3ØØ and the figure 4 on the left hand end is really 4 times 1ØØØ which is four thousand. We can draw up a table showing how the positions are allotted in the decimal system:

thous- ands	hund- reds	tens	units	decimal point	tenths	hund- redths	thous- andths
4	3	9	4				
			0	·	1	2	5
1	2	3	4	·	5	6	7
10^3	10^2	10^1	10^0		10^{-1}	10^{-2}	10^{-3}

The table shows the meanings of three decimal numbers, 4394, 0.125, and 1234.567, with the column heads showing the effect of the digit being in that particular column. Notice that the first place to the right of the decimal point is used to indicate tenths, the second place to indicate hundredths, and the third to show thousandths. And of course you can extend the columns to deal with as many digits as there are in your number.

It's easier if you look at the way the column heads are written as powers of ten, down at the bottom of the columns. So that, 4394 would be:

$$(4{\star}10^3)+(3{\star}10^2)+(9{\star}10^1)+(4{\star}10^0)$$

And you can check that it works on your computer by using this command:

PRINT (4*1Ø↑3)+(3*1Ø↑2)+(9*1Ø↑1)+(4*Ø↑Ø)

You might also check the second number:

PRINT (1*1Ø↑−1)+(2*1Ø↑−2)+(5*1Ø↑−3)

We have now seen exactly what *decimal* means, and we can extend the idea to numbers with other bases. We can try it with binary; meaning 'connected with two'. Let's see what a binary number might look like:

1ØØ1Ø11Ø1

The number on the *right* is a one, and it will mean (1*2↑Ø) – and you will recall that any number raised to the power zero is one, so that bracket simplifies to (1*1) which is 1.

The next numbers from the right is a zero, and means (Ø*2↑1) or, simplifying (Ø*2) – recall that any number raised to the power one is itself, so that 2↑1=2, and remember also that any number times zero is zero, so that (Ø*2)=Ø.

We can carry on, moving in to the next number from the right, but it may be clearer to use a table:

1	Ø	Ø	1	Ø	1	1	Ø	1	Digits
8	7	6	5	4	3	2	1	Ø	Power of 2
256	128	64	32	16	8	4	2	1	Multiplier

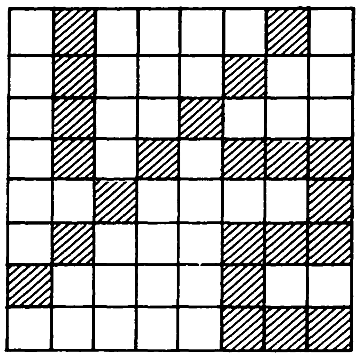
The number is copied into the top row and the column is numbered in the second row, so that the 1 in the far right column is really (1*2↑Ø) as we said above, and the middle *row* therefore shows powers to two. The third row just works out the powers of two for you. It just remains for you to go along the number from right to left, and if there's a one you add the number in the bottom row onto your total. Like this:

1+4+8+32+256=3Ø1

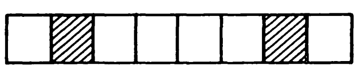
Binary systems have a great application in microcomputers because electric circuits only recognise the two states of ON and OFF. If we use zero for off, and one for on, we can get a circuit to simulate counting in binary, and then translate that into decimal.

Where it impinges on the user most is in the process of getting up a user-defined character. Let's say we are after a letter that the micro doesn't already have.

Suppose we want a symbol for a half ½. Design one on a grid of eight rows by eight columns like this:



Next, take the design one row at a time, and convert the row into an 8-digit binary number by making the black squares into 1 and the white squares into a Ø, so the top row:



becomes

Ø1ØØØØ1Ø

And you convert that binary number into a decimal number, which is 66.

Break the symbol down row by row and convert the numbers and you will get:

Row 1	01000010	66
Row 2	01000100	68
Row 3	01001000	72
Row 4	01010111	87
Row 5	00100001	33
Row 6	01000111	71
Row 7	10000100	132
Row 8	00000111	7

If you half shut your eyes you can make out your symbol-to-be in that block of binary numbers. In order to feed that data to the computer you would assemble it into a DATA statement, such as DATA 66,68,72,87,33,71,132,7 where the numbers are separated by commas, and run a program that would POKE it into memory. If you look at the Appendices at the back of the book, you will find the definition data for a whole set of useful symbols including the whole Greek character set.

Hex is short for hexadecimal (approximate Latin for sixteen), and describes a number system that has sixteen different types of number. But there aren't sixteen different kinds of number to write, so we use the capital letters from A to F. Like this:

Ø	8	
1	9	
2	A	(=1Ø)
3	B	(=11)
4	C	(=12)
5	D	(=13)
6	E	(=14)
7	F	(=15)

When we want to write the equivalent of 16, we have to start doubling up the digits again: 1Ø (note carefully that the number 1Ø in *hex* does not mean ten, but *sixteen*).

We could write a program to convert decimal numbers into hexadecimal numbers, or to do the opposite. But first we should have to know the way the powers of 16 go. Try this preliminary program to provide us with the required numbers:

Powers of Sixteen

The results get bigger very quickly, and you should get:

n	16^n
Ø	1
1	16
2	256
3	4096
4	65536

It is a property of powers of 16 that they end in the digit 6. Also, any power of sixteen must also be a power of two (because sixteen is itself a power of two - 16=2↑4 to be precise). And this is why hex has found computer applications. It's better suited than decimal which being based on ten, generates numbers that are not powers of two.

We will not go into the application of hex, because this book is not going to deal with such sophisticated computer matters as *machine code* or *assembly language*. But hex is an interesting example of another base system.

Program 32 HEX TO DECIMAL CONVERTER

```
1 REM PROGRAM 32
2 REM HEX TO DECIMAL CONVERTION
10 INPUT"  ENTER HEX NUMBER ";H$
20 LET D=0
25 REM CONVERT H$ (HEX) TO D (DEC)
30 IF H$>"" THEN FOR I=1 TO LEN(H$):A=AS
C(MID$(H$,I,1))-48:D=D*16+A+(A>9)*7:NEXT
40 PRINT:PRINT"   DECIMAL IS ";D
```

For example, if you want to know what FF is in decimal form, just run the program, and when you are asked for the number, enter: ØØFF. The answer will appear quickly: 255.

People who remember logarithms from the days before calculators think of them as columns of nasty numbers in a book of log tables. It took a lot of time to multiply two five-digit numbers, and even more to divide Ø.3479 by PI times 8.96584. Whole maths lessons used to consist of rows of children bashing through such meaningless calculations, helped by their log tables.

Think back to the EXP function,

y=EXP(x)

which can be found in maths books, looking like:

$y=e^x$

and means e to the power x (where e=2.718281828...) Remember this is called the *exponential function.*

The *logarithmic function* is the inverse of the exponential. Consider this: If y equals e to the power of x, what does x equal? In other words, how can we manipulate the function $y=e^x$ in order to get x on its own on the LHS?

We can't, not without having the inverse of the exponential function first. We defined the idea of inverse earlier on, but let's consider it now for a second or two. If you have a function and apply it to a number, and then apply the inverse of that function to the result, then you get back to what you started from. For example, if you were to take a number and multiply it by PI, then if you divided the result by PI, you'd get your original number back:

If,	$y=\pi\star4$
then,	y=12.566370̸61...etc
and,	12.566370̸61/π

will give us back 4, the number we first thought of.

Now, we need a function that is the inverse of the exponential function so that if

y=EXP(x)

we can apply the inverse function to *both* sides, and because the effects of the function and then the inverse of that function will cancel one another out, you will get:

(INVERSE EXP)y=(INVERSE EXP)EXP x

which is the same as:

(INVERSE EXP)y=x

And the real name for the (INVERSE EXP) function is the logarithm, abbreviated in Commodore BASIC (and also in some maths books) to LOG.

Our new expression is:

x=LOG(y)

which is exactly the same as,

y=EXP(x)

except that one expresses the relationship as x in terms of y, and the other expresses it as y in terms of x.

Now whenever you see an e to a power, you can deal with it. The log to the base e, as it is called (since it deals with e raised to some power) was discovered by the Scottish mathematician John Napier (1550–1617), and if you ever see a reference to 'Naperian logarithms' you will know that it refers to logs to the base e, often written in maths books as

$\log_e x$

But why choose e, a peculiar and ungainly number, for a base? Because it has to be a base of e in order to represent the inverse of e^x. There are many processes in nature that follow exponential functions, such as the growth of populations, so it is convenient to have a log to the base e to help deal with them. And that's why logarithms are called *natural.*

But that doesn't mean we can't have logarithms to any other base. We can have, in general terms, a log to base b, written

$\log_b x$ (meaning log to base b of x)

and b can take any positive value. The most common value happens to be 1Ø, because we use a decimal system, and so we have

$\log_{10} x$ (meaning log to base 1Ø of x)

and because it is the most common form of log it is frequently found without the little 1Ø being written after and below it as a reminder. So if you see

$\log_{10} x$

it means log to the base 1Ø of x, and if you see

$\log_e x$ or lnx

it means log to the base e of x.

Now for the program.

Program 33 NAPIER'S BONES

```
1 REM PROGRAM 33
2 REM NAPIERS BONES
10 PRINT "":REM CLEAR SCREEN
20 FOR X=1 TO 20
30 LET Y=LOG(X)
40 PRINT X,Y
50 NEXT X
```

This will print out the numbers 1 to 20 with their corresponding logarithms. As you can see, it takes off quite slowly, as you'd expect for a function that is the inverse of an exponential.

We can construct an even more striking representation of the function y=LOGx by putting it on a graph:

Program 33a NAPERIAN GRAPH

```
1 REM PROGRAM 33A
2 REM NAPERIAN GRAPH
5 PRINT ""
10 CHAR=67:COL=1
20 FOR X=0 TO 39:Y=12:GOSUB 1000:NEXT X
30 CHAR=66:COL=1
40 FOR Y=0 TO 24:X=19:GOSUB 1000:X=24:GO
SUB 1000:NEXT Y
50 CHAR=81:COL=3
60 FOR X1=0.2 TO 3.95 STEP 0.2
70 LET Y1=LOG(X1)
80 X=INT(X1*5+19.5):Y=INT(Y1*7+12.5)
85 GOSUB1000
90 NEXT X1
95 COL=1:REM PUT NUMBERS ON AXIS
100 X=25:Y=11:CHAR=49:GOSUB 1000
110 X=39:Y=11:CHAR=52:GOSUB 1000
120 X=0:Y=11:CHAR=45:GOSUB 1000
130 X=1:Y=11:CHAR=52:GOSUB 1000
```

```
140 X=18:Y=24:CHAR=50:GOSUB 1000
150 X=17:Y=0:CHAR=45:GOSUB 1000
160 X=18:Y=0:CHAR=50:GOSUB 1000
200 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This is more or less the same kind of thing we were doing before with the exponential function. It illustrates quite a bit about the log function. First, it is in the positive half of the graph, which means that the values of x never go negative. (In fact we chose to take values between Ø.2 and 3.95, because this is the most interesting region of the graph.)

Notice that as x gets closer to zero, the value of y gets more and more negative, and it looks as though when x *is* zero we can't put a definite figure on the function LOG(x). Mathematicians say that the function is *not defined* for this value. It may be that LOG (x) equals infinity, but we can't use that result on a computer, nor does it help to use it in any mathematical calculation.

As x grows towards one, the function grows closer to zero, and the curve actually crosses the x-axis at the point 1,Ø meaning that LOG(x)=Ø. Enter the command,

PRINT LOG(1)

and you will get Ø, which should prove that the computer agrees with us. Enter

PRINT LOG(Ø)

and you will see what is meant by invalid arguments!

The Scot, Napier, had an English friend called Henry Briggs (1561–1631) from Halifax in Yorkshire, who might be said to be the father of the log table. He popularised the log to base 1Ø. These are called *common* logarithms, but more formally (nowadays rarely) are called *Briggsian* logarithms.

You won't find any LOG_{10} function on the CBM 64. But as we shall see, that will not prevent us from using logs to base ten, or any other base we want. There are more functions than can be stored in a minicomputer. See page 140 of Commodore 64 User Guide. But we can reduce most of them to a combination of those functions which *do* appear on the computer.

Remember that multiplication is just repeated adding; there is a similar relationship between logs of different bases. It goes like this:

$\log_a b = \log_a c * \log_c b$

First let's take the Naperian and Briggsian logs, and find a relationship between them. Try to follow this argument:

If,

$\log_a b = \log_a c * \log_c b$

then we can let a=1Ø, b=x and c=e, and write:

$\log_{10} x = \log_{10} e * \log_e x$

So if we have a log of the Naperian kind ($\log_e$) of a number we can find the Briggsian log ($\log_{10}$). We can simplify this equation, because $\log_{10} e$ has a value Ø.434294481. (If you can find a set of log tables you can check this.) So our equation becomes:

$\log_{10} x = Ø.434294481 * \log_e x$

It's also true to say that

$$\log_a X = \frac{\log_a N}{\log_x N}$$

And that holds true for any base you want to use. If 1ØØ=1Ø^2 (check that it does) then it follows that

$\log_{10} 1ØØ = 2$

And if you enter this in your machine

PRINT LOG(1ØØ)/LOG(1Ø)

you get the answer 2, showing that it all holds true.

Also, we said that $\log_{10} x = Ø.434294481 * \log_e x$, so we have

$\log_{10}$ 1ØØ=Ø.434294481*LOG(1ØØ)

If you PRINT LOG(1ØØ) you will get 4.6Ø517Ø19, so we have

$\log_{10}100=0.434294481*4.60517019$

and a straightforward calculation of the RHS gives the result 2, by using PRINT Ø.434294481*4.6Ø517Ø19

And that is how to get at logs with bases other than e. Logs came to be associated with multiplying and dividing because of an aspect of powers that we have not investigated, so let's look at it now. Here is an equation that is obviously true:

100,000=100*1000

Now if we re-write it thus:

$10^5=10^2*10^3$

All we've done is convert the numbers into their power-of-ten forms. Notice the index numbers: 5, 2 and 3 are related to one another, since 5=2+3. This is not a coincidence. It's a general rule that if

$x^a=x^b*x^c$

then

$a=b+c$

And we have found a method of simplifying the operation of multiplication. It's obviously easier to add than to multiply, and so we have only to get our problem into index form to make that simplification possible. This is what taking logs does.

In general we can say that,

log (a*b)=log a+log b

and also,

log (a/b)=log a−log b

Adding replaces multiplication, and subtraction replaces division. Furthermore, you can make raising to a power simpler. Taking logs simplifies it to multiplication:

log (a↑b)=(log a)*b

This is all very interesting, but perhaps academic since the invention of calculators and computers. It is important, though, to have an idea of how functions work: this not only helps understanding of other areas, but helps mathematicians to think of new ways to use them.

6

ANGLES

1 Angles

Babylon was a middle-eastern kingdom 3,000 to 5,000 years ago. Astrology had the importance of a religion there, and in examining the heavens the astrologers presided over the beginnings of modern science in the form of astronomy.

The Babylonians used a number system that had a base of not ten, but *sixty*, perhaps because they were chiefly concerned with astronomy and used their number system in measuring angles. This does not mean that the Babylonians had sixty fingers, but they did have the longest surviving system of measuring time, and its base-sixty aspect is still intact.

Sixty seconds	=	one minute
sixty minutes	=	one hour (or one degree of a circle or an angle)

That holds true for both angles and time.

A six times sixty degrees (360) is one complete circle, or revolution. The way we indicate degrees, minutes and seconds is similarly antiquated:

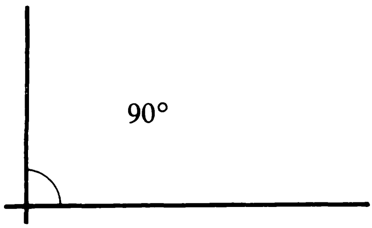
degrees shown by	$^\circ$
minutes shown by	$'$
seconds shown by	$''$

So if I were to write $24^\circ 15' 30''$ you would know what it meant.

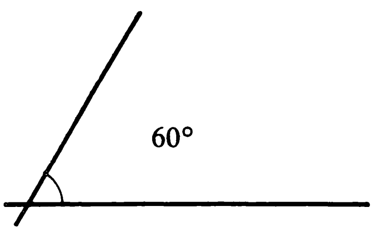
In geometry, angles appear most often in connection with the corners of triangles and, because things that rotate are connected with angles, with circles as well.

1 revolution = 360°
½ revolution = 180°
¼ revolution = 90°
1 degree = 60′ = 3600″
1 revolution = 21,600′ = 1,296,000″

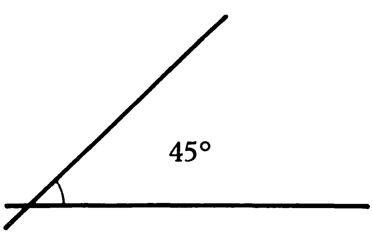
Angles of these commonly-found sizes look like this:



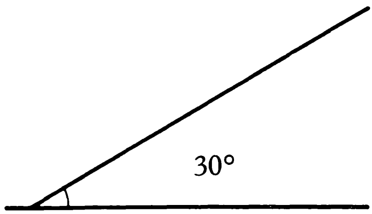
This is an angle of 90° called a ‘right angle’.



Angles less than 90° are called *acute*. 60° is an acute angle commonly found.



45° is half a right angle, also acute because 45°<90°.



The last of our examples, again acute, again very common. Note that 30°+60°=90°.

Let's ask the computer to demonstrate it:

Program 34 ANGLE DRAW

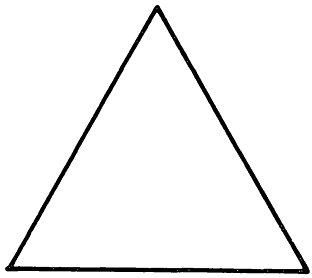
```
1 REM PROGRAM 34
2 REM ANGLES
10 PRINT ""
20 INPUT "ANGLE IN DEGREES ";A
30 PRINT
40 PRINT
50 IF A<90 THEN PRINT "ANGLE IS ACUTE"
60 IF A=90 THEN PRINT "ANGLE IS A RIGHT
ANGLE"
70 IF A>90 AND A<180 THEN PRINT "ANGLE I
S OBTUSE"
80 IF A=180 THEN PRINT "ANGLE IS A STRAI
GHT ANGLE"
90 IF A>180 THEN PRINT "ANGLE IS REFLEX"
```

The program sorts the angle you give it into whatever *type* it is: acute, right angle, obtuse, straight angle and reflex.

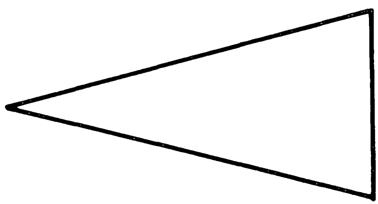
2 Sine, cosine and tangent

We will begin with a look at triangles. We said that a square was a shape made of four lines all the same length and all joined together at 90° to one another; a triangle is *three* lines all joined together, and the lines can be any length, and the angles can be just about any acute angle, so long as they add up to 180°.

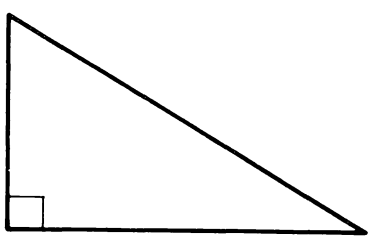
An *equilateral* triangle has three sides all the same length (the Latin *aequi* means equal, and *latus* means side):



An *isosceles* (pronounced I-sos-sell-eez) triangle has *two* sides the same length:

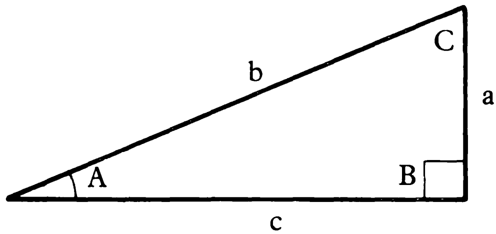


And a *right-angled* triangle has one angle in it of 90°:



We shall concentrate on an investigation of the right-angled triangle, to discover things about SIN, COS and TAN.

First we need to label the triangle with names for the sides and angles, so we can keep track of what we're talking about:



The angle A is made special by drawing an arc in it, and it is usual to make the right angle obvious by drawing in it an L-shape. So we have angles A, B and C and we have three sides a, b and c. The sides *opposite* the angles have the same letter: side a is opposite angle A, side c is opposite angle C, and side b is opposite angle B. The side opposite the right angle is called the *hypotenuse*, which comes from a Greek word meaning 'to stretch under'.

So far as the angle A is concerned, we have:

side a is the opposite side
side b is the hypotenuse
side c is the adjacent side.

Here is a small program to show it nicely.

Program 35 OPPOSITE, ADJACENT & HYPOTENUSE

```
1 REM PROGRAM 35
2 REM OPPOSITE ADJACENT AND HYPOTENUSE
10 PRINT ""
20 CHAR=1:COL=3:X=0:Y=0:GOSUB 1000:REM P
RINT A
30 INPUT "ADJACENT SIDE (38) ";C
35 REM DRAW LINE A-C
40 FOR X=1 TO INT(1+C):COL=7:CHAR=81:Y=1
:GOSUB 1000:NEXT X
50 INPUT "OPPOSITE SIDE (18) ";A
55 REM DRAW LINE C-B
60 FOR Y=1 TO INT(A+1):X=C+1:GOSUB 1000:
NEXT Y
70 REM DRAW LINE A-B
```

```
80 FOR Y=1 TO INT(A+1):X=INT(C*Y/INT(A+1
)+.5):GOSUB 1000:NEXT Y
90 LET AA=ATN(A/C)
100 PRINT "ANGLE=";AA*180/3.1415927;" DE
GREES"
110 FOR T=0 TO 3000:NEXT T:GOTO 10
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This program demonstrates that *the size of an angle in a right-angled triangle can be calculated if you know the lengths of any two sides*. There are three functions because there are three pairs of sides:

If you want to find the angle in terms of the opposite and opposites, you use the TANgent function.

If you want to find the angle in terms of the adjacent and hypotenuse sides, you use the COSine function.

If you want to find the angle in terms of the opposite and hypotenuse sides you use the SINe functions.

With reference to our labelled triangle above:

TAN A=a/c

The *tangent* of an angle A is equal to the length of the *opposite* side divided by the length of the *adjacent* side. And that the *cosine* of angle A equals the *adjacent* side over the *hypotenuse*, and

SIN A=a/b

means that the *sine* of A is equal to *opposite* over *hypotenise*.

The functions are just the ratios of sides in a right-angled triangle, but the range of their applications is extremely wide in surveying, architecture and countless other pursuits.

3 Inverse Trig Functions

SIN, COS and TAN are known as the trigonometric functions, or trig functions for short, and of course there is a set of three *inverse* trig functions.

The inverse of SIN is ASN ($\sin^{-1}$)
The inverse of COS is ACS ($\cos^{-1}$)
The inverse of TAN is ATN ($\tan^{-1}$)

In maths books the inverse trig functions are often written in the form shown in the brackets above. It is read 'sine to the minus one' or more simply *arcsine* (and so on for the other two). Commodore BASIC, however, recognises only ATN because we can manufacture the other two like this: ARCSIN X= ATN(X/SQR(−X*X+1) and ARCCOS X=(π/2)−ATN (X/SQR(−X*X+1)).

We could rewrite the three trig functions in terms of their inverses, just as we did with EXP and LN.

Remember that if,

y=EXP(x)

then another way of writing it would be,

X=LN(y)

And in the same way we can write,

SIN A=a/b

or inversely as,

A=ASN (a/b)

which is read: 'Angle A equals arcsine of a over b'.

And similarly, if

COS A=c/b

we can write

A=ACS (c/b)

And also if

TAN A=a/c

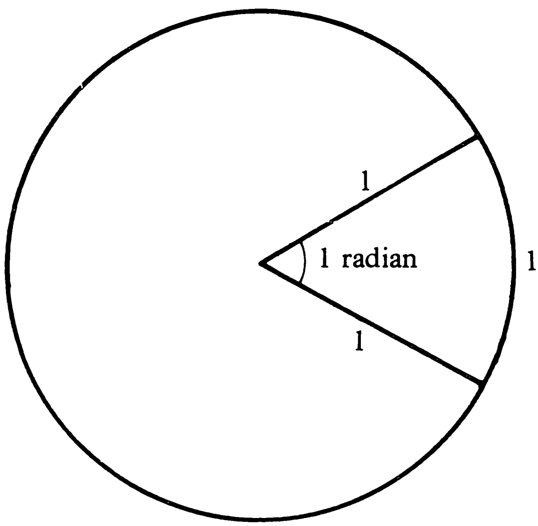
then we have

A=ATN (a/c)

One final point: the use of brackets here is *essential*. It's the same as we discovered earlier, that (a+b)*c is not the same as a+(b*c). If in doubt, it's a good idea to put brackets into an expression to make sure the computer doesn't misunderstand.

Experiment with program 35. Try entering the same number for adjacent side and opposite side, for example. The point of having adjacent and opposite sides the same length is that you would expect the angle to be 45°, and you can check that the program works correctly by seeing if it gives you 45° every time.

You can also verify the result of dividing by zero, by trying to enter Ø when you are asked for the adjacent side. The TAN function goes towards infinity as our angle goes towards 90°, and as the angle gets closer to zero, and in fact TAN Ø=Ø. Also try entering zero for the opposite side.



4 The Radian

Degrees have proved a durable unit of measure: The Babylonians used them, and so do we. But we have another form of angle measurement as well, called the *radian*. There are 360 *degrees* in a circle or a revolution, and there are 2*π radians in a revolution. 2*π is about 6.2831853 (Just do PRINT 2*π). If a circle is drawn with two radius lines on it (and make them 1 unit long), and arranged so that the part of the circumference between the radius lines is also one unit long (a part of the circumference is called an arc), then the angle made by the two radii (radii is plural for radius) is 1 radian. It's much clearer on the diagram:

As you can see, it looks to be about 60 degrees or so, but we can work it out exactly:

If

1 revolution=360°

and

1 revolution=2*PI radians

then

2*PI radians=360°

and, more simply, dividing both sides by 2,

PI radians=180°

so we have

$$1 \text{ radian}=\frac{180°}{PI}=180/3.14159\text{etc}$$

which brings us to

1 radian=57.29578 degrees

or in °,′,″ form,

1 radian=57° 17′44.8″

The Commodore is designed to expect angles quoted in radians and not in degrees. So if it is to print out the angle in degrees, the program must incorporate an equation that converts the units. Multiply a radian quantity by 180/PI and you have it in degrees, and inversely, multiply a degree quantity by PI/180 and you get it in radians.

This is a handy little program to generate a table of conversions:

Program 36 DEGREES AND RADIANS AND TANGENTS

```
1 REM PROGRAM 36
2 REM DEGREES RADIANS AND TANGENTS
10 PRINT""
20 PRINT "PRESS F1 WHEN TABLE STOPS SCRO
LLING"
30 FOR T=0 TO 1000:NEXT T:REM DELAY
35 N=1:PI=3.1415927
40 FOR X=0 TO 90
45 REM PRESS F1 EVERY 20 RESULTS
50 IF N/20=INT(N/20) THEN GET A$:IF A$<>
"" THEN 50
55 REM PRINT RESULTS
60 PRINT TAB(0);X;TAB(05);X*PI/180;TAB(2
0);TAN(X*PI/180)
70 N=N+1
80 NEXT X
```

When the angle x is less than about 10 degrees the radian measure of x is very similar to the TAN x value. As the conversion enters its third screenful, TAN 45° is 1 (which is what we would expect since the ratio of opposite and adjacent sides is one when the sides are equal). As we have seen, 1 radian is between 57 and 58 degrees, and in the last screenful, the value of the TAN function takes off rapidly, and is too big to calculate when we have 90°.

Mathematicians would say that 'as x tends to 90°, the value of the function tan x tends to infinity'. They won't say that it *is* infinity because they've never been to infinity to explore it – infinity is just out of the mathematical ball park. This 'tends to' business is quite convenient though; we could say, for example, that as the angle tends to 180° then the radian measure tends to PI.

Wherever a table of numbers can be generated a graph can be got out of it somehow:

Program 36a GRAPH OF TAN & RADIAN

```
1 REM PROGRAM 36a
2 REM GRAPH OF TAN AND RADIAN
10 PRINT "":CHAR=81:PI=3.1415927
```

```
20 FOR X1=0 TO 89
30 LET Y1=TAN(X1*PI/180)
40 LET X=INT(X1/2.5)
50 COL=3:Y=INT(Y1*2):GOSUB 1000
60 COL=2:Y=INT(X1*PI/180*10):GOSUB 1000
70 NEXT X1
100 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

See if you can improve on this program. (In fact, see if you can improve on any of the programs; after all you only get better with practice.) You should treat this book as a starting point for explorations of your own.

For following up the further ramifications of trig functions, we can get a table together for the SIN and COS functions. Let's construct a graph for a range of values of the function not over a ¼ revolution, but over a whole one; that is, from 0° to 360°. We will choose intervals of 5° so that we can get through all the data generated without trouble.

Program 37 SIN & COS

```
1 REM PROGRAM 37
2 REM SIN AND COS
10 PRINT ""
20 PRINT "PRESS F1 WHEN SCREEN STOPS SCR
OLLING":PRINT
25 PI=3.1415927
30 FOR T=0 TO 2000:NEXT T
40 FOR X=0 TO 360 STEP 5
45 IF INT((X+5)/16)=(X+5)/16 THEN GET A$
:IF A$<>"" THEN 45
50 Y1=SIN(X*PI/180)
60 Y2=COS(X*PI/180)
```

```
70 PRINT TAB(0);X;TAB(6);Y1;TAB(23);Y2
80 NEXT X
100 END
```

The first page full of data shows that when x is Ø Sin x is Ø also, but that Cos x is 1. Then as x goes towards 90°, Sin x rises from Ø to 1, so that when x=90°, Sin x=1. At the same time, Cos x is doing the opposite. It's going from Ø down to 1, so that when x reaches 90°, Cos x=Ø.

The next page follows the functions from 90 to 180 degrees, at which point Sin x has declined to zero again, Cos x, on the other hand has continued to decline right down to minus one.

By the time the functions have reached three quarters of a revolution at 270° they are at minus one for Sin x and zero again for Cos x, and by the time they are right round one complete revolution at 360° they have got right back to where they started with Sin x=Ø and Cos x=1.

A graph is definitely called for here, so we can see what shape these functions give us. We must first see that both functions are in the range from minus one to one. A mathematician might write:

$-1 \leqslant f(x) \leqslant 1$ (where f(x) is Sin x or Cos x)

Remember that the function notation 'f(x)' is read as 'a function of x' or 'f of x' for short. If we use the horizontal axis to measure off degrees, five at a time, and the vertical axis to measure from minus one to plus one, then we will get a graph extending over 255*5 degrees, =1275 degrees, or a bit over three and a half complete revolutions. See how these functions repeat. First the SIN.

Program 38 SINE GRAPH

```
1 REM PROGRAM 38
2 REM SINE GRAPH
10 PRINT"":PI=3.1415927
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B 1000:NEXT X
30 CHAR=97:COL=1:FOR Y=0 TO 24:X=0:GOSUB
 1000:NEXT Y
```

```
40 FOR X1=0 TO 640  STEP 5
50 Y=INT(10*SIN(X1*PI/180)+.5)+12:X=INT(
X1/15+1)
60 CHAR=81:COL=3:GOSUB 1000
70 NEXT X1
80 FOR T=0 TO 5000:NEXT T
100 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

It looks a bit like something moving in a spiral down a tube. The Sin function seems to have a lengthways straight-line aspect and also circular aspect, giving a helix, or spiral. But this is a 2–D graph and what you are looking at is a set of just over one and half revolutions of the sine wave. You can identify the points along the line. Where it starts is zero degrees, where it ends is 360 degrees (and notice that the sine curve has a value of zero at these points.) It also has a value of zero at the middle, and a quick cross-check will reveal that this point must be 180°. Similarly, when the curve goes down to −1, it must be 270° along the line, and when it's up at 1, it must be at 90°. The symmetry is very nice.

Consider the data thrown up by program 37, and think how a cosine wave will differ from a sine wave. It's the same form of curve, except that it's shifted along the x axis by 90°, (or PI/2 radians). Furthermore, the curves actually cross one another at two places. One is up in the positive top half of the screen, the other time is below the line. The positive one lies somewhere between 0° and 90°. For the curves to cross, at that point Sin x must equal Cos x, so we could ask: if Sin x=Cos x then *what value of x* are we talking about? If we

PRINT SIN (PI/4), COS(PI/4)

we will get two numbers that are the same. PI/4 is equivalent to 45°, so if Sin x=Cos x then x must be 45°. But what about the negative cross-over point? Correction: If Sin x=Cos x then x must either be 45° or (180+45)°. Only these two values of x *satisfy* the equation. (We'll be looking at that 'satisfy' concept later.)

To recap: first, three trig functions that come from the triangle. Then three inverse trig functions. Then measuring of angles in radians so that:

$$2\pi \text{ radians}=360°$$
$$\pi \text{ radians}=180°$$
$$\pi/2 \text{ radians}=90°$$

and so forth. Now we have the sine and cosine waves, and the observation that the tan function goes off to infinity and is not well-behaved like the other two.

What was that value for Sin 45° again? The same as Cos 45°, but what was its *value*?

Commodores like radians, so:

PRINT SIN (π/4) (which means Sin 45°)

and we get Ø.7Ø71Ø6781.

We are seeing that in maths everything is connected to everything else, but you only have a quarter of the jig-saw pieces in front of you so far. To illustrate the point, and elegantly introduce the next topic, try this:

PRINT 1/SQR(2)

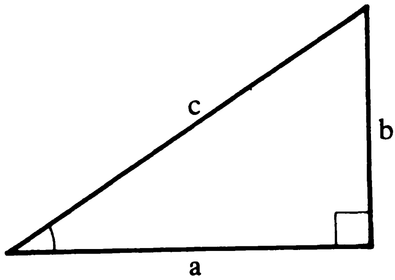
and we get Ø.7Ø71Ø6781. That number again!

6 The Theorem of Pythagoras

A theorem is a statement that requires proof.

Pythagoras was an ancient Greek mathematician (born 572 BC; died 497 BC). He thought that the Earth was the centre of the Universe, but he is remembered most for his theorem which states that in any right-angled triangle the length of the hypotenuse is equal to the square root of the sum of the squares of the other two sides.

Bring back the right-angled triangle:



In symbols, Pythagoras's Theorem states:

$$c^2=a^2+b^2$$

or, taking the square root of both sides:

$$c=\sqrt{(a^2+b^2)}$$

The bracket contains the 'sum of the squares of the other two sides' and the square root of it is equal to the hypotenuse.

Program 39 Pythagoras

```
1 REM PROGRAM 39
2 REM PYTHAGORAS
10 PRINT""
20 INPUT "INPUT SIDE 1 (<39) ";A
30 CHAR=81:COL=5:FOR X=0 TO A:GOSUB 1000
:NEXT X
40 PRINT"SIDE 1=";A
50 INPUT "INPUT SIDE 2 (<18) ";B
60 FOR Y=0 TO B:X=A:GOSUB 1000:NEXT Y
70 PRINT"SIDE 2=";B
80 FOR T=0 TO 2000:NEXT T
90 FOR X=0 TO A:Y=INT(B*X/A+.5):GOSUB 10
00:NEXT X
100 LET C=SQR(A^2+B^2)
110 PRINT"HYPOTINUSE=";C
120 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
```

```
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

You may recognise this as being similar to program 35, except that it works out the third side instead of an angle. And you will no doubt recognise Pythagoras' Theorem doing the working out in line 1ØØ.

Try entering a side of 4 and the other side of 3 and you will get 5. This is the schoolmasters' favourite, the 3,4,5 triangle. It will be shown to ranks of bored and lifeless schoolboys until the end of time, and all because conveniently, $5^2=3^3+4$*2* (Or if you like, 25=9+16).

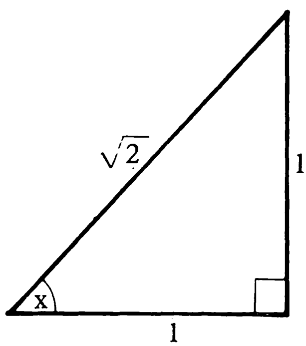
Now, in program 39, try entering 1 for side 1, and 1 for side 2. (Just consider it first, it won't draw a very good triangle.) We have:

Hypotenuse $=\sqrt{(1\uparrow 2)+(1\uparrow 2)}$

One squared is one times one (which is *one*), so our hypotenuse is the square root of one plus one, or in other words, the square root of two.

Hypotenuse $=\sqrt{1+1}=\sqrt{2}$

But let's look closer at that triangle:



We said that we could define trig functions in a right-angled triangle, and so if we look at the Sin of the angle marked as x, we will have

$$\text{Sin x}=\frac{\text{opposite}}{\text{hypotenuse}}=\frac{1}{\sqrt{2}}=\text{Ø.7Ø71Ø678}$$

which is where we came in ...

7 Cofunctions

You know what SIN, COS and TAN are now, but here's a few more you might meet in maths books:

SEC	Also known as Secant
COSEC	Alias the Cosecant
COT	Often called the Cotangent

You won't find them on the Commodore because once again you can manufacture them (see page 140 of the User Guide).

These co-functions are rarely used, but you should know a little about them. All triangles have 180°; that is, the three angles of any triangle will have a sum of 180°, or, if the angles are A, B and C, we have:

$A+B+C=180°$ (=PI radians)

In particular, the right-angled triangle for which we know already one angle is 90° (let's suppose that is angle C) will obey this relationship:

$A+B+90°=180°$

or, subtracting 90° from both sides,

$A+B=90°$

Angles A and B are called *complementary* angles because they complement one another (meaning they add up to 90°). The two left-over angles of any right-angled triangle are *always* complementary. (If one is 45 the other is 45, if one is 30 the other must be 60, and so on.) This is a consequence of a right angled triangle already having one angle of 90° leaving 90° to be split between the other two.

And that's where the *co* comes from in cofunction. The sine of 30° is equal to the *co*sine of 60° for example. And we can generalise:

SIN x=COS (90−x)

Then we have also

TAN x=COT (90–x)

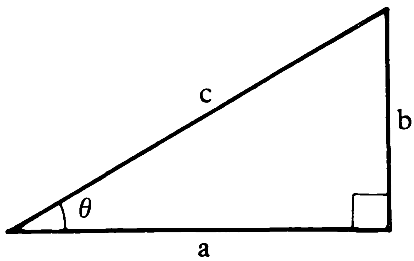
And also

SEC x=COSEC (90–x)

So, for example, we can say Tan 20° is Ø.36397Ø234,just do PRINT TAN (2Ø*π/18Ø) - so Cot(90–20)=Cot 70° is Ø.36397Ø234 too.

This will crop up again when we examine something called TRIGNOMETRIC IDENTITIES.

To summarise the six trig functions:



(Note that the angle here has been marked θ which is the Greek letter 'theta'.)

Reciprocal Relations

SIN θ=b/c	COSEC θ=c/b	COSEC θ=1/SIN θ
COS θ=a/c	SEC θ=c/a	SEC θ=1/COS θ
TAN θ=b/a	COT θ=a/b	COT θ=1/TAN θ

In this context *reciprocal* means 'one over' so that the reciprocal of n is 1/n, the reciprocal of Sin x is 1/Sin x, or as we have just seen, Cosec x. Now if

Sin x=b/c   and Cos x=a/c

then

Sin x/Cos x is $\frac{b/c}{a/c}$

and that can be simplified by multiplying the top and bottom by c, leaving b/a, and b/a is equal to Tan x. So that

$$\frac{\text{Sin x}}{\text{Cos x}}=\text{Tan x}$$

And taking the reciprocal of both sides (remember that when you do 'one over' to a fraction it just turns upside down, so that 1/(Sin x/Cos x)=Cos x/Sin x), so that

$$\frac{\text{Cos x}}{\text{Sin x}}=\text{Cot x}$$

(because 1/Tan x is Cot x).

Let's try a program.

Program 40 TRIGNOMETRIC FUNCTIONS

```
1 REM PROGRAM 40
2 REM TRIGONOMETRIC FUNCTIONS
10 PRINT ""
20 INPUT"INPUT ANGLE IN DEGREES ";X
25 IF X=0 THEN X=X+1E-6
30 PRINT:PRINT
40 PRINT "ANGLE IN DEGREES ";X
50 LET R=X*3.1415927/180
60 PRINT "ANGLE IN RADIANS ";R
70 LET S=SIN(R)
80 PRINT "SINE OF ANGLE ";S
90 LET C=COS(R)
100 PRINT "COSINE OF ANGLE ";C
110 LET T=TAN(R)
120 PRINT "TANGENT OF ANGLE ";T
130 LET SEC=1/COS(R)
140 PRINT "SECANT OF ANGLE ";SEC
150 LET CSEC=1/SIN(R)
160 PRINT "COSECANT OF ANGLE ";CSEC
170 LET COT=1/TAN(R)
180 PRINT"COTANGENT OF ANGLE ";COT
190 PRINT
200 PRINT "PRESS F1 KEY"
210 GET A$:IF A$<>"" THEN 210
220 GOTO 10
```

For any of the interesting values of angles like 0°, 90°, 180° etc. there will be no print out. That's because there is usually one function that goes to infinity and stops the computer executing the program. That's why writing a program that will make a graph of these funtions over one revolution is not as easy as it was to draw out SIN and COS in program 38 and its variants.

Program 41

```
1 REM PROGRAM 41
2 REM SIX TRIG GRAPHS
5 PI=3.1415927
9 FOR N=1 TO 6
10 PRINT "":POKE 53281,0
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B 1000:NEXT X:REM X-AXIS
22 REM DRAW THREE Y-AXIS
25 FOR X=9 TO 29 STEP 10
30 CHAR=66:FOR Y=0 TO 24:GOSUB 1000:NEXT
 Y
35 NEXT X
36 COL=1:CHAR=48:Y=12:X=0:GOSUB 1000:CHA
R=94:X=19:GOSUB 1000
37 CHAR=81:COL=N
40 FOR X=0 TO 39:DEGREES=360*X/39
50 RAD=DEGREES*PI/180
60 IF N=1 THEN Y=INT(12+10*SIN(RAD)+.5)
70 IF N=2 THEN Y=INT(12+10*COS(RAD)+.5)
80 IF N=3 THEN Y=INT(12+10*TAN(RAD)+.5)
85 IF COS(RAD)=0 OR SIN(RAD)=0 OR TAN(RA
D)=0 THEN 125:REM TRAP DIV BY 0 ERRORS
90 IF N=4 THEN Y=INT(12+4*(1/SIN(RAD))+.
5)
100 IF N=5 THEN Y=INT(12+4*(1/COS(RAD))+
.5)
110 IF N=6 THEN Y=INT(12+4*(1/TAN(RAD))+
.5)
120 GOSUB 1000
125 NEXT X
```

```
180 FOR T=0 TO 5000:NEXT T
190 NEXT N
200 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Let's analyse what we have. The first vertical line represents 90°, the second 180°, and the third 270°. The right hand edge of the screen is 360°, and the left hand edge is 0°. Both Sin and Cos have done as we might have expected from what we already know. But what about the TAN function?

The TAN function tends to infinity as the angle tends to 90°, then it reappears as if from minus infinity, until it reaches zero at 180°. Then between 180 and 360, it does exactly the same as it did between 0 and 180.

The Cosec Function comes down from infinity at 0°, levelling out at value 1 when the angle is 90°. It then does the reverse between 90° and 180° zooming up from 1 to infinity. Then between 180 and 360 it does something similar, only with minus infinity and minus one.

The SEC function draws a picture that looks like a sad face. It is a shifted version of the COSEC function, its limits being again plus and minus infinity and plus and minus one.

The COT function makes a double journey from infinity to minus infinity.

If you are trying to read this book without looking at the programs on the screen as well, it might look like gibberish. You've got to run the programs *and* read the book in order to get a grasp of the fundamentals of Computer maths, so that you are ready to go on to ideas that are a bit more advanced.

7

POLYNOMIALS

1 Polynomials

Time for some more Greek jargon.

Prefix	*Meaning*
Mono-	single
Bi-	twice
Poly-	many

If we attach those prefixes to the word *nomial*, meaning number, we get monomial, binomial and polynomial.

A *monomial* is an algebraic expression involving only one term, so that

$2x$	(2*x)
$-5xy$	(−5*x*y)
$3q^2p$	(3*q*q*p)

are all monomials.

A *binomial* involves two terms:

$3x+4y$	(3*x)+(4*y)
p^2-q	(p*p)−q
$a+x$	

are all binomials.

Polynomials have many terms, and

$x^2+3xy+15$(x*x)+(3*x*y)+15

is an example.

In manipulating algebraic expressions we have to follow a few simple rules.

First, we can only add (or subtract) terms if they are *like* terms (of the same type). You can't add five oranges to six apples, but you may add three pears to six pears.

For example, if we have $8xy-2xy$, it can be simplified to $6xy$.

Or, if we want to add $4x^2-10xy+5y^2$ and $2x^2+4xy-2y^2$ together, the result would be:

$4x^2$ added to $2x^2$ giving $6x^2$

and

$-10xy$ added to $4xy$ giving $-6xy$

and

$5y^2$ added to $-2y^2$ giving $3y^2$

and when we collect them up, we have:

$6x^2-6xy+3y^2$

Now, how about multiplying and dividing? Suppose we had $3x$ and $5y$ and $6xy$ and wanted to multiply them together. It would be:

3*x*5*y*6*x*y

which would be 3*5*6*x*x*y*y or, considering it further, 90*x^2*y^2 which we could write

$90x^2y^2$

Suppose we had two binomials and wanted to multiply them together:

(x+y)*(x+y)

Then we would have to multiply each of the two terms of the first binomial by the terms of the second binomial, so that

x*x would give x^2
x*y would give xy
y*x would give yx
y*y would give y^2

But notice that x*y=y*x, so that when the terms are collected we get

$x^2+xy+xy+y^2$

or

$x^2+2xy+y^2$

In full, this result can be shown:

$(x+y)(x+y)=x^2+2xy+y^2$

The reverse of this process is called *factorising*, and the two *factors* of $x^2+2xy+y^2$ are said to be $(x+y)$ and $(x+y)$.

Remember that everything in mathematics is connected to everything else. At the end of chapter four we saw that the equation of a straight line is of the form $y=m*x+c$, where m the slope of the line and c is the point where the line crosses the y-axis (called the y-axis intercept). Ask what condition must be satisfied if two lines are parallel when they are written

$y=m_1*x+c_1$

and

$y=m_2*x+c_2$

If the lines are to be parallel, their slopes must be the same; and that means (in our two equations) that $m_1=m_2$. An equation of the form $y=mx+c$ is called a *linear* equation (because its graph is a *line*); equations that include terms that have x^2 in them are called *quadratic* equations.

2 Quadratic Equations

These have a general form,

$a*x^2+b*x+c=0$

and it might be useful to examine this closely.

First, we can see that there is a term in x^2, one in x^1, and one in x^0. The term x^2 is multiplied by a constant here called a, but in practice taking any value positive or negative. The term in x^1 (and we leave out the index because x^1 *is* x) is multiplied by another constant able to take any value and here called b. The term in x^0 is *one*, and it is multiplied by a third constant (here called c), and since writing $c*1$ would be going the long way about it, the final form in general terms is:

$a*x^2+b*x+c=0$

One further point about the form of the equation is that it is set equal to zero. This is because if it was set equal to any other number, you could always subtract that number from both sides and get back to the standard general form. For example, if we had

$3x^2+4x+5=7$

we could subtract 7 from both sides and we would get

$3x^2+4x-2=0$

But why go to all this trouble to get the equation into a standard general form? The question requires a better understanding of what the equation actually represents. If we go back to a linear equation, we can see that if for example

$3x+5=0$

we can subtract 5 from each side and get

$3x=-5$

and then divide each side by 3 and get

$x=-5/3$

In other words, we have found a *value* for x that *satisfies* the equation $3x+5=0$. We can see that it satisfies the equation because if we substitute $-5/3$ for x in our original equation, it checks out as true:

$3\star(-5/3)+5=0$

Mathematicians say that they have *solved the equation and that the value* $x=-5/3$ is the *solution.*

In this case, there is no other number that will satisfy the equation. The equation has only one solution, called a *unique solution.*

But there are equations with more than one solution that satisfies them. Here is an example:

$x^2-16=0$

This is obviously not a linear equation because it has a term in x squared, and in fact it fits the standard quadratic form where a is 1, b is zero, and c is -16:

$1*x^2+0*x+(-16)=0$

which simplifies to our equation:

$x^2-16=0$

Let's try to solve it, first adding 16 to both sides:

$x^2=16$

and then taking the square root of both sides:

$x=4$

And there we have it, a *solution*, and we can check it out by substituting this value of x back in the equation like this:

$4^2-16=0$

which is

$16-16=0$

But is 4 the *only* solution? There is another value of x that satisfies the equation:

$x^2=16$

Clearly, minus four times minus four is 16 (because two negative numbers are multiplied together the negatives cancel one another out) and when we substitute back into the original equation we have our check.

The most convincing evidence for this can be got from your CBM 64 by commanding:

```
PRINT -4*-4
```

and you get 16.

While you're at it, try this:

```
PRINT -4*-4*-4
```

and you get -64, and the minus sign is back because two of the minus signs cancel one another and one is left over.

Let's repeat that this particular quadratic equation has *two* solutions, 4 and -4, and we could write the solution as $x=\pm4$ (read as x equals plus or minus four).

The next step is the substance of this chapter and the basis for a very important equation that all students of maths must know very well. So far we have seen how easy it is to solve a quadratic equation which has the b constant equal to zero. It is much more difficult if that b constant is non-zero. If the term in x^1 has a non-zero coefficient, a mathematician might say that the puzzle is *locked*. It's not obvious how to solve it.

(Just to keep you on the tracks, you'd better know that the constants a, b and c in our general form are called *coefficients*. Let's look at an example: with coefficient a equal to 2, coefficient b equal to −3, and coefficient c equal to 1, we get

$$2x^2-3x+1=0$$

It's not at first clear how to solve that, but in fact the solutions of any quadratic can be found from this equation:

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

That means that the coefficients of any quadratic can be plugged into the RHS of that equation and it will give us the *two* values of x that are the solution of the original quadratic.

To demonstrate this on the quadratic we quoted above: we had a=2, b=3 and c=1, so our solutions are given by

$$x=\frac{-(-3)+SQR((-3)\star(-3)-(4\star 2\star 1))}{2\star 2}$$

Let's simplify it one step at a time:

$$x=\frac{3\pm SQR(9-8)}{4}$$

so

$$x=\frac{3\pm SQR1}{4}$$

and SQR 1 is 1, so we have our two solutions:

$$x=3/4+1/4=1$$

and,

$$x=3/4-1/4=\tfrac{1}{2}$$

Substitute first one, then the other of these solutions back to the original quadratic to check that our magic solution finder works.

We will use this discovery to form a useful program, but there are some hidden dangers. Recall that you can't properly take the square root of a negative number, but our solution finder contains a square root of an expression and it is possible that the coefficients will combine in such a way that the expression goes negative.

Specifically, our root is $\sqrt{b^2-4ac}$, and if b^2 is greater than 4ac, we proceed as above. If $b^2=4ac$, then the root is zero, and our solution is just $-b/2a$. But if b^2 is less than 4ac, then we have a square-root-of-a-negative-number situation, and this means that we have no *real solutions.*

In fact, however, there are things called *imaginary* numbers and although we have no *real* solutions, we have a pair of *imaginary* solutions.

We will explore this in greater depth later. Meanwhile, here is a quadratic solving program.

Program 42 QUADRATIC REAL ROOTS

```
1 REM PROGRAM 42
2 REM QUADRATIC REAL ROOTS
10 PRINT""
20 PRINT TAB(8)"**ROOTS OF QUADRATICS**"
:PRINT
30 INPUT"ENTER COEFFICIENT A ";A
35 PRINT
40 INPUT"ENTER COEFFICIENT B ";B
45 PRINT
50 INPUT"ENTER COEFFICIENT C ";C
55 PRINT:PRINT:PRINT
60 PRINT"    ";A;"* X^2";
70 GOSUB200
80 PRINT;"+";B;"* X";
90 GOSUB200
100 PRINT;"+";C;"=0"
110 GOSUB200
120 IF B*B<4*A*C THEN PRINT:PRINT"          N
O REAL ROOTS":GOSUB200:GOTO10
130 R1=(-B+(SQR((B*B)-(4*A*C))))/(2*A)
```

```
140 R2=(-B-(SQR((B*B)-(4*A*C))))/(2*A)
150 PRINT:PRINT
160 PRINT TAB(10)"FIRST ROOT=";R1
165 PRINT
170 PRINT TAB(10)"SECOND ROOT=";R2
199 GOSUB200:GOSUB200:GOSUB200:GOTO10
200 REM DELAY SUBROUTINE
210 FOR T=0 TO 2000 :NEXT T
220 RETURN
```

It is important to understand that the *roots* of a quadratic are its solutions, and a quadratic's solutions are its roots.

3 More Quadratics

With a graph of the quadratic function we can see the effects of the constants in the equation. The curve drawn by a quadratic is a shape called a *parabola*, and it is possible to see how the coefficients effect it using program 43. You will see that if coefficient a is positive, then the curve is U-shaped. If a is negative, then the curve is an upside-down U-shape.

You can try using a range of different a coefficients, but the range −5 to 5 will be enough to show that the bigger the value of a, the narrower the curve, and the bigger negative values give narrower inverted parabolas.

Then you can try varying the b coefficient, keeping the a and c coefficients the same in order to investigate what effect this has. You will find that varying b shifts the parabola left and right (along the x-axis, that is) and varying the c coefficient has an effect on the vertical shift (along the y-axis)

If you try a=Ø, then you are reducing the equation

$y=ax^2+bx+c$

down to

$y=bx+c$

which you should by now recognise as the general equation of a line. Lines can be drawn with this program therefore by setting a=Ø. Try a=Ø, b=3, c=Ø, and you get a good enough straight line which goes through the origin (obviously, because c=Ø implies the y-axis intercept is zero.)

Program 43 QUADRATIC PARABOLAS

```
1 REM PROGRAM 43
2 REM QUADRATIC PARABOLAS
10 PRINT""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=10:GOSU
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 20:X=19:GOSUB1000:
NEXT Y
40 N=0
41 INPUT "ENTER A ";A
42 INPUT "ENTER B ";B
43 INPUT "ENTER C ";C
50 CHAR=81:COL=N:N=N+1
60 FOR X=0 TO 39:X1=19-X
70 LET Y=INT(A*X1*X1+B*X1+C+.5)+10:GOSUB
1000
80 NEXT X
90 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>20 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Program 43a will be useful if you want to explore the effects of varying only one coefficient at a time. (You will find that the apparently simple 'shift' caused by varying b is not really so simple. It can best be seen by letting a=Ø so that it reduces to a linear form.)

Program 43a A FAMILY OF PARABOLAS

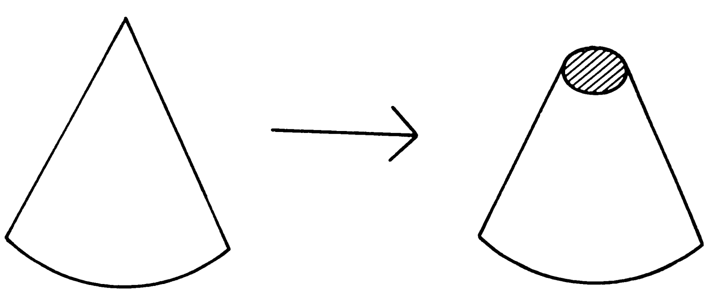
```
1 REM PROGRAM 43A
2 REM A FAMILY OF PARABOLAS
10 PRINT ""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=10:GOSU
```

```
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 20:X=19:GOSUB1000:
NEXT Y
39 N=0:B=0:C=0
40 FOR A=-.2 TO .2 STEP .1
50 CHAR=81:COL=N:N=N+1
60 FOR X=0 TO 39:X1=19-X
70 LET Y=INT(A*X1*X1+B*X1+C+.5)+10:GOSUB
1000
80 NEXT X
90 NEXT A
120 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>20 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This program is set up to vary b between −6 and 6, stepping 2, to give curves with b=−6, −4, −2, 0, 2, 4, 6 and a and c both equal to zero. You can investigate the effect of varying other coefficients by altering lines 39 and 4Ø to suit, *not* forgetting to alter line 9Ø to conform with line 4Ø.

4 Conics

The shape of the parabola was known to the ancient Greeks, because it belongs to a group of curves called *conics*, into which they did a lot of research. If the point of a cone is directly above the centre of the base circle, we call the solid a *right* cone



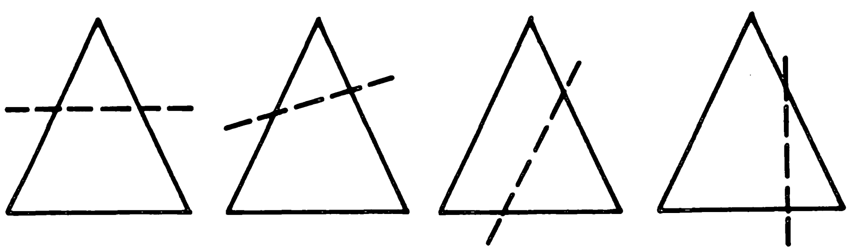
(because the line that runs down the centre of the cone makes a right angle with any radius of the base circle). The point of a cone is called the *vertex*. A cone could also have an oval base, so to be quite unambiguous one must refer to a *right circular* cone, which is the kind we will look at.

If you cut the top off the cone in such a way that the cut is parallel with the base, the shape of the cut will be circular.

To *section* means to cut, from the Latin *secare*. In mathematics, then, the cutting of cones is a subject called *conic sections*. There are four shapes you can get by sectioning cones:

1. Circle
2. Ellipse
3. Parabola
4. Hyperbola

If the cut is *exactly horizontal* we get a circle. If the cut is oblique we get an ellipse. If the cut is *exactly parallel* to the slanting side we get a parabola, and if it is more vertical than the slanting side we get a hyperbola. (*Para* is the Greek prefix for 'beside', and *hyper* is the Greek prefix for 'beyond'. Ellipse comes from a root meaning 'to leave behind'.)



Meditate for a moment upon the shapes you get from sectioning a cone. Like other geometrical shapes they crop up often in the real world. The ellipse, for example, became the talk of the scientific world in 1609 when Johannes Kepler (1571–1630) published a book explaining that the planets moved in orbits which were ellipses. (Until then, it was thought the planets moved in a complicated combination of circles.) But the real expert on conic sections was Apollonius, a Greek who lived between 260 and 190 BC, and who is responsible for much of this chapter.

Because we know a circle quite well already, let's look at it and the equation of it, and the coordinate geometry of it. If we draw a circle *with its centre at the origin*, then the equation relating the x and y coordinates of all points on the circle is

$$x^2+y^2=R^2$$

where R is the radius of the circle.

This needs to be put into a computer program to see it clearly. If a radius of 10 is chosen, then, since R=10, we have R^2=R*R=100, so our equation becomes

$$x^2+y^2=100$$

and we can get y alone on the LHS by rearranging it to

$$y^2=100-x^2$$

and taking the square root of both sides

y=SQR(100−(x*x))

we now have it in a form where we can run it through several values of x, and generate points we can plot out. (That way we can see if we *really* get a circle). Look for the basic equation in line 50.

Program 44 CIRCLE ONE ROOT

```
1 REM PROGRAM 44
2 REM CIRCLE ONE ROOT
10 PRINT ""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 24:X=19:GOSUB1000:
NEXT Y
40 FOR X=3 TO 19:X1=11-X
50 LET Y=INT(SQR(64-X1*X1+.5))+12
60 CHAR=81:COL=1:GOSUB 1000
70 NEXT X
120 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
```

```
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

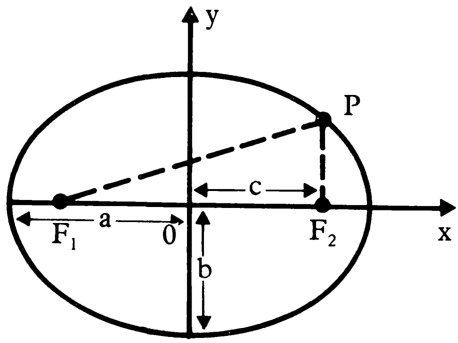
But this is only *half* a circle. Take a look at the equation. There is a square root again, and square roots have *two* solutions (one positive and one negative), and the half circle represents the positive root, and the other (negative root) can be arranged as follows:

Program 44a   CIRCLE BOTH ROOTS

```
1 REM PROGRAM 44A
2 REM CIRCLE BOTH ROOTS
10 PRINT ""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 24:X=19:GOSUB1000:
NEXT Y
40 FOR X=3 TO 19:X1=11-X
50 LET Y1=INT(SQR(64-X1*X1+.5))+12
52 Y=Y1:CHAR=81:COL=1:GOSUB 1000
55 LET Y2=-INT(SQR(64-X1*X1+.5))+12
60 Y=Y2:CHAR=81:COL=1:GOSUB 1000
70 NEXT X
120 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

This makes the point nicely. The two halves don't join up exactly, but this is because we're choosing steps of 1 for x. If we use STEP Ø.1 in line 2Ø you'd get a better effect, and STEP Ø.Ø1 would be better still.

Is there a general equation for an ellipse then? Of course. To see what it means, look at the diagram below:



The jargon of ellipses is that the long axis is called the *major axis* and the short axis is called the *minor axis* (if the two are equal then your ellipse is a circle). An ellipse has two points called *foci* (the plural of *focus*), and if you choose any point on the ellipse, then the sum of the distances to the two foci is a constant, which means yields a definition of an ellipse: that curve given by a point which moves so that the sum distance to two fixed points is a constant.

In symbols, we have this, which you can relate to the diagram:

$F_1P+F_2P=$a constant.

It happens that when the point P is at the end of the major axis we can see easily that the constant is actually the length of the major axis, so that, since we've written a for the *semi*major axis, we can say,

$$F_1P+F_2P=2a$$

(It could also be figured out, using geometry and triangles and a couple of hours that $a^2=b^2+c^2$.)

The distance a is called semimajor axis (for half the major axis) and b is called semiminor axis (for half the minor axis). These distances a and b are maximum and maximum radius lengths, if you think of the ellipse as a squashed circle. The use of the semimajor and semiminor axes gives us this general equation for the ellipse when the centre of the ellipse is at the centre of a coordinate (graph) system

$$\frac{x^2}{a^2}=\frac{y^2}{b^2}=1$$

which you can compare with the general equation of the circle by setting both a and b equal to R.

To get the subject of the equation as y (as we had to do for the circle above) some work has to be done on the standard equation. First, multiply both sides by a^2 and b^2

$$\frac{x^2a^2b^2}{a^2}+\frac{y^2a^2b^2}{b^2}=a^2b^2$$

Then things can be cancelled on the top and bottom of fractions:

$$x^2b^2+y^2a^2=a^2b^2$$

Then we can subtract x^2b^2 from both sides:

$$y^2a^2=a^2b^2-x^2b^2$$

Divide both sides by a^2:

$$y^2=b^2-\frac{x^2b^2}{a^2}$$

And then make the RHS into a bracket by taking out a factor b^2, and take the square root:

y=±b*SQR(1−(x*x)/(a*a))

and we're all set to do our ellipse program.

Program 45 ELLIPSE

```
1 REM PROGRAM 45
2 REM ELLIPSE
10 PRINT ""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 24:X=19:GOSUB1000:
NEXT Y
35 A=10:B=5
40 FOR X1=-10 TO 10:X=X1+19
50 LET Y1=INT(SQR(B*B*(1-(X1*X1)/(A*A)))
+.5)+12
52 Y=Y1:CHAR=81:COL=1:GOSUB 1000
55 LET Y2=-INT(SQR(B*B*(1-(X1*X1)/(A*A))
```

```
)+.5)+12
60 Y=Y2:CHAR=81:COL=1:GOSUB 1000
70 NEXT X1
120 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Comparing this with program 44a. There are a great many similarities. The principal difference is that we have altered the circle-giving formulas into the ellipse-giving formulas, setting values for a and b in line 2Ø. Choose your own values for a and b to see the effect it has on the ellipse. Notice again that when a=b (that is, when you give them both the same value) you get an equation that is in essence the equation of a circle.

The flattening of the ellipse is specified by the choice of a and b, and mathematicians and astronomers call this *flattening the eccentricity*. The circle has an eccentricity of zero, and as the eccentricty increases so the ellipse gets more cigar-shaped.

There is a definition of eccentricity, and a symbol for it: e (not to be confused with the exponential constant, e). Think about the diagram of the ellipse we programmed earlier, and see that the distances we have marked as c is the distance from the centre of the ellipse to its focus (or one of its foci – it doesn't matter which one because the ellipse is symmetrical), so we can write

the focal distance is c
the semimajor axis is a

and we *define* the eccentricity as:

e=c/a

and since we had earlier that $a^2=b^2+c^2$, then it follows that $c^2=a^2-b^2$ and therefore,

$$e=\frac{SQR(a^2-b^2)}{a}$$

and a quick manipulation will give you:

$e=SQR(1-(b^2/a^2))$

So you can see that the eccentricity depends upon a and b in this curious manner. It would, of course, be possible to add line 9Ø to your program 45, giving a print-out of the eccentricity.

```
9Ø PRINT "ECCENTRICITY=";SQR(1-(b*b)/(a*a))
```

This particular program can't cope with situations where a<b, but running through the sample of values we find that:

a	b	e
1Ø	1Ø	Ø(when a=b it's a circle)
1Ø	5	Ø.8660254 (=Sin 60°)
1Ø	1	Ø.99498744
1Ø	Ø.1	Ø.99995

As the ellipse gets more flattened, the eccentricity gets closer to one.

The question arises: what happens when the eccentricity *is* one? The answer is that you get a *parabola*. But first, to give you a practical example of the ellipse, let's look at those elliptical wanderers, the planets.

PLANET	Eccentricity	Mean distance (millions of Km)
Mercury	Ø.2Ø56289	57.91
Venus	Ø.ØØ67864	1Ø8.21
Earth	Ø.Ø1672Ø9	149.60
Mars	Ø.Ø933791	227.94
Jupiter	Ø.Ø48455Ø	778.34
Saturn	Ø.Ø5564Ø2	1427.Ø1
Uranus	Ø.Ø472421	2869.6
Neptune	Ø.ØØ8584Ø	4496.7
Pluto	Ø.25Ø2	5898.9

As you can see, the eccentricities are quite small, so the orbits are very close to being circles. I might add that the *mean distance* of a planet from the Sun is going to be important to us if we mean to calculate the *maximum* and *minimum* distances between planet and Sun, the mean distance is equal to the semi-major axis. *Planets travel in elliptical orbits with the Sun at one focus.* This statement is known as Kepler's First Law (there are two others) and it allows calculation of the distances. The mean distance will be (referring back to the diagram) equal to the semi-major axis a. The maximum distance is a+c and the minimum distance is a−c. So in terms of the eccentricity e,

Maximum distance=a+ae=a(1+e)
Minimum distance=a−ae=a(1−e)

And you could write a program that calculated these figures for all the planets.

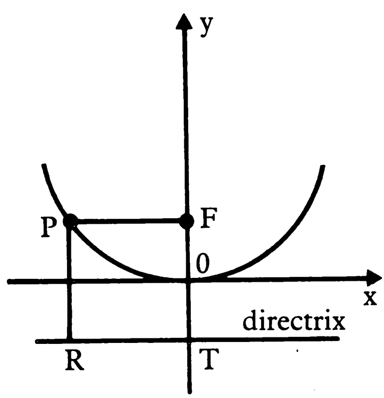
Now for the parabola. It has an eccentricity of 1, and it is also the result of a quadratic. So the general equation of a parabola is

x^2=4py (where p is a constant called the focal distance)

This has a quadratic sort of form, but has a more useful shape for our purposes. We have to rearrange it to get y as the subject, so that

$y=(1/4p)*x^2$

and the diagram is:



This special form of parabola is the standard one. It is defined as the curve given when a point P moves so that its distance to a fixed point F, and a line (called the *directrix*) RT is equal. In symbols, wherever you are on the parabola, and wherever you put point P, this holds true:

RP=PF (where RP is the line through P, and at right angles to the directrix.)

This equation is for the parabola when you draw it on Cartesian coordinates with the point O as the origin. (Note: *not* point T). The focal distance p is equal to OF (so it must also be equal to OT because of the definition of the parabola.) And once again, we have everything we need for a program.

Program 46 PARABOLA

```
1 REM PROGRAM 46
2 REM PARABOLA
10 PRINT ""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 24:X=19:GOSUB1000:
NEXT Y
35 P=3
40 FOR X1=-15 TO 15:X=X1+19
50 LET Y=INT(X1*X1/(4*P)+.5)+12
52 CHAR=81:COL=1:GOSUB 1000
70 NEXT X1
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Experiment with different values of p to get parabolas with different amounts of bend.

The parabola has proved very important to astronomers too, but in a very different way to the ellipse. Astronomers like to use very big telescopes, and the way to make very big telescopes is by using a curved mirror instead of a lens to focus the light rays. This type was first developed by Sir Isaac Newton (the great English genius) and the curve of the mirror is based on a parabola. (A parabola in three dimensions is called a paraboloid.) The parabola has the property of reflecting all the rays parallel to its axis to the focus.

This leaves us with one more conic: the hyperbola. Let's look over a complete list of eccentricities for our conics:

CONIC SECTION	ECCENTRICITY RANGE
Circle	Ø
Ellipse	Ø<e<1
Parabola	1
Hyperbola	e>1

and also look at the general equation of the hyperbola compared with that of the ellipse.

Ellipse

$$\frac{x^2}{a^2}+\frac{y^2}{b^2}=1$$

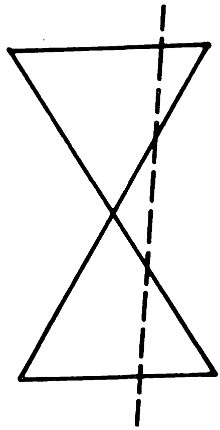
Hyperbola

$$\frac{x^2}{a^2}-\frac{y^2}{b^2}=1$$

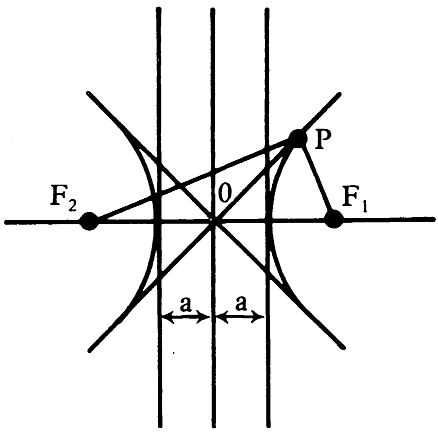
They are similar, the minus sign making all the difference. There are some more interesting differences. First, we have to reconsider the cone at the bottom of all this.

Recall the four cones that were drawn to demonstrate the four conic sections. In the first three, that cone was sufficient, but once you cut at an angle greater than the angle of the cone itself and begin to get hyperbolas, you have to include the *other* cone that is upside down and standing on top of the original cone.

Take a pencil, or something else long and thin, and hold it in the middle with one hand between thumb and index finger. Now grasp the end of the pencil and rotate it in a circle. Each half of the pencil sweeps out an equal and opposite cone surface in the air. Similarly, the hyperbola has *two* curves, because we are looking at the section of two cones. (In the case of the other conic sections, the angle of cut means that the other cone escapes being cut, and there is only one curve.)



And when the hyperbola is cut, the curve is as below:



The two branches of the hyperbola each have a focus. The general equation only holds true if the coordinate system used in drawing a graph of the curve has its origin at O. As you can see, each branch comes in from infinity and goes out to infinity, getting ever closer to one of the diagonal lines (called *asymptotes*) but never actually reaching it.

The hyperbola is defined as a curve which has a constant difference of distances from two fixed points, so that

ABS(PF_2−PF_1)=ABS(PF_1−PF_2)

It must be possible to rearrange the general equation to get y as the subject, and to exploit it for program purposes, but there arises the potential problem of negative square roots.

Starting with the general equation, we have

$$\frac{x^2}{a^2}-\frac{y^2}{b^2}=1$$

Rearranging it,

$$\frac{y^2}{b^2}=\frac{x^2}{a^2}-1$$

and multiplying by b^2 on both sides,

$$y^2=\frac{x^2b^2}{a^2}-b^2$$

and taking out a factor of b^2 on the RHS, to form a bracket,

$$y^2=b^2((x^2/a^2)-1)$$

and finally, taking square roots,

y=±b*SQR((x*x/a*a)−1)

we're all set. Remember, however, that we must avoid situations where the brackets that we're going to take a square root of goes negative. In other words, x^2/a^2 must always remain greater than 1.

Program 47 HYPERBOLA

```
1 REM PROGRAM 47
2 REM HYPERBOLA
10 PRINT ""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FOR Y=0 TO 24:X=19:GOSUB1000:
NEXT Y
35 A=5:B=5
40 FOR X1=-15 TO 15:X=X1+19
45 IF X1*X1/(A*A)<1 THEN 70
50 LET Y1=INT(B*SQR((X1*X1/(A*A)-1))+.5)
```

```
+12
52 Y=Y1:CHAR=81:COL=1:GOSUB 1000
60 LET Y2=-INT(B*SQR((X1*X1/(A*A)-1))+.5
)+12
62 Y=Y2:CHAR=81:COL=1:GOSUB 1000
70 NEXT X1
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

You will find that this program is more sensitive about the values of a and b you can use. But again, your Commodore won't burst into flames if you put in an out-of-range value, so don't be afraid to explore the limits of the program, and play about with the variables so that the program will do what you want it to.

The conic sections come from solid geometry (solid just means 3–D), and find many applications in science and engineering. You could certainly use the parabola in games designing because the parabola is the curve followed by a projectile: a cannon ball, or any other object thrown into the air, moves in a curve that is a parabola.

Remember that the equations we've used to describe the conic sections are only true if you consider their graphs to be drawn on a *Cartesian coordinate system* (like graph paper, with the x-direction at 90° to the y-direction) and with the *origin* of that coordinate system at the centres of circles and ellipses, at the vertex of a parabola or between the two parts of the hyperbola.

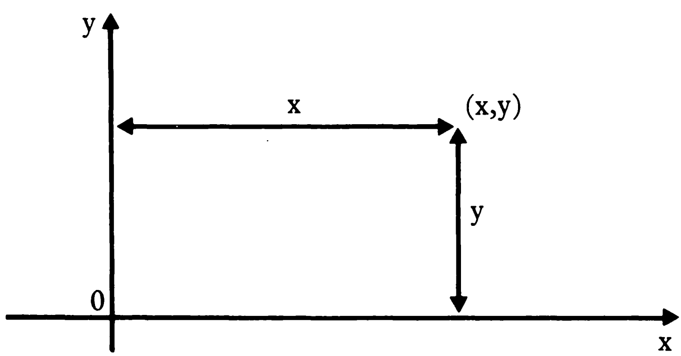
But there are other coordinate systems. One major coordinate system that we are going to have a look at is the one used in radar screens, star maps and maps of Antarctica. Instead of the directions x and y, you have a point of origin with lines radiating out, and concentric circles round it. (Concentric means circles all drawn with their centres in the same place.) Next we will look at polar coordinates.

8

POLAR COORDINATES

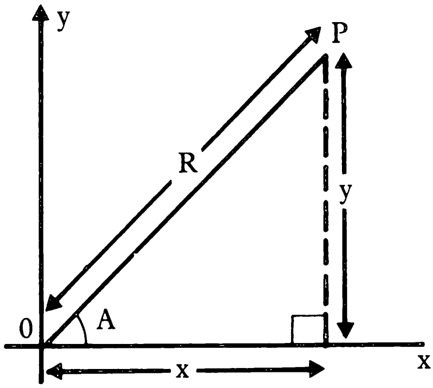
1 Polar Coordinates

If you have a point on a plane and an ordinary set of Cartesian coordinates, then you can call the point (x,y)



Another way of finding the point is to use the *length* of a line from origin (O) to the point and the *angle* that line makes with the x-axis. Any point you like can be specified by x and y values, and also can be specified by length and angle values. It is usual to call these variables R (for length) and θ (for angle), but we must alter the theta, θ, into an ordinary letter that the Commodore will recognise as a variable. Let's call it A (for angle).

The diagram below shows both coordinate systems, because it is a good idea to be able to convert from one to the other.



A conversion can be made by considering the geometry of the situation. We have a right-angled triangle, and so we can use both Pythagoras' Theorem and the trigonometric ratios. It follows that

$$x^2+y^2=R^2$$

or, if you like,

$$R^2=x^2+y^2$$

It also follows that,

$$\text{Tan } A=y/x$$

We can find both R and A if we know x and y. But what if we know R and A, and want to work out x and y? In that case, we need these equations:

$x=R \text{ Cos } A$ (from $\text{Cos } A=x/R$)

and

$y=R \text{ Sin } A$ (from $\text{Sin } A=y/R$

Now we can convert our cumbersome and inelegant equation for a circle (say) into a form far more suited to the computer. This length-and-angle system is called the *polar* coordinate system. Some scientific calculators have buttons marked P and R, which stand for *Polar* and *Rectangular* (meaning the x,y system), so that you can easily convert between the systems.

What such calculators do is employ the equations we've just looked at.

For example, if you take the equation of the circle in rectangular form, you have

$x^2+y^2=R^2$ (Where R is the radius.)

This is easy to convert into Polar form, because the conversion equations automatically give

x=R Cos A
y=R Sin A

Furthermore, all a program has to do is run the angle A round one complete revolution, and since it has to use radians with angles on the computer, the FOR loop we use will have the form

FOR A=Ø TO 2*PI

The problem is that 2*PI is a shade over six, so not many points are plotted unless we step Ø.1 or something small. Stepping at intervals that are multiples of PI plots the points in a symmetrical pattern. Run program 48 to see how superior this polar system is to the rectangular one.

Program 48 POLAR CIRCLE

```
1 REM PROGRAM 48
2 REM POLAR CIRCLE
10 PRINT ""
20 LET R=10:PI=3.1415927
30 FOR A=0 TO 2*PI STEP .15
40 LET X=INT(R*COS(A)+.5)+19
50 LET Y=INT(R*SIN(A)+.5)+12
60 CHAR=81:COL=3:GOSUB1000
70 NEXT A
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

You can alter the radius of the circle by setting R equal to a different value, and you can draw semi-circles or other fractions of circles by changing the limits of the FOR loop. Try for example

```
20 FOR A=PI TO 5*PI/2 STEP PI/50
```

And you can move the centre of the circle just as easily. You can also do ellipses easily, because for ellipses

x=R*COS A   becomes x=aa*COS A

and

y=R*SIN A   becomes y=bb* SIN A

(Because the program uses the variable A for the angle, the semi-major axis normally called a is aa and the semi-minor axis is bb.)

Program 49   POLAR ELLIPSE

```
1 REM PROGRAM 49
2 REM POLAR ELLIPSE
10 PRINT ""
20 LET AA=18:BB=10:PI=3.1415927
30 FOR A=0 TO 2*PI STEP .2
40 LET X=INT(AA*COS(A)+.5)+19
50 LET Y=INT(BB*SIN(A)+.5)+12
60 CHAR=81:COL=3:GOSUB1000
70 NEXT A
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Polar equations can be used for other shapes. A famous shape called the *cardioid* (so called because it's heart-shaped) is found in acoustics, has a polar equation is

R=2a(1−Cos θ)

or in BASIC form,

R=2*aa*(1−COS A)

Program 50 CARDIOID

```
1 REM PROGRAM 50
2 REM CARDOID
10 PRINT ""
20 LET AA=9:PI=3.1415927
30 FOR A=0 TO 2*PI STEP .25
40 LET R=AA*(1-COS(A))
50 LET X=INT(R*COS(A)+.5)+25
55 LET Y=INT(R*SIN(A)+.5)+12
60 CHAR=81:COL=3:GOSUB1000
70 NEXT A
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Now try this:

Program 51 ARCHIMEDEAN SPIRAL

```
1 REM PROGRAM 51
2 REM ARCHEMEDIAN SPIRAL
10 PRINT ""
20 LET AA=2:PI=3.1415927
30 FOR A=0 TO 2*PI STEP .4
40 LET R=AA*A
50 LET X=INT(R*COS(A)+.5)+19
55 LET Y=INT(R*SIN(A)+.5)+12
60 CHAR=81:COL=3:GOSUB1000
70 NEXT A
80 END
```

```
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

The spiral named after the Greek scientist Archimedes (287–212 BC) has the property that the radius arm (the line from a point on the spiral to the centre) is proportional in length to the angle through which it has rotated. The bigger the angle A, the longer the line, R. It can be made easier to see if we make the computer draw the lines in for us. (This can be done with any of our polar graphs quite simply.)

The result is a beautiful pattern (obviously used by Nature to construct fossil shells made even more beautiful by the fact that you now understand it mathematically.

Then there is the *logarithmic spiral* given by the equation,

$$r=ae^{k\theta}$$

or in BASIC language,

r=aa*EXP(k*A)

It is a spiral curve in which the logarithm of the ratio of any two radius arms is proportional to the angle between them.

Program 52 LOGARITHMIC SPIRAL

```
1 REM PROGRAM 52
2 REM LOGARITHMIC SPIRAL
10 PRINT ""
20 LET AA=5:K=0.2:PI=3.1415927
30 FOR A=0 TO 2*PI STEP .4
40 LET R=AA*EXP(K*A)
50 LET X=INT(R*COS(A)+.5)+19
55 LET Y=INT(R*SIN(A)+.5)+14
60 CHAR=81:COL=3:GOSUB1000
70 NEXT A
80 END
```

```
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

It is interesting to run spirals through more than just one revolution (mathematicians use the word '*con*volution' to describe a spiral revolution).

There is nothing to stop you experimenting with polar equations plucked out of the air, trying them in this sort of equation form, and in the sort of program we've been using. You might throw up a few interesting shapes. This one

R=aa*COS(2*A)

gives a butterfly shape, and incidentally draws a circle.

R=8Ø SIN A

Then there's

R=1ØØ*(COS A)*(COS A) (R=100 Cos^2A)

This is the mathematical way of writing Cos A squared too; as you can see it's an infinity sign. Or try

R=3Ø*(1+COS A)

or even

R=3Ø*(1+3*COS A)

or perhaps

R=2Ø*(1+2*SIN A)

then there's

R=8Ø*SIN (3*A)

You can experiment your way to many more spirographic patterns.

2 Curve Sketching

Plotting the graph of a function helps to reveal how it works. Some curves are *periodic*, meaning that they repeat themselves after a certain period (examples are the SIN and COS curves). Is it possible to make graphs of polymials on the computer?

Yes, but they have to be arranged so that they fit the screen, and so that the computer can understand them properly. This means that the figures used to plot the functions must be carefully set up, and the equations put in as strings of x multiplied together instead of as x raised to a power:

x↑2 must be written x*x

and

x↑3 must be written x*x*x

Here is a simple prototype:

Program 53 CURVE SKETCHER

```
1 REM PROGRAM 53
2 REM CURVE SKETCHER
3 REM CHANGE LINE 3000 FOR THE FUNCTION
4 REM THAT YOU WANT OR CHANGE LINE 2000
5 REM AND MAKE THE CONSTANTS DIFFERENT.
6 REM OR DO BOTH !!!!!!
10 PRINT "":GOSUB 2000
12 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B 1000:NEXT X
14 CHAR=66:FOR Y=0 TO 24:X=19:GOSUB 1000
:NEXT Y
20 FOR X=0 TO 39:X1=X-19
30 LET Y=INT(FNA(X1)+.5)
40 CHAR=81:COL=3:Y=Y+12:GOSUB 1000
50 NEXT X
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

```
2000 A=0.01:B=0.01:C=0.01:D=0
3000 DEF FNA(X)=A*X^3+B*X^2+C*X+D
4000 RETURN
```

Try it out and see that you get the straight line you would expect from the equation y=x. (The graph is *stretched* 8 times.)

You can edit line 2ØØØ of the program to use other values of A, B, C and D, and line 3ØØØ to use other equations.

Then you can look at what are called *cubic* equations where there is a term in x cubed:

LET y=(a*x*x*x)+(b*x*x)+(c*x)+d

There are also equations where the highest term is of degree four (called *quartic*); you'll need to define another coefficient, e, in line 2ØØØ and in 3ØØØ DEF FNA(X)=(a*x*x*x*x)+(b*x*x*x) +(c*x*x)+(d*x)+e

And so on. Now that we know what we can do, how do we get useful information out of it? It's important to understand that a cubic equation gives a new shape. Let a=1, b=Ø, c=Ø and d=Ø and have as the equation the general cubic

LET y=(a*x*x*x)+(b*x*x)+(c*x)+d

Run this and see the curve come from the bottom of the screen, zig-zag through the origin and go off the top of the screen. It's as if the negative half of a parabola had been swung down.

Now try this. Let line 3ØØØ become

DEF FNA(X)=a*(x−b)*(x−c)*(r−d)

First the curve crosses the x-axis at three points, and then it crosses it at the points where x=b, x=c and x=d. Furthermore, you can see the effect of making a positive (say 1Ø instead of −1Ø – the curve still cuts the x-axis in three places and those places are the same (because you've done nothing to the coefficients b, c or d) but the curve goes the other way.

The point is that when the curve crosses the x-axis then the value of y is zero, and if the value of y is zero then either a is zero or one of the three brackets is zero. We can see that a is *not* zero, so therefore we can say that either

x−b=Ø (i.e. x=b)

or

$x-c=\emptyset$ (i.e. $x=c$)

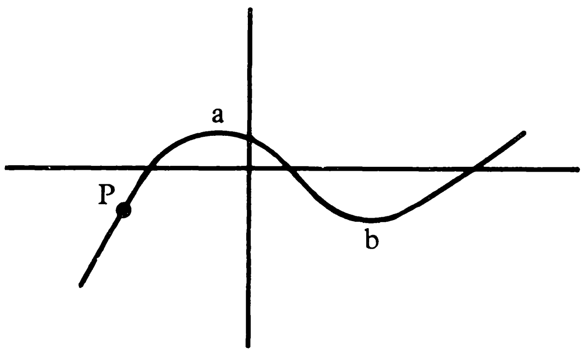
or

$x-d=\emptyset$ (meaning that $x=d$)

So it is possible to extract information from the curve, and if our equation had arisen from a real application, then examining the curve would give us information about that application.

It is a powerful tool in science and engineering, and leads us neatly on to a notion that we will meet a bit later, that of *maxima* and *minima*.

The *calculus* was discovered in the 17th century (independently by both Newton and Gottfried Wilhelm Leibniz (1646–1716), who was falsely accused of plagiarism in a famous scientific controversy). The calculus is a function for the rate at which things change – for example, speed being the rate at which distance changes. First we need a jargon lesson:



This curve has two *turning points*. The turning point at a is a *maximum* because it's going from uphill to downhill, and the turning point at b is a *minimum* because the curve is going from a downhill to an uphill. We say that a curve going uphill has a *positive slope* (which relates to what you already know about linear equations); conversely a curve on its way downhill has a *negative* slope. The slope of the curve at the turning points is zero.

Suppose a point P travels along the curve from left to right, and suppose also that a ruler is put up against the curve so that it only touches it at P. Now the slope of the curve at the point P is the same area as a line just touching the curve at that point (the slope of your ruler). We call that line just touching the curve a *tangent*, and it has absolutely nothing to do with the trig function TAN.

You can see that as the point moves along the curve towards point a the tangent gets more and more horizontal, until when you actually reach point a the tangent is *exactly* horizontal. If P continues to move right along the curve the tangent begins to dip downwards, and then flattens out again, becoming horizontal at b. So at a and b, the tangent (being horizontal) has zero slope. Between a and b, the point where the slope is at a maximum value is called a *point of inflexion*. Now we have to sort out a few other things before proceeding with the calculus.

3 The Solution of Triangles

Let us return to the humble triangle, and discover some equations that help us work them out.

Imagine a triangle made by three pencils, all different lengths. The lengths of the pencils are the lengths of the sides of the triangle, but how do you work out the angles?

Or imagine two pencils at an angle of 30° to each other, how do you work out how long the third pencil would have to be to make a triangle with the other two?

This is how it's done:

If you know three sides $\text{Cos A}=\dfrac{b^2+c^2-a^2}{2bc}$

This formula is known as the *Cosine Rule*, and you just plug in the values of the three sides a, b and c.

Program 54 COSINE RULE

```
1 REM PROGRAM 54
2 REM COSINE RULE
10 PRINT"":REM CLEAR SCREEN
20 INPUT"SIDE A ";AA:PRINT
30 INPUT"SIDE B ";B:PRINT
```

```
40 INPUT"SIDE C ";C:PRINT
50 LET X=(B^2+C^2-AA^2)/(2*B*C)
55 LET ACS=ATN(X/SQR(-X*X+1))
60 LET A=ACS*180/3.1415927
70 PRINT:PRINT:PRINT
80 PRINT "ANGLE OPPOSITE SIDE A"
90 PRINT "IS ";A;"DEGREES"
100 END
```

Test it works by deliberately feeding it an equilateral triangle – one with all three sides the same length, and therefore all three angles 6Ø degrees. You could also test it with a 3,4,5 triangle (which has an angle of 9Ø degrees). You will notice a slight inaccuracy in the calculation, but this is very tiny and could be removed if we required it by using the INT function suitably.

If you know two sides and the angle between them

$b^2=c^2+a^2-2ac \text{ Cos } R$

This is just a rearranged version of the cosine rule into which you plug your sides and angle.

Program 55 COSINE RULE AGAIN

```
1 REM PROGRAM 55
2 REM COSINE RULE AGAIN
10 PRINT"":REM CLEAR SCREEN
20 INPUT"SIDE A ";A:PRINT
30 INPUT"SIDE C ";C:PRINT
40 INPUT"ANGLE B (DEGREES) ";B
50 LET R=B*3.1415927/180
55 LET ACS=ATN(X/SQR(-X*X+1))
60 LET BB=SQR(A^2+C^2-2*A*C*COS(R))
70 PRINT:PRINT:PRINT
80 PRINT "LAST SIDE IS ";BB
90 END
```

Again, it can be tested with an equilateral or ,3,4,5 trinagle, and notice that the program converts the angle to radians so the computer can handle it properly, and that the variable is used to hold side b (so that the computer doesn't confuse it with the angle B). Note also the slight inaccuracy!

If you know one side and two angles

$$\frac{a}{\text{Sin A}}=\frac{b}{\text{Sin B}}=\frac{c}{\text{Sin C}}$$

This equation (or double equation) is called the Sine Rule. The way I've written it is just the shorthand version of:

$$\frac{a}{\text{Sin A}}=\frac{b}{\text{Sin B}}$$

and

$$\frac{b}{\text{Sin B}}=\frac{c}{\text{Sin C}}$$

or

$$\frac{c}{\text{Sin C}}=\frac{a}{\text{Sin A}}$$

If you know two of the three angles you can work out the third angle (all three add up to 180°), and once you know the length of one side the lengths of the other two sides can be got from the Sine Rule.

In *solving* a quadratic we are looking for values of x that will satisfy the equation. In *solving* triangles we are trying to find the values of the three angles and three sides. It is possible to solve all triangles if we know any of the three quantities, using the Sine and Cosine rules.

Program 56 SINE RULE

```
1 REM PROGRAM 56
2 REM SINE RULE
10 PRINT"":REM CLEAR SCREEN
30 INPUT"ANGLE A (DEGREES) ";A
40 INPUT"ANGLE B (DEGREES) ";B
50 INPUT"SIDE C ";CC
55 LET C=180-(A+B):PI=3.1415927
60 LET AA=CC*SIN(A*PI/180)/SIN(C*PI/180)
70 LET BB=AA*SIN(B*PI/180)/SIN(A*PI/180)
80 PRINT:PRINT:PRINT
```

```
90 PRINT "SIDE A IS ";AA
100 PRINT "SIDE B IS ";BB
110 PRINT "ANGLE C IS ";C;"DEGREES"
```

To summarise,

The Cosine Rule

$a^2=b^2+c^2-(2*b*c*\text{Cos } A)$	REMEMBER TO USE RADIANS
$b^2=c^2+a^2-(2*a*c*\text{Cos } B)$	
$c^2=a^2+b^2-(2*a*b*\text{Cos } C)$	USE THIS RULE FOR TWO SIDES AND INCLUDED ANGLE

Notice that the three forms are *cyclic*, so that if you memorize the first version you can get the second and third versions simply by moving the letters on by one (i.e. when you have an *a* make it a *b*, and where you have a letter *b* make it a *c*, and when you have a *c* make it into *a*).

And with a swift bit of rearranging you can get these:

$$\text{Cos } A=\frac{b^2+c^2-a^2}{2bc} \quad \text{REMEMBER TO USE RADIANS}$$

$$\text{Cos } B=\frac{c^2+a^2-b^2}{2ac} \quad \text{USE THIS FORM FOR THREE SIDES.}$$

$$\text{Cos } C=\frac{a^2+b^2-c^2}{2ab}$$

Note again the cyclic aspect.

The *area* of a triangle can be found using geometry but let's just use the result:

Let $s=\frac{1}{2}(a+b+c)$ (where a, b and c are the three sides)

This quantity s is half the sum of three sides, and is known as the *semi-perimeter*.

Then the area of the triangle is given by:

$$\text{Area}=\sqrt{s(s-a)(s-b)(s-c)}$$

So we can solve our triangle to the point where we know all three sides and all three angles, and then we can apply the area equation.

Program 57 AREA OF TRIANGLE

```
1 REM PROGRAM 57
2 REM AREA OF TRIANGLE
10 PRINT"":REM CLEAR SCREEN
30 INPUT"SIDE A ";A
40 INPUT"SIDE B ";B
50 INPUT"SIDE C ";C
55 LET S=(A+B+C)/2
60 LET AREA=SQR(S*(S-A)*(S-B)*(S-C))
70 PRINT:PRINT:PRINT
80 PRINT"AREA IS ";AREA;" SQUARE UNITS"
90 END
```

Several of our formulas seem to have appeared out of thin air. Actually they have been produced by mathematicians over the centuries from *first principles* (that means from basics) by complicated arguments. All this mathematical logic will not be reproduced here, but if you develop a taste for maths the theory is worth looking up.

4 Trigonometrical Equations

Here are a number of results which you will find useful if you come up against equations that contain trig terms. For example

$$\mathrm{Cos}^2x+\mathrm{Sin}^2x=1$$

or in BASIC

((COS x)*(COS x))+((SIN x)*(SIN x))=1

Now you can check that this relationship is true by using program 58. It runs through the values of x from Ø to 2*PI one degree at a time, and evaluates the expression 'cos squared x plus sin squared x' (which ought to be 1).

Program 58 TRIG IDENTITY

```
1 REM PROGRAM 58
2 REM TRIG IDENTITY
5 PI=3.1415927
10 FOR X=0 TO 2*PI STEP PI/180
```

```
20 PRINT ((COS(X)*COS(X))+(SIN(X)*SIN(X)
));TAB(14);X*180/PI
30 NEXT X
```

This is a handy way of testing any relationship over the range in which you want to use it. With trig quantities it follows that if they hold true in the range Ø to 2*PI they hold true for *any* value, because they repeat themselves.

Given the relationship that cos squared x plus sin squared x equals one, we can solve the following trig equation:

Cot x+Tan x=2

Now this is *not* an equation that holds true for all values of x. It's like the situation where we had a quadratic and were trying to solve it; there's a value of x that will satisfy cot x plus tan x equals two and we are trying to find that value of x.

If we start by noting that Cot x=1/Tan x, and that Tan x=Sin x then we can rewrite the equation as:

$$\frac{\text{Cos x}}{\text{Sin x}}+\frac{\text{Sin x}}{\text{Cos x}}=2$$

and if we multiply each term by Sin x and Cos x, we get

(Cos x)*(Cos x)+(Sin x)*(Sin x)=2*(Sin x)*(Cos x)

or

$\text{Cos}^2\text{x}+\text{Sin}^2\text{x}=2\ \text{Sin x Cos x}$

We know from our trig identity that the LHS of the above is equal to 1, so we can substitute:

1=2 Sin x Cos x

Trig identities are so called because they hold true for *all* values of the variable (so they're *identical*). Here's another on

Sin(2*x)=2*Sin x*Cos x

which you can check if you like in the manner of program 58. We can also employ this new identity to push our solution one stage further:

1=2 Sin x Cos x

becomes

$1 = \text{Sin } 2x$

or

$2x = \text{arcsin } 1$

and the angle whose Sin is 1 is 90°, so that

$2x = 90°$

therefore

$x = 45°$

which looks like a solution, and you can check it by substituting it back in the original equation.

It takes practice to get used to trig equations. Here is a list of trig identities that all hold true and which you can use for substitution into any trig equations you come up against.

1 *The three forms of* $\textit{Sin}^2 + \textit{Cos}^2 = 1$

$\text{Sin}^2 + \text{Cos}^2 x = 1$
$\text{Sec}^2 x = 1 + \text{Tan}^2 x$
$\text{Cosec}^2 x = 1 + \text{Cot}^2 x$

You can get any one of these from the first one by careful rearranging.

2 *Addition Formulae*

$\text{Sin }(A+B) = \text{Sin } A \text{ Cos } B + \text{Cos } A \text{ Sin } B$
$\text{Sin }(A-B) = \text{Sin } A \text{ Cos } B - \text{Cos } A \text{ Sin } B$
$\text{Cos }(A+B) = \text{Cos } A \text{ Cos } B - \text{Sin } A \text{ Sin } B$
$\text{Cos }(A-B) = \text{Cos } A \text{ Cos } B + \text{Sin } A \text{ Sin } B$

from which it follows,

$$\text{Tan }(A+B) = \frac{\text{Tan } A + \text{Tan } B}{1 - \text{Tan } A \text{ Tan } B}$$

$$\text{Tan }(A-B) = \frac{\text{Tan } A - \text{Tan } B}{1 + \text{Tan } A \text{ Tan } B}$$

3 Let A=B in the above identities and we get *Double angle formulae*: (A=B=x)

$\text{Sin } 2x = 2 \text{ Sin } x \text{ Cos } x$
$\text{Cos } 2x = \text{Cos}^2 x - \text{Sin}^2 x$
$= 1 - 2 \text{ Sin}^2 x$
$= 2 \text{ Cos}^2 x - 1$

4 *Half Angle Formulae*: Let $\theta = x/2$,

$$\text{Sin } \theta = 2 \text{ Sin}\left(\frac{\theta}{2}\right)\text{Cos}\left(\frac{\theta}{2}\right)$$

$$\text{Cos } \theta = \text{Cos}^2(\theta/2) - \text{Sin}^2(\theta/2)$$
$$= 1 - 2 \text{ Sin}^2(\theta/2)$$

$$\text{Tan } \theta = \frac{2 \text{ Tan}(\theta/2)}{1 - \text{Tan}^2(\theta/2)}$$

5 *Sum and Difference formulae (Factor Formulae)*:

$$\text{Sin A} + \text{Sin B} = 2 \text{ Sin } ((A+B)/2) \text{ Cos } ((A-B)/2)$$
$$\text{Sin A} - \text{Sin B} = 2 \text{ Cos } ((A+B)/2) \text{ Sin } ((A-B)/2)$$
$$\text{Cos A} + \text{Cos B} = 2 \text{ Cos } ((A+B)/2) \text{ Cos } ((A-B)/2)$$
$$\text{Cos A} - \text{Cos B} = 2 \text{ Sin } ((A+B)/2) \text{ Sin } ((A-B)/2)$$

6 *Product Formulae*:

$$\text{Sin A Cos B} = {}^{1/2}(\text{Sin}(A+B) + \text{Sin}(A-B))$$
$$\text{Cos A Sin B} = {}^{1/2}(\text{Sin}(A+B) - \text{Sin}(A-B))$$
$$\text{Cos A Cos B} = {}^{1/2}(\text{Cos}(A+B) + \text{Cos}(A-B))$$
$$\text{Sin A Sin B} = {}^{1/2}(\text{Cos}(A-B) - \text{Cos}(A+B))$$

All this would be too much to remember, but you can look them up any time – the important thing is to know about them. It will help you to see how some trig equations are manipulated if you know these formulae.

So where does the Commodore come in on all this? Look back at how in program 58 (which is only three lines long and probably a lot easier to remember than the trig identities themselves) we used the program to *verify* an identity. Whenever you come up with an alternative form of any of these identities you can test whether you've done it properly by making your computer print out an evaluation of each side of the equation for the whole range from Ø to 2*PI. Program 59 is a general program for verifications of this kind. You just plug one side of your identity into line 4Ø before the comma, and the other side of the identity in after the comma.

Program 59 IDENTITY VERIFIER

```
1 REM PROGRAM 59
2 REM IDENTITY VERIFIER
```

```
5 PI=3.1415927:N=0
10 PRINT ""
20 PRINT "    PRESS F1 TO CONTINUE LIST":
PRINT:PRINT
30 FOR X=0 TO 2*PI STEP PI/180:N=N+1
35 IF INT(N/20)=N/20 THEN GET A$:IF A$<>
"" THEN 35
40 PRINT COS(2*X);TAB(20);(2*COS(X)*COS(
X))-1
50 NEXT X
```

This lets you check one of the double angle formulae in section 3 above.

One of the limits of computation is that you cannot get complete accuracy without an infinite string of decimal places, so you will find that columns generated by program 59 are not absolutely identical, but again it's sufficiently similar for our purposes.

Another advantage of program 59 is that you can check that you have correctly rendered the mathematical forms of the identities into BASIC forms, and got your brackets in the right places.

This approach to mathematics would make a pure mathematician bite off his own head with disapproval, but it's the sort of thing that engineers and scientists do, actually using *numbers*. The complete logical steps from first principles to a final result that make the mathematical process, and would constitute what mathematicians would recognise as a proof, is called the *empirical* approach (from the Greek meaning 'the way of experience'). It is not a bad idea to study the formal approach for the sheer logical beauty of it – but in the meantime ... the computer is, after all, a labour-saving device.

9

COMPLEX NUMBERS

1 Complex Numbers

Although called 'complex', these are easy to understand.

1. a *real* number is a number found on the real line, it can be a fraction, integer, decimal or negative or irrational. Any usual number you know about is a real number.

2. An *imaginary* number, not found on the real line, is any multiple of the square root of minus one.

3. A *complex* number is a two-part number consisting of a real number and an imaginary number.

Consider this quadratic:

$$x^2+x-6=0$$

we get, using our formula and the fact that a=1, b=1 and c=6,

$$x=\frac{-b\pm\sqrt{b^2-4ac}}{2a}$$

so that

$$x=\frac{-1\pm\sqrt{(1*1)-(4*1*-6)}}{2*1}$$

which is

$$x=\frac{-1\pm\sqrt{1-(-24)}}{2}=\frac{-1\pm\sqrt{25}}{2}$$

and therefore

$$x=\frac{-1+5}{2} \text{ and } x=\frac{-1-5}{2}$$

so that

$$x=2 \text{ and } x=3$$

And we see that we have two roots, one of which is positive and one negative. BOTH OF THEM ARE REAL. But now try this quadratic (the discriminant (b^2-4ac) is less than zero:

$$2x^2-x+1=0$$

Here, a=2, b=1 and c=1, so that $(b^2-4ac)=(1-8)=-7$
And when it is substituted into our quadratic solver:

$$x=\frac{-(-1)\pm\sqrt{(-7)}}{4}$$

so that

$$x=\frac{1+\sqrt{-7}}{4} \text{ and } x=\frac{1-\sqrt{-7}}{4}$$

And there it was stuck for centuries, because the square root of a negative number cannot be found, until someone decided to factorise the root whenever it's of a negative number, so that we get

$$\sqrt{-7} \quad =\sqrt{7}\star\sqrt{-1}$$

Then $\sqrt{7}$ can be calculated in the normal way and a letter written next to it to indicate the square root of minus one. So that, going back to our quadratic,

$$x=\frac{1+\sqrt{7}\star j}{4} \text{ and } x=\frac{1-\sqrt{7}\star j}{4}$$

and since root seven is about 2.6, we have:

$$x=\tfrac{1}{4}+j(2.6/4) \text{ and } x=\tfrac{1}{4}-j(2.6/4)$$

Looking at the first root, $\frac{1}{4}$ is a real number, and j(2.6/4)= j(Ø.65) where j represents $\sqrt{-1}$ is an imaginary number, and the whole thing, $\frac{1}{4}$+j(Ø.65) is a number with both real and imaginary parts, a *complex* number.

You will sometimes see the letter i used to represent the square root of minus one, but that is now a bad choice, because since electricity was discovered the current in a wire has been described by the letter i, and in complex number analysis of electric circuits more than one i is definitely not needed.

Here is a program that handles *all* roots of quadratics, be they real or complex. We must make the computer print up the letter j whenever it would normally try to extract the root of a negative number. Take a look back at program 42 and consider program 60 as a development of it.

Program 60 QUADRATIC ALL ROOTS

```
1 REM PROGRAM 60
2 REM QUADRATIC ALL ROOTS
10 PRINT ""
20 PRINT TAB(8)"ROOTS OF QUADRATICS":PRI
NT
30 INPUT "        ENTER COEFFICIENT A ";A:
PRINT
40 INPUT "        ENTER COEFFICIENT B ";B:
PRINT
50 INPUT "        ENTER COEFFICIENT C ";C:
PRINT:PRINT:PRINT
60 PRINT TAB(08);A;"* X^2 +";:GOSUB200
70 PRINT B;"* X +";:GOSUB200
80 PRINT C
90 IF B*B<4*A*C THEN 150
100 LET R1=(-B+(SQR((B*B)-(4*A*C))))/(2*
A)
110 LET R2=(-B-(SQR((B*B)-(4*A*C))))/(2*
A)
120 PRINT:PRINT" FIRST ROOT=";R1
130 PRINT:PRINT" SECOND ROOT=";R2
140 PRINT:PRINT:PRINT:PRINT"
   HIT F1"
145 GET A$:IF A$<>"" THEN 145
146 GOTO 10
150 LET R=-B/(2*A)
160 LET I=SQR((4*A*C)-(B*B))/(2*A)
170 PRINT:PRINT"FIRST ROOT=";R;"+J(";I;"
)"
```

```
180 PRINT:PRINT"SECONDROOT=";R;"-J(";I;"
)"
190 GOTO 140
200 REM DELAY SUBROUTINE
210 FOR T=0 TO 2000:NEXT T
220 RETURN
```

Line 150 makes the computer work out the real part, and line 160 makes it work out the imaginary part's coefficient (which appears in brackets), by first multiplying the discriminant by minus one, so that (b^2-4ac) becomes $(4ac-b^2)$. Lines 170 and 180 then print out the answers, remembering to put in a j symbol to compensate for that trick with the discriminant. Try it with $a=1$, $b=-6$ and $c=34$ and see what you get.

So we can write a general form for our complex number:

$$x+jy$$

where x is any real number and y is any real number. A few examples would be:

$$3+j4$$
$$2-j10$$
$$-5+j0.5$$
$$\pi-j\tfrac{1}{8}$$

and so on.

Now you know what real numbers are, what imaginary numbers are, and that complex numbers have a real and imaginary part. Later we will do arithmetic with complex numbers (that is, add them, subtract them, multiply and divide them) but for now, let's look at our symbol j. It should be clear that

$$j=\sqrt{-1}$$

and that whenever we find a number that is the square root of a minus number, we can do our factorising ruse, so that

$$\sqrt{-64}=j\sqrt{64}=j8$$

or

$$\sqrt{-0.32}=j\sqrt{0.32}=j0.5657$$

But we can do algebra with j itself, since if

$j=\sqrt{-1}$

then, squaring both sides,

$j^2=-1$

and multiplying both sides by j,

$j^3=1\star j=-j$

and again,

$j^4=-1\star j^2=-1\star 1=1$

To summarise:

$j = j$
$j^2=-1$
$j^3=-j$
$j^4= 1$

Higher powers repeat this pattern. Program 61 will print out any power of j for you.

Program 61 POWERS OF j

```
1 REM PROGRAM 61
2 REM POWERS OF J
10 PRINT ""
20 INPUT " ENTER POWER (POS. INT.) ";N
30 IF N-INT(N)<>0 THEN 10
40 IF N/4-INT(N/4)>=0 THEN A$="1"
50 IF N/4-INT(N/4)>=0.25 THEN A$="J"
60 IF N/4-INT(N/4)>=0.5 THEN A$="-1"
70 IF N/4-INT(N/4)>=0.75 THEN A$="-J"
80 PRINT:PRINT:PRINT"    J^";N;"=";A$
90 PRINT:PRINT:PRINT:PRINT" HIT F1"
100 GET A$:IF A$<>"" THEN 100
110 GOTO 10
```

Lines 40 to 70 sort out the power to see if it's a multiple of 1,2,3 or 4, by taking a look at what's left over when you subtract an inteber from a decimal. (Divide any integer by 4 and your number will end in a decimal part which is either .25, .5, .75 or zero.)

Line 3Ø is there to stop you putting in a power that is not an integer.

These powers of j may not seem like much, but it becomes more interesting when you can replace them with whatever power they are equal to. For example, if you come across any j^2 terms, you can replace them with -1.

Now for some arithmetic of the complex number kind. First we have addition. Let's take two complex numbers:

$a+jb$ and $c+jd$

and then let's add them together. All you have to remember it that you add the real parts and imaginary parts separately, so that

$$(a+jb)+(c+jd)=a+c+j(b+d)$$

And subtraction is

$$(a+jb)-(c+jd)=a-c+j(b-d)$$

again treating the real and imaginary parts separately.

Now suppose you have two binomials (expressions involving two terms) and you want to multiply them together – two linear expressions for example:

$$(ax+b)*(cx+d)$$

Then you would multiply each of the terms of the first binomial by each of the terms of the other binomial, drawing in curved lines to illustrate it:

$$(ax+b)*(cx+d)=(acx^2+adx+bcx+bd)$$

Now you can see what happens when you apply this concept to complex numbers (which are of a binomial form in that they have two numbers):

$$(a+jb)*(c+jd)=(ac+jad+jbc+j^2bd)$$

which we can simplify by substituting -1 for j^2

$$(ac+jad+jbc-bd)$$

and collecting together the real terms on the left and imaginary terms on the right

(ac−bd)+j(ad+bc)

This is the product we were after. And it should not be difficult to write small programs that allow us to perform these three complex operations (complex only in the sense that they are operations with so called complex numbers).

Program 62 COMPLEX ADDITION ETC

```
1 REM PROGRAM 62
2 REM COMPLEX ADDITION E.T.C
10 PRINT""
20 PRINT "                    FIRST NUMBER"
30 PRINT:PRINT:INPUT "  REAL PART";R1
40 PRINT:INPUT "  IMAGINARY PART ";I1
50 PRINT:PRINT:PRINT "                    SECOND
 NUMBER"
60 PRINT:PRINT:INPUT "  REAL PART";R2
70 PRINT:INPUT "  IMAGINARY PART ";I2
75 REM WRITE OUT EQUATION
80 PRINT:PRINT:PRINT
90 PRINT " ";R1;"+ J";I1;:GOSUB 200
100 PRINT "PLUS";:GOSUB 200
110 PRINT R2;"+ J";I2;:GOSUB 200
120 PRINT "EQUALS":GOSUB 200
130 PRINT:PRINT
140 PRINT "              ";R1+R2;"+ J";I1+I
2
150 END
200 REM DELAY SUBROUTINE
210 FOR T=0 TO 1500:NEXT T
220 RETURN
```

All you have to do is change line 140 and you have yourself a *Subtraction* variant. (Of course, it makes more sense visually if you replace the word PLUS with the word MINUS...)

Furthermore, following closely the reasoning above regarding the prescribed method of multiplication, we can do this:

```
14Ø PRINT "    ";(R1*R2)-(I1*I2);"+j(";(R1*I1)+
(R2*I1);")"
```

and with the proviso that the PLUSes are altered into TIMESes it becomes a program that multiplies complex numbers. Now, for the division of complex numbers, we need to make an empirical discovery. In program 62, if you play with it long enough, you will get an occasional answer which has its j coefficient zero, and when a complex number has a zero imaginary part, all that's left is a real number.

The condition for the j coefficient to be zero is:

R1*I2+R2*I1=Ø

since the expression on the LHS is the very one we've been using to generate the j coefficient on our program. And if we rearrange the above equation we get

R1*I2=−R2*I1

and finally

R1/R2=−(I1/I2)

If we can find numbers where the above expression holds true, then we've found a pair of complex numbers which when multiplied together gives a simple real number. Here is such an example:

(3+j5) and (3−j5)

This is easier to spot in practice than in theory. One is the same as the other except its connecting sign is − instead of +, and any pair like this when multiplied together gives a real number. The pair is important to us, and we say that one complex number is the *complex conjugate* of another complex number if the above relationship holds true.

Using this complex conjugate idea, you can multiply top and bottom of a complex division by the conjugate of the bottom and reduce it to a real quantity. In other words

$$\frac{(2+j8)}{(3+j4)} \text{ is equal to } \frac{(2+j8)*(3-j4)}{(3+j4)*(3-j4)}$$

and as we've just explained, (3+j4)*(3−j4) is a real number.

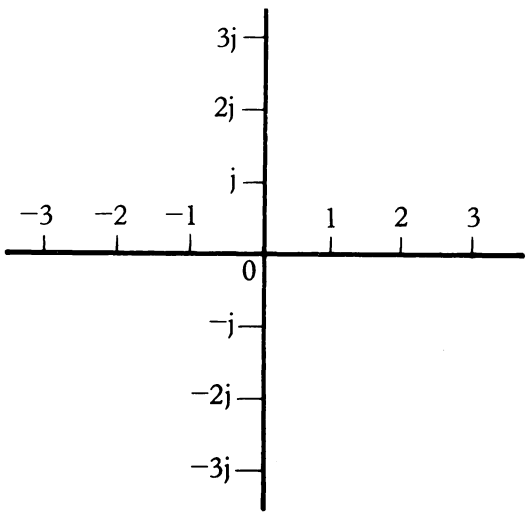
Using program 62 we find that (3+j4)*(3−j4)=25, and (2+j8)*(3−j4)=(38+j16), so the answer is:

$$\left(\frac{38}{25}+j\frac{16}{25}\right)$$

Now for a division program, using yet another alternative line 14Ø. (You should also alter line 1ØØ from PLUS to OVER.)

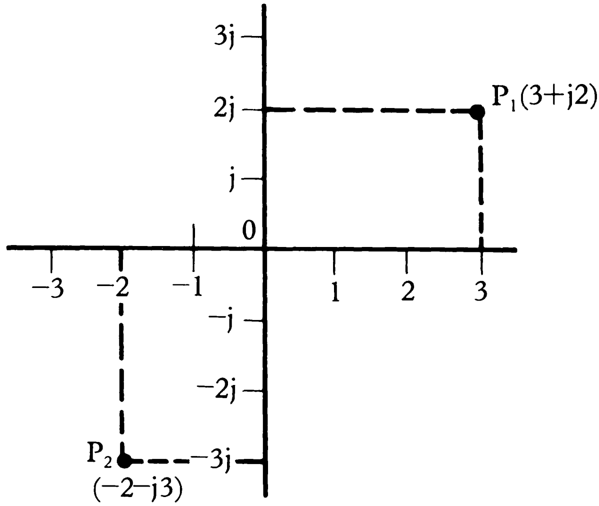
```
14Ø PRINT "   ";((R1*R2)+(I1*I2))/((R2*R2)+(I2*I2));
"+j(";((R2*I1)-(R1*I2))/((R2*R2)+(I2*I2));")"
```

Now we know how to do all the arithmetical things to complex numbers, let us discover the *Argand Diagram*. This is a method of showing complex numbers on a graph. Take the real line and make it into the x-axis of a graph, then use an imaginary line, marked in multiples of j instead of 1, as a y-axis:



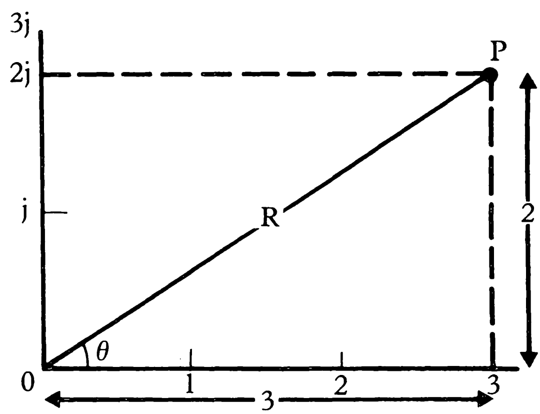
An Argand Diagram

The four quadrants you get here make up what is called the *complex plane*. It's called that because you can represent any complex number on it just by measuring along as far as the real coefficient of the number, then measuring up (or down) as far as the imaginary coefficient. Like this:



Two points have been drawn on the complex plane, P_1 and P_2, which represent two different complex numbers. P_1 represents 3+j2, and P_2 represents −2 −j3.

You can plot any complex number at all on the complex plane, and if you can do it with the Cartesian system you can also represent complex numbers on a polar diagram. Below is a representation of just the top right quadrant of the complex plane, and the number 3+j2.



From this we see that,

$R^2=3^2+2^2$

or

$R=\sqrt{9+4}=\sqrt{13}=3.606$

and

$\text{Tan } \theta=2/3 \text{ or } \theta=\text{Tan}^{-1}(2/3)=33.7°$

In fact it is an alternative form for 3+j2 to be represented 3.606 $\underline{|33.7°}$

If we have a complex number, z, then we can write

$z=a+jb$ (standard form)
$z=r(\text{Cos } \theta+j \text{ Sin } \theta)$ (polar form)

where

$r=\sqrt{a^2+b^2}$

and

$\theta=\text{Tan}^{-1} b/a$

Also,

$a=r \text{ Cos } \theta$ and $b=r \text{ Sin } \theta$

This allows us to interconvert standard to polar and vice versa. We will construct a program to do it, but first there is a third way of expressing a complex number, called the *exponential* form, derived from the polar form. As it is the exponential form, you'd expect it to be e to-the-power-something, and as it is a representation of a complex number, you'd expect it to involve j:

$re^{j\theta}$

where r is the same as it is in polar form, and theta is the angle appearing in the polar form except that it *must be in radians*.

Conversion of polar to exponential and vice versa are easy:

$z=re^{j\theta}=r(\text{Cos } \theta+j \text{ Sin } \theta)$

If for example we have

$z=3(\text{Cos } 45°+j \text{ Sin } 45°)$

it can be written as

$z=3e^{i\frac{\pi}{4}}$

The reason for having all these different forms is that when people had to calculate things by hand it was found to be easier to add and subtract in standard form, easier to multiply and divide using the polar form, and easier to take the logarithm of a complex number if it was expressed exponentially.

For the record, when multiplying complex numbers in polar form you *add the angles* and *multiply the r's* together. For division, we subtract the angles and divide the r's.

Taking the log in exponential form, if

$z=re^{j\theta}$

then

$\log_e z=\log_e r+j\theta$

One of Sir Isaac Newton's friends, a refugee from religious persecution in France, was Abraham De Moivre (1667–1754), who found a relationship known as *De Moivre's Theorem*. If

$z=r(\mathrm{Cos}\theta+j\,\mathrm{Sin}\theta)$

then

$z^n=r^n(\mathrm{Cos}(n\theta)+j\,\mathrm{Sin}(n\theta))$

(De Moivre's theorem works for any value of n, positive, negative or even fractional.)

which helps us to raise complex numbers to a power.

We now come to a useful programming tip, which allows you to choose the number of decimal places in a displayed number and is connected to the INT function.

Suppose you have a number like 45.346578 and you want to quote it to one decimal place. It would be possible to quote it to zero decimal places using INT:

INT(45.346578)

returns 45 (disposing of the unsightly decimal part).

But to get *one* decimal place, you must first *multiply the number by ten* (which causes one place of decimals to hop over the decimal point), *then apply INT* to the number, which in our example would give 453, *then, divide by 10 again* to recover our original number. And it can all be done in one line so that if x=45.346578, PRINT INT((x*1Ø)/1Ø) gives 45.3

Remember that the use of brackets makes sure that first the bracket is computed, then the INT is taken, then the result is divided by 1Ø. Try this:

```
1Ø LET x=45.346578
2Ø PRINT x, INT((x*1Ø)/1Ø)
```

And of course, if you want to get a number to more than one decimal place, you have to multiply by 1ØØ for two decimal places, 1ØØØ for three decimal places, and so on, always remembering that you divide by the same number you multiply by.

You can see it working by doing

2Ø PRINT x, INT((x*1ØØ)/1ØØ) giving 45.34

or

2Ø PRINT x, INT((x*1ØØØ)/1ØØØ) giving 45.346

Notice that this doesn't round the number, but truncates it. You should take care using negative numbers. Since INT −4.3 will return −5, you'd have to overcome that problem by adding one at a suitable point. If

```
1Ø LET x=−45.346578
2Ø PRINT x, INT(((x*1Ø)+1)/1Ø)
```

you get the correct truncation. This digression explains how lines 7Ø and 8Ø in this program work.

Program 63 COMPLEX NUMBER CONVERSION

```
1 REM PROGRAM 63
2 REM COMPLEX NUMBER CONVERTION
5 PI=3.1415927
10 PRINT""
20 INPUT "  REAL COEFFICIENT ";A:PRINT
30 INPUT "  IMAGINARY COEFFICIENT ";B
40 LET R=SQR(A*A+B*B)
50 LET THR=ATN(B/A)
60 LET THD=THR*180/PI
```

```
70 PRINT:PRINT
80 PRINT A;"+ J(";B;")"
90 PRINT:PRINT
100 PRINTINT(R*1000)/1000;" (COS ";INT(T
HD*100)/100;" +J SIN ";INT(THD*100)/100;
")"
110 PRINT:PRINT
120 PRINTINT(R*100)/100;" EXP(J";INT(THR
*100)/100;")"
```

The variables Thd and Thr stand for (and hold) Theta in degrees, and Theta in radians, and the three forms are printed out on the screen for you to examine.

2 Hyperbolic Functions - (The Introduction)

This subject is a bit of a con trick, because you can learn all about hyperbolic functions without ever understanding why they call them hyperbolic functions. They have little to do with hyperbolas.

First, some jargon:

Hyperbolic Function	*Symbol*	*Pronunciation*
Hyperbolic Sine	sinh	'shine'
Hyperbolic Cosine	cosh	'cosh'
Hyperbolic Tangent	tanh	'than'

The h is there after the symbol to remind us that we're dealing with the hyperbolic function and not the normal (circular) functions.

You will find SIN, COS and TAN on the computer but Commodore didn't bother with the hyperbolic functions. That won't stop us. Now a few definitions:

$$\text{Sinh } x = \tfrac{1}{2}(e^x - e^{-x})$$
$$\text{Cosh } x = \tfrac{1}{2}(e^x + e^{-x})$$

and just the same as circular functions,

$$\text{Tanh } x = \frac{\text{Sinh } x}{\text{Cosh } x} = \frac{(e^x - e^{-x})}{(e^x + e^{-x})}$$

So we can define our own functions in terms of the EXP function, and in BASIC we can write:

LET sinhx=((EXP x)−(EXP−x))/2

and

LET coshx=((EXP x)+(EXP−x))/2

and

LET tanhx=((EXP x)−(EXP−x))/((EXP x)+(EXP−x))

Using these definitions over a suitable range we can draw a small graph of Sinh, Cosh and Tanh using programs 64, 65 and 66 respectively:

Program 64 GRAPH OF SINH X

```
1 REM PROGRAM 64
2 REM GRAPH OF SINH X
10 PRINT""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FORY=0 TO 24:X=19:GOSUB 1000:
NEXT Y
40 FOR X1=-3 TO 3 STEP 0.15
50 LET SNHX=(EXP(X1)-EXP(-X1))/2
60 CHAR=81:COL=3:Y=INT(SNHX+.5)+12:X=INT
((X1+3)*6.5+.5):GOSUB 1000
70 NEXT X1
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Program 65 GRAPH OF COSH X

```
1 REM PROGRAM 65
2 REM GRAPH OF COSH X
10 PRINT""
```

```
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FORY=0 TO 24:X=19:GOSUB 1000:
NEXT Y
40 FOR X1=-3 TO 3 STEP 0.15
50 LET CSHX=(EXP(X1)+EXP(-X1))/2
60 CHAR=81:COL=3:Y=INT(CSHX+.5)+12:X=INT
((X1+3)*6.5+.5):GOSUB 1000
70 NEXT X1
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Program 66 GRAPH OF TANH X

```
1 REM PROGRAM 66
2 REM GRAPH OF TANH X
10 PRINT""
20 CHAR=67:COL=1:FOR X=0 TO 39:Y=12:GOSU
B1000:NEXT X
30 CHAR=66:FORY=0 TO 24:X=19:GOSUB 1000:
NEXT Y
40 FOR X1=-3 TO 3 STEP 0.15
50 LET TNHX=(EXP(X1)-EXP(-X1))/(EXP(X1)+
EXP(-X1))
60 CHAR=81:COL=3:Y=INT(TNHX*10+.5)+12:X=
INT((X1+3)*6.5+.5):GOSUB 1000
70 NEXT X1
80 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

With the last graph you will find that the curve begins quite flat and ends quite flat.

It is rare to find hyperbolic function tables these days, and so you might have a use for the next program, which asks you for an x and supplies Sinh x, Cosh x and Tanh x:

Program 67 HYPERBOLIC FUNCTIONS

```
1 REM PROGRAM 67
2 REM HYPERBOLIC FUNCTIONS
10 PRINT""
20 INPUT "  ENTER X ";X
30 PRINT:PRINT:PRINT " IF X EQUALS ";X;"
..."
40 PRINT:PRINT"  SINH X IS     ";(EXP(X)-
EXP(-X))/2
50 PRINT:PRINT"  COSH X IS     ";(EXP(X)+
EXP(-X))/2
60 PRINT:PRINT"  TANH X IS     ";(EXP(X)-
EXP(-X))/(EXP(X)+EXP(-X))
70 PRINT:PRINT:PRINT:PRINT"       HIT F1"
80 GET A$:IF A$<>"" THEN 80
90 GOTO 10
```

Unlike the circular functions which recur every two pi radians, hyperbolic functions do not come round again.

If we continue our parallels with circular functions we'll eventually ask ourselves what the *inverse* of hyperbolic functions are. Remember that the symbol $Sin^{-1}x$ does *not* mean 1/Sin x (because $(Sin\ x)^{-1}$ would mean that), $Sin^{-1}x$ means 'the angle whose Sine is x', so that for example, if

$$Sin\ x=0.5$$

then

$$x=Sin^{-1}\ 0.5$$

Remember that because of the confusion that can arise with power of minus one notation, it was decided to adopt the 'arc' notation, whereby we write x=arcsin 0.5 instead of $x=Sin^{-1}0.5$. The hyperbolic functions have their arc functions too:

arcsinh x or $\sinh^{-1}x$
arccosh x or $\cosh^{-1}x$
arctanh x or $\tanh^{-1}x$

In practice, we would have to use these inverse functions, for example, to find the value whose sinh is 1. Let's follow that particular problem through to see how we might solve it.

We want to find $\sinh^{-1}1$, in other words

$$\text{Sinh } x=1$$

therefore

$$\frac{e^x-e^{-x}}{2}=1$$

so

$$e^x-e^{-x}=2$$

amd multiplying by e^x throughout

$$(e^x)^2-1=2*e^x$$

and rearranging, we get quadratic in e^x

$$(e^x)^2-2(e^x)-1=\emptyset$$

to which we can apply our formula for solving quadratics and find that e^x equals

$$\frac{2\pm\sqrt{4+4}}{2} \qquad (a=1,\ b=2,\ c=1)$$

so that

$$e^x=1+\sqrt{8/2} \text{ or } e^x=1-\sqrt{8/2}$$

so that

$$e^x=2.414 \text{ or } e^x=-0.414$$

and since e^x has to be a positive number, then the first root is the one we're interested in, and taking logs we get 0.881:

$$e^x=2.414 \text{ therefore } x=0.881 \text{ (approx)}$$

We can check this by substituting it back into the original equation, or even simpler, running program 67 with x=

Ø.881373587 (to quote it to great length) and the computer reports back that sinh Ø.881373587 is one. We have found the number whose hyperbolic sine is one, but what a long business it was. We have program 68 to simplify it.

If in the previous argument we had let $y=\sinh^{-1}x$, implying that $x=\sinh y$, then we could have followed the calculation through in general terms, arriving finally at $e^y=x+\sqrt{x^2+1}$, and taking natural logs we'd have

$$y=\log_e(x+\sqrt{x^2+1})$$

and since, $y=\sinh^{-1}x$, we have

$$\sinh^{-1}x=\log_e(x+\sqrt{x^2+1})$$

And you can do the same with cosh and tanh, getting

$$\cos^{-1}x=\log_e(x+\sqrt{x^2-1})$$

and

$$\tanh^{-1}x=½\log_e((1+x)0(1-x))$$

which we can render into BASIC and use to write a program.

Program 68 INVERSE HYPERBOLIC

```
1 REM PROGRAM 68
2 REM INVERSE HYPERBOLIC FUNCTIONS
10 PRINT""
20 PRINT:INPUT " ENTER X";X
30 PRINT:PRINT:PRINT " THE NUMBER X IS "
;X
40 PRINT:PRINT"  ARCSINH X IS    ";LOG(X
+SQR(X*X+1))
50 PRINT:INPUT " ENTER X";X
60 PRINT:PRINT:PRINT " THE NUMBER X IS "
;X
70 PRINT:PRINT"  ARCCOSH X IS    ";LOG(X
+SQR(X*X-1))
80 PRINT:INPUT " ENTER X";X
90 PRINT:PRINT:PRINT " THE NUMBER X IS "
;X
100 PRINT:PRINT"  ARCTANH X IS    ";0.5*
```

```
LOG((1+X)/(1-X))
105 PRINT:PRINT "       HIT F1"
110 GET A$:IF A$<>"" THEN 110
120 GOTO 10
```

Now on to cofunctions. You'd have to look a long time before finding one of these on your calculator, but they do crop up occasionally in calculus so let's have a brief look at them.

We have found that 'reciprocal' means 'one over'. The reciprocal of *four* is one over four, or $\frac{1}{4}$. Similarly, the reciprocal of Tan x is 1/Tan x, or Cot x, and so we have our hyperbolic cofunctions (otherwise known as reciprocal hyperbolic functions):

coth x =1/tanh x	(prounced 'coth')
sech x =1/cosh x	(pronounced 'shek')
cosech x=1/sinh x	(pronounced 'co-shek')

It is enough that you know what these are, and how they relate to the hyperbolic functions. This will allow you to create defintions like

LET cothx=((EXP x)+(EXP–x))/(EXP x)–(EXP–x))
LET sechx=2((EXP x)+(EXP–x))
LET cosechx=2/((EXP x)–(EXP–x))

You could explore them on your own. Meanwhile there are two more revelations to cover concerning hyperbolic functions: first there's the relationship between hyperbolic and ordinary circular trig functions, and then the hyperbolic equivalents of the identities we listed for circular trig functions.

Remember that there are two ways of writing the complex number. After the *standard form* (a+jb) you could have the *exponential form* $re^{j\theta}$, and the *polar form* r(Cos θ+jSinθ), so we can say, if

$$z=r\star e^{j\theta}$$

and

$$z=r\star(\mathrm{Cos}\theta+j\mathrm{Sin}\theta)$$

then it must be true to say

$$r\star e^{j\theta}=r\star(\mathrm{Cos}\theta+j\mathrm{Sin}\theta)$$

and dividing by r on both sides we get

$$e^{j\theta}=\text{Cos}\theta+j\,\text{Sin}\theta$$

It also works out that

$$e^{-j\theta}=\text{Cos}\theta-j\,\text{Sin}\theta$$

(Note the way the minus sign appears on both sides – you might like to spend a few minutes seeing how we get that result from basics.)

The next step (bringing closer the definition of hyperbolic functions) is adding these expressions involving the exponential, and at the same time adding their polar equivalents. We get:

$$e^{j\theta}+e^{-j\theta}=\text{Cos}\theta+j\,\text{Sin}\theta+\text{Cos}\theta-j\,\text{Sin}\theta$$

The terms in $\text{Sin}\theta$ cancel out, leaving

$$\begin{aligned} e^{j\theta}+e^{-j\theta}&=\text{Cos}\theta+\text{Cos}\theta \\ &=2\star\text{Cos}\theta \end{aligned}$$

Remembering, now, our definition of Cosh x,

$$\text{Cosh } x=\frac{e^{x}+e^{-x}}{2}$$

and substituting $j\theta$ for x,

$$\text{Cosh } j\theta=\frac{e^{j\theta}+e^{-j\theta}}{2}$$

But from our arguments above, we have already decided that

$$e^{j\theta}+e^{-j\theta}=2\star\text{Cos}\theta$$

and so that, dividing both sides by 2, we get

$$\frac{e^{j\theta}+e^{-j\theta}}{2}=\text{Cos}\theta$$

which in turn, is equal to $\text{Cosh } j\theta$. We can therefore write:

$$\text{Cos}\theta=\text{Cosh } j\theta$$

which is the first of our relationships. The others can be produced in the same way – try it yourself – but here are the results:

$$j\,\text{Sin}=\text{Sinh } j\theta$$

and

j Tanθ=Tanh jθ

Here is a tabulation and alongside the reverse forms have been tabulated too, allowing you to see Tanh in terms of Tan etc, and the reverse: Tan in terms of Tanh etc:

Sinh jθ=j*Sinθ	Sin jθ=j*Sinh jθ
Cosh jθ=Cosθ	Cos jθ=Coshθ
Tanh jθ=j*Tanθ	Tan jθ=j*Tanhθ

All this means that we can now set about listing the hyperbolic trig identities, or at least a few of them:

HYPERBOLIC TRIG IDENTITIES
$\text{Cosh}^2\theta-\text{Sinh}^2\theta=1$
Sinh 2θ=2*Sinhθ*Coshθ
Cosh 2θ=$\text{Cosh}^2\theta+\text{Sinh}^2\theta$
Cosh(A+B)=CoshA*CoshB+SinhA*SinhB

You will see that if you compare these with the ordinary circular identities, the differences are only of sign- these arise when you have a Sinh^2 involved (either directly or indirectly) with the calculation, because you get an extra factor of j^2 in the working which is, of course, minus one, and will alter the sign.

All this serves only to show you the maths and what it's really all about. There are still a few simple ideas left to examine.

10

THE FACTORIAL

1 The Factorial

Occasionally in maths you will see a number with an exclamation mark after it. This is a factorial.

You can calculate a factorial of a number, if that number is a positive integer, by multiplying together all the other positive integers that are less than it. For example:

factorial three=3!=3*2*1=6

or,

factorial six=6!=6*5*4*3*2*1=720

Let's see a program:

Program 69 FACTORIAL

```
1 REM PROGRAM 69
2 REM FACTORIAL
10 PRINT""
20 PRINT "      HIT F1 TO CONTINUE LIST":
PRINT
30 LET F=1
40 FOR X=1 TO 33
45 IF X/20=INT(X/20) THEN GET A$:IF A$<>
"" THEN 45
50 LET F=F*X
60 PRINT X,F
70 NEXT X
```

This routine will calculate the factorial numbers up to factorial 33. It can't cope with greater than 33 because 34 gives you an error report. It's easy to see why when we look at the speed with which this function takes off. Run program 69 and see that by the time x has reached 13, x! has been able to outstrip the computer's ability to render numbers in ordinary form, and the computer has resorted to scientific notation. By the time it reaches 34 it has outstripped the computer's ability to represent numbers at all. Even a bigger calculator can only manage, say, 69 (which is approximately equal to 1.711224525*10↑98), because it can only show exponents as far as two digits, and 7Ø! is greater than 1Ø↑1ØØ.

In fact such huge numbers are so big that they outstrip the number of things in the universe. You can't use them for counting, but they crop up in various places in maths, especially in *probability*. Think what the odds are against rolling a thousand successive double sixes with the dice.

2 Series

There is yet another way of getting all these functions. All the Sines and Cosines, the exponentials and logs and many others can be got from a string of numbers called a *series*.

The number e that we have been using in the exponential calculations is a curious number. We've already noted that its value is 2.718281828 ... etc but we can calculate it from a series of numbers. Here it is:

$$\frac{1}{0!}+\frac{1}{1!}+\frac{1}{2!}+\frac{1}{3!}+\frac{1}{4!}+\frac{1}{5!}+\frac{1}{6!}\ldots\text{etc}$$

In other words, keep adding a list of the reciprocals of the factorials and you get closer and closer to e. The first term is 1/0!, and 0! is taken to be 1 (*not* zero as you might have thought), so that,

After one term we have as our sum	1
After two terms we have	1+1=2
After three terms we have	$2+\frac{1}{2}=2.5$
After four terms we have	2.5+0.16666etc
	=2.666666666etc

After four terms, we've added 1/4! which is 1/(4*3*2*1) which is 1/24 which is 0.0416666etc bringing our total to:

2.708333333etc

The fifth term amounts to 0.00833333etc, and if we add that to our total we get 2.716666666etc, and so on. Each term adds on a progressively smaller number, and eventually we'd get to our value of 2.718281828... to whatever accuracy we like.

Program 7∅ does it for you automatically so that all you have to do when the program stops with n? is enter 1 the first time, enter 2 the second time, 3 the third time, and so on, until you get to 1∅. The program will do the spadework and give you back approximations to e to 1+n terms. When n=1∅ the computer's calculation of e is put up. The subroutine 1∅∅ to 14∅ you might recognise as a modified form of program 69, which supplies the appropriate factorials.

Program 70 APPROXIMATION TO e

```
1 REM PROGRAM 70
2 REM APPROXIMATION TO E
10 LET T=1
20 FOR A=1 TO 10
30 GOSUB 100
40 LET T=T+1/F
50 PRINT A,T
60 NEXT A
70 PRINT:PRINT:PRINT"    E=",EXP(1)
80 END
100 LET F=1:INPUT"N ";N
110 FOR X=1 TO N
120 LET F=F*X
130 NEXT X
140 RETURN
```

And if our computer let us do it, we'd be able to approximate e to as many places as we wished with this method. As a matter of fact that original series is a series for e^x (e raised to the power x) when x has been set equal to one, and for a general x, we can write

$$e^x = 1 + \frac{x^1}{1!} + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots \frac{x^n}{n!}$$

The symbol at the end of the series, $x^n/n!$ is called the general term and shows us how to calculate any term - set n to increasingly great values to get the terms of the series.

Furthermore, there are other series approximations:

$$\log_e(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \ldots$$

which works when x has a value between -1 and 1; and we also have

$$\log_e(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} - \frac{x^4}{4} - \ldots$$

by substituting $-x$ for x in the first equation.

And there are series for our three trig functions and three hyperbolic functions too. They come from a discovery known as Maclaurin's series, but let's just pluck them out of thin air:

$$\text{Sin } x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} \ldots$$

$$\text{Cos } x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} \ldots$$

$$\text{Tan } = x + \frac{x^3}{3} + \frac{2x^5}{15} + \ldots$$

where x is in radians. Also,

$$\text{Sinh } x = x + \frac{x^3}{3!} + \frac{x^5}{5!} + \frac{x^7}{7!} \ldots$$

$$\text{Cosh } x = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \frac{x^6}{6!} \ldots$$

and there are others. The important thing is that some functions can be expanded into a series, and evaluated for any given value. This is in fact very similar to what the computer does inside its own processor when you tell it to come up with a SIN or an EXP etc. (More precisely, it uses something called a Chebychev polynomial, but never mind that.) The story of how mathematicians came up with these series expansions is interesting, but you have to know a bit of calculus before looking at it usefully. You can replace any occurrence of one of these functions with its series expansion in any program you are using.

3 Progressions

Now we are in a position to play with some sequences of numbers. There are two which are of main importance:

1. Arithmetic Progression
2. Geometric Progression

A progression is simply a sequence of numbers written one after another, so that

1,2,3,4,5,6,...

is a progression of sorts, and so is

2, 4, 6, 8, 10,...

and even

$-1,-3,-5,-7,\ldots$

An *arithmetic* progression is one in which the next number can always be got by adding on a constant number, so that:

2,4,6,8,10,...

is an arithmetic progression because you add *two* to the 1st number to get the second, and you add two to the second number to get the third, and so on. If I were to ask you what the next number in that arithmetic progression is (after 10) you could tell me, because you would know that it must be 10 plus 2, which is 12. Arithmetic progressions are known to mathematicians as APs; geometric progressions are GPs.

In a geometric progression the next number is got by *multiplying* by a constant number. For example:

2, 4, 8, 16, 32, ...

Here you are multiplying by two each time.

The constant number you add each time in an AP is called the *common difference*, and the constant number you multiply by each time in a GP is called the *common factor*. Here's a GP with common factor 3:

3, 9, 27, 81, ...

It doesn't only work with integers. A common difference or common factor can be fractional, giving something like:

1, $1\frac{1}{2}$, 2, $2\frac{1}{2}$, ... (common difference is a half)

which is an AP, or

1, 0.5, 0.25, 0.125, ... (common ratio is 0.5)

which is a GP.

You can write a program that makes your computer generate progressions. Program 71 asks for a 'G' for geometric progression or 'A' for arithmetic progression. Depending on the choice you are asked to specify the common difference or common ratio; then you have to tell the computer where to start the progression – in other words, you have to enter the first term.

Then your machine will list the first few terms of the progression and give you the value of the term you specify as well as the sum of all the terms as far as that term. Play with it:

Program 71 PROGRESSIONS

```
1 REM PROGRAM 71
2 REM PROGRESSIONS
10 PRINT ""
20 INPUT "  ARITHMETIC OR GEOMETRIC ";A$
30 IF A$="G" OR A$="GEOMETRIC" THEN 130
40 PRINT"":PRINT TAB(10)"ARITHMETIC PROG
RESSION"
50 PRINT:INPUT"  ENTER FIRST TERM ";A
55 PRINT"  FIRST TERM IS ";A
60 PRINT:INPUT"  ENTER COMMON DIFFERENCE
 ";D
65 PRINT "  COMMON DIFFERENCE IS ";D
70 PRINT:PRINT"  PROGRESSION IS:"
75 PRINT:PRINT"  TERM NO.    TERM"
80 FOR X=1 TO 10:PRINT TAB(3);X;TAB(14);
A+(X-1)*D:NEXT X
```

```
90 PRINT:INPUT"  ENTER ANY TERM NO. ";N
100 LET NTH=A+(N-1)*D:PRINT"  TERM NO. "
;N;" IS ";NTH
110 LET SN=N/2*(2*A+(N-1)*D):PRINT"  SUM
 TO ";N;" TERMS IS ";SN
120 END
130 PRINT"":PRINT TAB(10)"GEOMETRIC PROG
RESSION"
140 PRINT:INPUT"  ENTER FIRST TERM ";A
150 PRINT"  FIRST TERM IS ";A
160 PRINT:INPUT"  ENTER COMMON RATIO ";R
170 PRINT "  COMMON RATIO IS ";R
180 PRINT:PRINT"  PROGRESSION IS:"
190 PRINT:PRINT"  TERM NO.    TERM"
200 FOR X=1 TO 10:PRINTTAB(3);X;TAB(14);
A*R^(X-1):NEXT X
210 PRINT:INPUT "  ENTER ANY TERM NO. ";
N
220 LET NTH=A*(R^(N-1)):PRINT"  TERM NO.
 ";N;" IS";NTH
230 LET SN=A*(1-R^N)/(1-R):PRINT"  SUM T
O ";N;" TERMS IS ";SN
```

This is a lengthy program, but it is in fact two programs in one. Line 3Ø is concerned with giving you either the first half of the program or the second half. If you choose an AP you get line 8Ø running round its FOR loop to give you the first ten terms of the AP you have specified. Then line 1ØØ will work out the value of any term you like (try asking for the *hundredth* term!) and line 11Ø will tell you the sum of all the terms up to the one you've chosen.

Had you gone for a GP, you would get the terms written out by the FOR loop in line 2ØØ and the other facilities provided by lines 22Ø and 23Ø respectively.

There are, of course, formulas that control these things, and which appear in their BASIC form in the program.

For an AP, the nth term is given by this equation:

nth term=a+(n−1)*d

(If you've never come across 'nth' before, it's a way of saying first, second, third, fourth etc in general terms, so that if n equals six then nth is sixth)

Also, for an AP, we have that the sum to n terms (usually denoted by S_n) is given by:

$$S_n=\frac{n}{2}(2a=(n-1)d)$$

By the same token, for a GP the nth term is given by

$$\text{nth term}=a\star r^{(-1)}$$

and the sum to n terms by

$$S_n=a\star\frac{(1-r^n)}{(1-r)}$$

Again, a proper maths text will tell you how these equations were derived in the first place.

Try an AP where the first term is one and the common difference is also one. If you try it on program 71 you'll get

1, 2, 3, 4, 5, ...

It's the positive integers. And you can rewrite the sum to n terms for this particular case by putting $a=1$ and $d=1$ in our general AP sum to n terms formula:

$$S_n=\frac{n}{2}(2a+(n-1)d)$$

becomes

$$S_n=\frac{n}{2}(2\star 1+(n-1)\star 1)$$

which simplifies to

$$S_n=\frac{n}{2}(n+1)$$

which could be useful in a program sometime. It's a bit like the factorial except that the numbers are added together instead of multiplied together.

What we've been calling S_n or in words 'the sum to n terms' has a proper mathematical notation, which is that instead of using the Roman letter S to stand for sum, the Greek letter S is used; it's called 'sigma' and the capital sigma looks like this:

Σ

The appendix tells you that the user-defined graphic data for our letter sigma (upper case) is:

126, 34, 16, 8, 16, 34, 126, Ø

You can get the mathematicians' form of sum to n terms. The number above sigma shows the maximum number you are summing to, and the number below shows where you're starting from. For example, if we were to write 'the sum of integers from 1 to n', we'd have,

$$\sum_{1}^{n} r = \frac{n(n+1)}{2}$$

which is a result we've established above.

If we wanted to write the 'sum of r squareds as r takes the values from 1 to n', we would write:

$$\sum^{n} r^2 = \frac{n(n+1)(2n+1)}{6}$$

which is a result that you could derive with the help of a maths book. You may have occasion to use this in a program if you get into more advanced programming.

To add a drop more practice, you would write

$$\sum_{1}^{n} r^3 = \left[\frac{n(n+1)}{2}\right]^2$$

which is the sum to n terms of the integers *cubed*:

$1^3, 2^3, 3^3, 4^3, 5^3, \ldots$

which is

1, 8, 27, 64, 125, ...

To test it out, let n=5 in our formula so that

$$\sum_{1}^{5} r^3 = \left[\frac{5(6)}{2}\right]^2 = 15^2 = 225$$

which of course works for any value of n.

This notation of 'the sum from (bottom number) to (top number)' will reappear in a slightly different guise when we get onto what the mathematicians call integration.

4 Sum to Infinity

The progressions we have looked at are sequences of numbers, and they can be as long as we choose to make them. The computer will print out terms until they get too big to handle, because there are an infinite number of terms. We saw that the positive integers were nothing more than an AP with $a=1$ and $d=1$, and there are an infinite number of positive integers. Whatever AP we choose we could never find the sum of all of them, because the sum gets bigger by a bigger amount as we go on. But this is not true of a GP.

Consider the following:

1, 1/2, 1/4, 1/8, 1/16, ...

You can recognise that as a GP with $a=1$ and $r=\frac{1}{2}$. You can also see that the terms are getting smaller as we go on, so that as the number of the term tends to infinity, the term itself tends to zero. (Remember that 'tends to' indicates something that can't be demonstrated.) Mathematicians write a little arrow to mean 'tends to' and they show infinity by this symbol:

$$\infty$$

which looks like a number 8 lying down.

We already know that our sum to n terms of a GP is given by

$$S_n = \frac{a(1-r^n)}{(1-r)}$$

so for our particular series, with $a=1$ and $r=\frac{1}{2}$,

$$S_n = \frac{1(1-\frac{1}{2}^n)}{(1-\frac{1}{2})} = 2(1-1/2^n)$$

and, as $n \to \infty$, $(1/2^n) \to 0$. (As n tends to infinity, $(1/2^n)$ tends to zero.)

We can write the sum to infinity as S_∞, and know that it means the sum of all the terms in the series.

And by substituting infinity for n, and zero for $(1/2^n)$, we have

$$S_\infty = 2(1-0) = 2$$

Which is remarkable because without being able to use infinity we've been able to see what the sum of an infinite GP is. And you can check it out, adding your terms on each time until you reach the limit of the computer's calculating power. Use program 71 to do this. You will see that two is indeed the limit that the sum of the series approaches:

$$S_{10} = 1.9980469$$

and

$$S_{25} = 1.9999999$$

and

$$S_{26} = 2$$

and any number bigger than 26 will yield 2 also.

There's an old riddle based on this, about a frog in a well. The frog is able to jump one metre up on its first jump and (presumably because it gets tired) can manage to jump half as far on its second jump, half as far again on its third jump and so on. The well is two metres deep; how long will it take to jump out? Since it has to jump an infinite number of times to reach two, the poor old frog is stuck for all eternity!

Does this mean that we can always find the sum to infinity of a GP? No, there are clearly some GPs which get bigger and bigger, depending on the value of r. If it is bigger than one then it can't be done, but if it's less than one it can.

Now note that program 71 gets in a twist if you try to enter an r with a negative value. This is too bad, because you can easily think of a GP with a negative r. But then you are trying to raise a negative number to a power (which the computer doesn't want to do). In fact the condition that must be observed if our sum to infinity is going to be a finite number (and not infinity) is that r must lie between one and minus one. In other words the range of r is:

$-1<r<1$

So far as program 71 is concerned we must have,

$0<r<1$

If you think your programming is up to it, why not figure out a program which *does* deal with negative r (it's possible).

5 Convergence

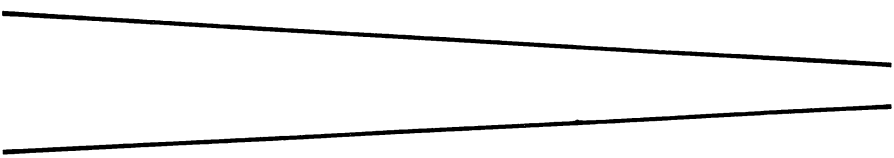
There are series that are neither arithmetic nor geometric progressions. For example, we could have a series which is got from the general term 1/n, by letting n equal successive positive integers:

1, 2, 3, 4, 5, ...

so that our series becomes:

1, 1/2, 1/3, 1/4, 1/5, ...

And if this was a GP (which it is *not*) you would think to yourself that since each terms is getting smaller there would be no trouble finding a sum to infinity:



This is what we mean by *convergence*. If we look at a *real* GP, we find that it converges:

1, 1/2, 1/4, 1/8, 1/16, ...

And clearly the GP

1, 2, 4, 8, 16, ...

must be *divergent*. But what about our example above generated by 1/n? There can be no doubt that each term gets smaller, but that does not mean it is going to be convergent.

If I were to say 'All dogs have four legs', I could not go on to say, 'This animal has four legs, therefore it must be a dog.' It could be a goat or an aardvark. However, it would be true to say 'This animal is a dog, therefore it must have four legs.'

Similarly, simply because a series has terms that tend to zero as the number of the term tends to infinity does not mean that the series necessarily converges. It certainly won't converge if the numbers don't get smaller – but there are some series in which, although the terms get smaller, they don't get smaller fast enough. Our series generated by $1/n$ is an example, and this series is actually divergent.

A test which shows a series that will not converge is called the *comparison test.* It uses a series that is convergent, and compares it with the one to be tested. If the terms of your test series are each less than the corresponding terms of your known series, then your test series will be convergent too. And the test series will be divergent if the corresponding terms are greater than the terms of the known series.

But which known series can we use for comparison? Mathematicians often use this one (which is known to converge):

$$\frac{1}{1^x}+\frac{1}{2^x}+\frac{1}{3^x}+\frac{1}{4^x}+\ldots$$

This series converges if x is greater than one. It diverges if x is less than or equal to one.

So that's the comparison test. There is also *D'Alembert's Test,* named after Jean Le Rond D'Alembert (1717–1783). Some people prefer to call it the *generalized ratio test.*

It is usual to write the terms of a series in general as the letter u with a subscript number, so that u_2 is the first term, u_1 is the second term, u_3 is the third term and so forth. These tests only apply to positive terms. It is perfectly possible to have a series with negative terms, or even alternate terms which are positive and negative, but we're confining ourselves to series with only positive terms.

D'Alembert's ratio test is useful because it involves no other series apart from the one under test. There are no comparisons to be made, except for comparing the terms in the series with their neighbouring terms.

So that, if our series is,

$u_1\ u_2,\ u_3,\ u_4,\ u_5, \ldots$

we can call our general term u_n. And the next one will be u_{n+1}. D'Alembert asks us to make the ratio

$$\frac{u_{n+1}}{u_n}$$

and then find the limiting value as n tends to infinity.

If the limiting value as n tends to infinity is less than one, then the series converges; if it is greater than one then the series diverges. If it is *exactly* one, then the series might converge, or it might diverge.

What is meant 'limiting value'? Suppose you have this series and you want to know if it converges:

$$\frac{1}{2}+\frac{2}{3}+\frac{2^2}{4}+\frac{2^3}{5}+\frac{2^4}{6}+\ldots$$

If u_1 is the first term, and u_2 is the second term, then we can call the *nth* term u_n. The nth term is a general term and can be represented by a formula so that when you substitute a value for n, you get any term you like. If you think about the series we have above, you can see that the nth term is given by the equation,

$$u_n = \frac{2^{n-1}}{1+n}$$

(Try putting n equal to 5, to give the fifth term.) It stands to reason that the term after the nth term is the *n+ th* term, and to get that you just substitute n+1 in the equation for the nth term, like this:

$$u_{n+1} = \frac{2^n}{2+n}$$

And we can therefore get the ratio D'Alembert is remembered for:

$$\frac{u_{n+1}}{u_n}\ \frac{2}{2+n} \div \frac{2^{n-1}}{1+n}$$

which can be written:

$$\frac{u_{n+1}}{u_n}=\frac{2^n}{2+n}\star\frac{1+n}{2^{n-1}}$$
$$=\frac{2(1+n)}{(2+n)}$$

Now, we can divide top and bottom by n, to get

$$\frac{u_{n+1}}{u_n}=\frac{2(\frac{1}{n}+1)}{(\frac{2}{n}+1)}$$

Now comes the idea of the *limit*, because we know that as n tends to infinity 1/n tends to zero, and 2/n also tends to zero. The shorthand version of the phrase 'the limit of D'Alembert's ratio as n tends to infinity' is written,

$$\lim_{n\to\infty}\frac{u_{n+1}}{u_n}=\frac{2(0+1)}{(0+1)}=2$$

so that this limit has been shown to be greater than one. Therefore the series diverges.

The reason for looking at all this is that using a FOR loop we can get the computer to simulate the above process. In program 72, we call the numerator (top part of the fraction) of D'Alembert's ratio uN, and we call the denominator (bottom part of the fraction) of D'Alembert's ratio uD; then we get the computer to print out successive values of the ratio uN/uD as we make n bigger. That way it indicates the limit to which it is tending. It's important to remember that the limit we're looking at is D'Alembert's ratio, not the limit of the series itself.

Program 72 D'ALEMBERT'S RATIO

```
1 REM PROGRAM 72
2 REM D'ALEMBERTS RATIO
10 PRINT""
20 FOR N=0 TO 10 STEP 0.5
30 LET UD=2^(N-1)/(1+N)
40 LET UN=2^N/(2+N)
50 PRINT N,UN/UD
60 NEXT N
```

This program uses the example we've discussed, and by plugging in your general term u_n as uD in line 3Ø and your next term u_{n+1} as uN in line 4Ø it is possible to get D'Alembert's Ratio for any series. Try running program 72 with line 2Ø taking the limit closer by substituting:

```
2Ø FOR N=1Ø TO 1ØØ STEP 1Ø
```

and your results will show you clearly that the limit is indeed 2. Extend the range some more, and eventually the computer will not cope with 2↑n and you get an error report.

6 The Binomial Expansion

The Binomial Expansion is the expansion of a binomial expression – one with just two numbers – which has been raised to a power. It will come up again later, and has many higher applications.

The simplest version is (1+x) raised to the power n, and $(1+x)^n$ has an expansion that goes like this:

$$1+nx+\frac{n(n-1)x^2}{2!}+\frac{n(n-1)(n-2)x^3}{3!}+\ldots$$

The first term is one, the second term is n times x, the third term is n times one-less-than n times x to the power two over two factorial, and so on. The pattern is that each successive term is composed of three elements: the string of n's getting smaller each time all multiplied together, the x raised to a power that gets bigger by one each time, and the denominator which is that power made factorial.

A few observations: When n is positive there are n+1 terms in all. When n is fractional or negative there are an infinite number of terms. The expansion only works when x is greater than −1 and less than plus 1. (i.e. the range of x is $-1<x<1$)

The last by itself would allow us to evaluate something like $(1+\frac{1}{2})^2$ for example (try it and see). But it has much more weighty implications – try a binomial expansion with x set equal to 1/n, so that you have $(1+\frac{1}{n})^n$, and if you play with that long enough you may discover how we get the expansion of e^x that was quoted earlier in this chapter.

A more general binomial expansion is $(a+x)^n$ which you can see is equal to $(a(1+\frac{x}{a}))^n$ and the expansion of that is,

$$a^n\left\{1+n\frac{x}{a}+\frac{n(n-1)}{2!}\left(\frac{x}{a}\right)^2+\frac{n(n-1)(n-2)}{3!}\left(\frac{x}{a}\right)^3+\ldots\right\}$$

As before, when n is positive there are n+1 terms, when n is fractional or negative there are an infinite number of terms. And in this case, the expansion is only valid if (x/a) is in the range from −1 to 1.

Let's look now at a particular expansion of $(1+x)^n$, setting n equal to zero and increasing it by one each time:

$$(1+x)^0=1$$
$$(1+x)^1=1+x$$
$$(1+x)^2=1+2x+x^2$$
$$(1+x)^3=1+3x+3x^2+x^3$$
$$(1+x)^4=1+4x+6x^2+4x^3+x^4$$

and so on ...

Here the brackets on the LHS are multiplied out to get the sequence of powers of x on the RHS. Alternatively, you can think of it in terms of the binomial expansion giving the RHS in each case. Now look carefully at the coefficients of the terms x^0, x^1, x^2, ... and you can write them out in the form of a pyramid:

```
        1
      1   1
    1   2   1
  1   3   3   1
1   4   6   4   1
```

and so on ...

Can you see how it's building up? And can you tell what the next layer of the pyramid would be? Clearly it would be the coefficients from expansion of $(1+x)^5$, and you could work it out from first principles that way, but it's easier to look at the pyramid directly and generate the next row by adding together the two elements directly above to left and right, so that we would get this:

```
  1   4   6   4   1

1   5   10  10  5   1
```

Once you get this pattern it's possible to generate any expansion of $(1+x)^n$, and this fact was first remarked on by the Frenchman Blaise Pascal (1623–1662). In fact the pyramid structure is called Pascal's Triangle.

Program 73 will generate these values for you, labelling them in powers of x down to $(1+x)^{10}$, and you can modify the program to calculate the coefficients of other expansions.

Program 73 PASCAL'S TRIANGLE

```
1 REM PROGRAM 73
2 REM PASCALS TRIANGLE
10 PRINT"":PRINT"X0  X1  X2  X3  X4  X5
 X6  X7  X8"
20 FOR N=0 TO 8
30 FOR R=0 TO N
40 GOSUB 100:GOSUB 200:GOSUB 300
50 PRINTTAB(R*4);FA/(FBR*FR);
60 NEXT R:PRINT
70 NEXT N
99 END
100 FA=1:FOR X=1 TO N:FA=FA*X:NEXT X:RET
URN
200 FBR=1:FOR X=1 TO N-R:FBR=FBR*X:NEXT
X:RETURN
300 FR=1:FOR X=1 TO R:FR=FR*X:NEXT X:RET
URN
```

The structure of Program 73 is that of a main program (lines 1 to 99) with subroutines at 1ØØ, 2ØØ and 3ØØ allowing the necessary calculation. Those subroutines each calculate a factorial, and their structure can be compared with program 69. It rests on the fact that the general term for the coefficients in our expansion is given by the expression

$n!/(n-r)!r!$

7 Fibonacci Series

Leonardo of Pisa (1170–1230) picked up his mathematics from the Arabs, and introduced our present day numbers to Europe (previously Roman numerals and an assortment of Greek letters had been used). But he's also remembered for a device that links together such diverse things as the arrangement of seeds in the centre of a sunflower, the curve of a nautilus shell, Greek architecture and art, Mahler's symphonies, foolscap paper and Alice in Wonderland.

Leonardo Da Pisa was known by his pseudonym 'Fibonacci' and his sequence of numbers goes like this:

1, 1, 2, 3, 5, 8, 13, 21, ...

Try to figure out the next term. If you get 34 then you have correctly seen that each term is the sum of the two previous terms so that 5 is 3 plus 2, and so on.

Program 74 does it for you:

Program 74 FIBONACCI SERIES

```
1 REM PROGRAM 74
2 REM FIBONACCI SERIES
10 PRINT"":PRINT"  TERM NO.          TERM"
20 PRINT"    1             1"
30 PRINT"    2             1"
40 A=1:B=1
50 FOR N=3 TO 21
60 F=A+B
70 PRINTTAB(2);N;TAB(15);F
80 A=B:B=F
90 NEXT N
```

Alter line 3Ø to generate as many terms as you like so that,

3Ø FOR n=3 TO 1ØØ

will give you 1ØØ terms. If we call these terms of the Fibonacci series $u_1, u_2, u_3, \ldots u_n$ then it happens that

$$u_{n+1}^2 = u_n * u_{n+2} + (-1)^n$$

which you can check for n=6:

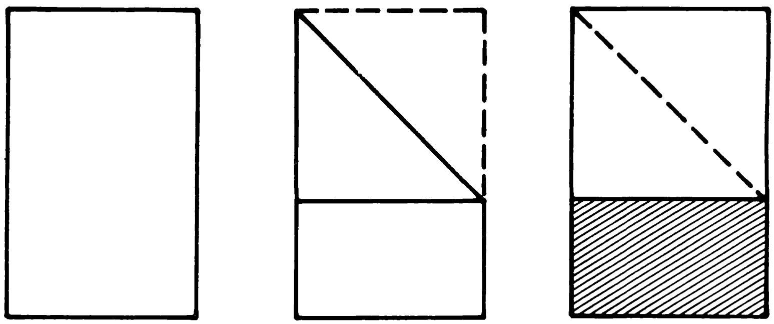
$u_n=8$, $u_{n+1}=13$, $u_{n+2}=21$ and $(-1)^6=1$

so that

$13^2=8*21+1$
$169=168+1$

This was first discovered by a mathematician called Charles Dodgson (1832–1898) who is rather better known for his books, published under the name Lewis Carroll (*Alice in Wonderland*).

Before the days of metric measures, paper in England used to be measured in Imperial sizes, and one size was called *foolscap*. If you bent down one corner of a sheet of foolscap as if you were making a paper aeroplane:



and then bent it back, you would get the crease across the paper making a square, and there would be an oblong left at the bottom shown on the third diagram as shaded. The property of the foolscap size is that the shaded bit left over is the same shape as the original foolscap, only smaller.

We can look at this mathematically. If we call the width of the foolscap sheet 1 unit of length, and the length of it φ (Greek letter phi) units of length, then the ratios of the bit left over are the same as the whole thing. In other words, 1 is to φ as $(\varphi-1)$ is to 1.

In mathematical terms, we can represent these ratios as fractions:

$$\frac{1}{\varphi}=\frac{(\varphi-1)}{1}$$

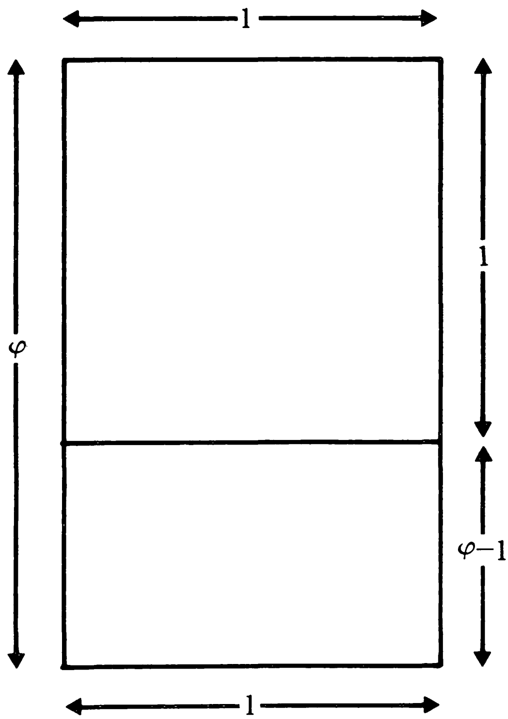
And we can rearrange that and solve for φ, using the quadratic technique

$1=\varphi(\varphi-1)$
$0=\varphi^2-\varphi-1$

so that

$$\varphi = \frac{1 \pm \sqrt{5}}{2}$$

and root five has the value 2.236067977 so φ equals -0.618033988 or φ equals 1.618033989. Since φ is a length, it is



meaningless if it's negative, so we choose the positive value.

This ratio, 1 to 1.618033989, was known to the Greeks and was called the Golden Section. It was a proportion much used in art and architecture.

But it's not obvious what all this has to do with the Fibonacci series. The connection is that the ratios of consecutive terms of the Fibonacci series tend to φ as the number of the term tends to infinity, so that

$$\lim_{n \to \infty} \frac{u_{n+1}}{u_n} = \varphi \quad \text{(Remember D'Alembert?)}$$

The third term is 2, the fourth 3, the ratio equals 3/2=1.5. The thirteen term is 233 and the fourteenth 377, our ratio now equals 377/233=1.618025751 which is much closer. The 31st term over the thirtieth is 1346269/832040=1.618033989 which is correct to 9 decimal places.

The connection with Mahler's symphonies, the spirals of the nautilus shell and sunflower seeds you can discover yourself (here's a clue: look up the word *phyllotaxis* in a good encyclopaedia).

11

VECTORS

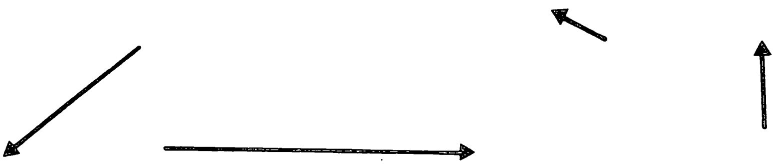
1 Vectors

Almost anything you can measure in numbers is either a *scalar* or a *vector*. A scalar quantity is one that has only size, whereas a vector quantity has *both size and direction*.

An example of a vector quantity is *velocity*. you can have 30 miles per hour due north, or 130 km/h north west, or 1 metre per second along the x-axis, and so on.

An example of a scalar is *mass*, so that if you have a brick of one kilogram, the mass of that brick is not associated with any direction. A kilogram is simply a unit of weight. You can represent scalars by numbers, but for vectors you have to have way of conveying both the magnitude and the direction.

To represent a vector quantity, we draw a line. The *length* of the line represents the *magnitude* part of the vector and the *direction* of the line represents the *direction* of the vector. Two vectors are only equal if *both* their magnitudes and directions are equal. Below is a collection of different vectors:



The reason they are arrows rather than just lines is that a line can point in opposite directions at once, whereas the arrow is definitely pointing in only one direction.

Imagine a ball on the end of a string, and swinging it around your head. Then think of the string running from your fist to the ball as being a vector. Assuming you keep a tight hold on the string you have a constant magnitude (because the length of the string remains constant) but the direction moves continuously.

Program 76 shows 'snapshots' of the string as it makes its way around the full 360°.

Program 75 RADIUS VECTORS

```
1 REM PROGRAM 76
2 REM RADIUS VECTORS
5 PI=3.1415927
10 PRINT""
30 FOR A=0 TO 2*PI STEP PI/8
35 FOR MAG=0 TO 11
40 X=MAG*COS(A):Y=MAG*SIN(A)
50 CHAR=81:COL=3:X=INT(X+19.5):Y=INT(Y+1
2.5):GOSUB 1000
55 NEXT MAG
60 FOR T=0 TO 400:NEXT T:REM DELAY
65 PRINT""
70 NEXT A
99 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

You have to remember that a line is not a vector until you specify its direction, so we can think of the set of radius vectors swept out in program 76 as all being directed from the centre outwards, (or if you like, inwards to the centre).

If instead of string you used elastic, you could get the mganitude of the radius vector to vary as well as its direction, and you could get a set of shapes other than circles. The sun is a fist swinging the ball of the earth around on a string made of gravity and the properties of gravity are such that the shape the earth moves in is an ellipse.

Program 76 RADIUS VECTORS OF AN ELLIPSE

```
1 REM PROGRAM 76A
2 REM RADIUS VECTORS OF AN ELLIPSE
5 PI=3.1415927
10 PRINT""
30 FOR A=0 TO 2*PI STEP PI/8
35 FOR M1=0 TO 15:M2=4*M1/11
40 X=M1*COS(A):Y=M2*SIN(A)
50 CHAR=81:COL=3:X=INT(X+19.5):Y=INT(Y+1
2.5):GOSUB 1000
55 NEXT M1
60 FOR T=0 TO 400:NEXT T:REM DELAY
65 PRINT""
70 NEXT A
99 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

Vectors may be treated like numbers and have arithmetic done on them. You can add them and subtract them and so on, but there are special rules. If we have a vector A, then the vector −A has the same length and direction but points in the opposite direction.

If you have a vector A, and want to add to it a vector B, you end up with a vector C such that if you represent the vectors as lines with the right lengths and angles, the sum vector C will be represented by a line that starts at the beginning of the first vector and ends at the end of the second vector. The triangle that is formed by this process is half of what's called the Parallelogram of vectors. Program 77 will make everything very clear by adding two vectors A and B on and on.

Program 77 ADDITION OF VECTORS

```
1 REM PROGRAM 77
2 REM ADDITION OF VECTORS
```

```
10 PRINT"":PRINT "VECTOR A";
40 CHAR=81:COL=3:FOR X=0 TO 25 STEP 3:Y=
INT(5*X/15+.5):GOSUB1000:NEXT X
50 FOR T=0 TO 2000:NEXT T:REM DELAY
60 PRINT" PLUS VECTOR B";
80 CHAR=81:COL=3:FOR X=25 TO 30:Y=INT(10
*(X-25)/5+10.5):GOSUB1000:NEXT X
90 FOR T=0 TO 2000:NEXT T:REM DELAY
95 PRINT" EQUALS VECTOR C"
100 CHAR=81:COL=3:FOR X=0 TO 30 STEP 3:Y
=INT(20*X/30+.5):GOSUB1000:NEXT X
200 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

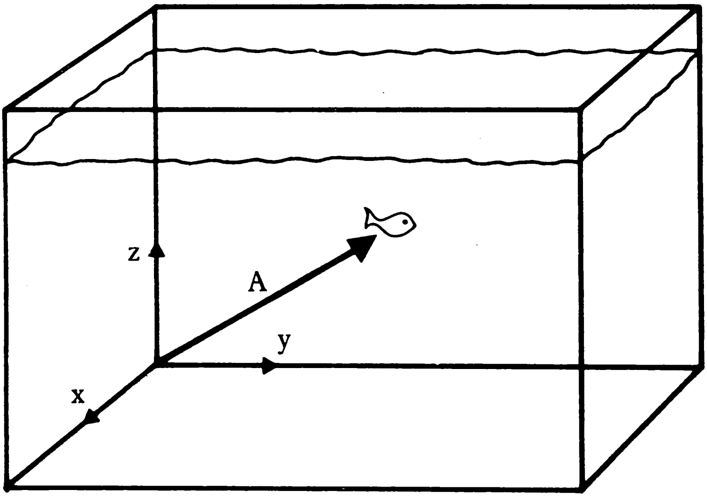
The sum C is called the *resultant*. Other aspects of vector arithmetic are:

Subtraction
We can think of A−B as being A+(−B) and we already know that −B represents B with the same magnitude but exactly opposite direction.

The product of a vector and a scalar
If you multiply a vector A by 10, the effect is to make it ten times as long and leave the direction unchanged. Multiply by a negative scalar and you get the direction turned opposite. (This stands to reason since we know about −A, and that's just A*(−1) after all.)

The null vector
This is just the vector version of zero. If A=B then the null vector is defined as A−B. It has zero magnitude and no definite direction.

One useful application of vector maths is three dimensional mapping. If you think of a fish tank with water in it, and one fish, then decide to call the back left-hand corner of the tank the origin, then you can construct a 3–D coordinate system like the one below:



If we make the width of the tank the x direction, the length of it the y direction, and the height the z direction as shown, then wherever the fish swims, we can imagine a vector (maybe a thin beam of light!) which we can call A, from the origin to the fish. As the fish swims about, A changes. It gets longer and shorter, so the magnitude alters, as well as the direction.

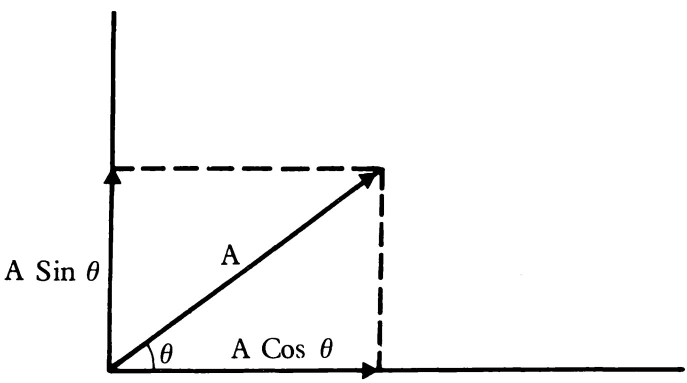
We can break the position of the fish, and hence the vector, into three components along the x,y and z directions, and this process is know as *resolving* the vector into *components*..

Program 78 RESOLVING

```
1 REM PROGRAM 78
2 REM RESOLVING VECTORS
10 PRINT "CHOOSE A VECTOR MAGNITUDE, R"
20 INPUT R
30 PRINT "CHOOSE ANGLE OF VECTOR"
40 INPUT A
45 LET A=A*π/180
50 LET HC=R*COS(A)
```

```
60 LET VC=R*SIN(A)
70 PRINT "HORIZONTAL COMPONENT IS ";HC
80 PRINT "VERTICAL COMPONENT IS ";VC
90 INPUT "PRESS RETURN TO RUN AGAIN";X
100 GOTO 10
```

If you have a vector in three dimensional space you can represent it as three numbers standing for 'so much along', 'so much up', 'so much in'. It also works in situations where there are only two dimensions, and in this case you have two directions in which to resolve the vector. If we set up a 2D coordinate system and draw in a general vector A, the components of A in the x direction and y directions are A*Cosθ and A*Sinθ respectively.

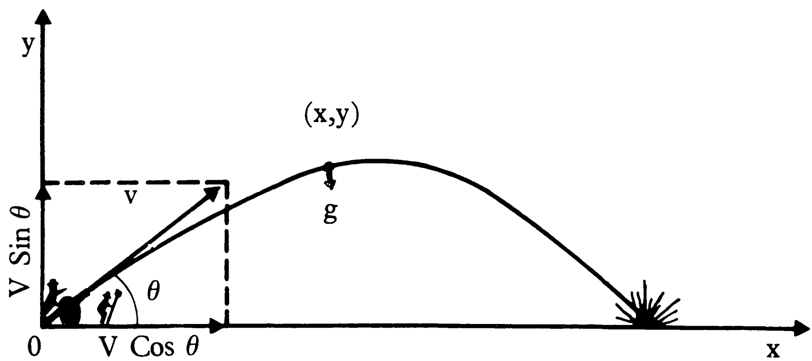


This is fundamentally important in considering the behaviour of projectiles, which means anything thrown and moving under gravity. If we examine the basis for projectiles, we can easily make a computer simulation of the way a cannon ball would move.

Suppose we have an artillery piece firing a ball into the air so that the barrel of the gun is making an angle of θ degrees with the ground. Then the only factor we can vary is the amount of powder charge we use to fire the ball, and the more gun powder the farther the ball will be projected. Because of the way gravity operates, the shape of the curve the ball traces out is a parabola (if we ignore the minor factor of air resistance).

We need to specify the muzzle velocity (the speed at which the ball leaves the barrel, which depends on the charge), and once the ball is airborne there is only one force acting on the ball(gravity).

Since gravity only acts downwards, when we resolve the vectors into directions parallel to and perpendicular to the earth, only the vertical component is affected by gravity. We therefore have a horizontal component that is a constant velocity, and a vertical component that is a constant acceleration in the downward direction.



Suppose your muzzle velocity is v m/s (metres per second) and t represents the time elapsed after firing the gun. Since in the horizontal direction the velocity is constant, the distance of the ball's travel is given by the equation,

$$x=v\star\mathrm{Cos}\theta\star t$$

The vertical motion is more involved and is governed (or at least described) by an equation used by Newton. If we consider the downward acceleration to be called g, then the vertical position y is given by an equation similar to the one that yields x, but with an extra bit to represent the effect of gravity. Newton figured it out to be,

$$y=v\star\mathrm{Sin}(\theta\star t)-gt^2$$

These equations form the basis of program 79. The equation for y gives us the time taken for the ball to arrive, since when it *does* arrive y=Ø, and we can solve the quadratic in t to get

$$\text{time of flight}=\frac{2*v*\text{Sin}\theta}{g}$$

By substituting this back into the equation for x, we can get the horizontal distance covered – the *range*.

$$\text{Range}=\frac{v^2*\text{Sin }2\theta}{g}$$

To illustrate all this, try program 79, which shows it happening graphically, and program 8Ø, which gives you the numbers.

Program 79 PROJECTILES

```
1 REM PROGRAM 79
2 REM PROJECTILES
10 PRINT""
20 INPUT "  MUZZLE VELOCITY ";V
30 INPUT "  ANGLE OF ELEVATION ";A
40 G=10:A=A*3.1415927/180
50 FOR T=0 TO V*SIN(A/G) STEP 0.1
60 LET X=INT(V*T*COS(A))
70 LET Y=INT(V*T*SIN(A)-G*T*T)
80 CHAR=81:COL=3:GOSUB 1000
90 NEXT T
100 END
1000 IF X<0 OR X>39 OR Y<0 OR Y>24 THEN
RETURN
1005 ADR=X+(24-Y)*40
1010 POKE 1024+ADR,CHAR
1020 POKE 55296+ADR,COL
1030 RETURN
```

When the program asks for the muzzle velocity, give it a number between Ø and 1ØØ. Try a variety of angles of elevation, and prove to yourself that a given charge gets the greatest range if the angle is 45°. You could make this into a computer game that simulates an artillery duel between two armies.

But for those who prefer exact numbers:

Program 80 PROJECTILE PARAMETERS

```
1 REM PROGRAM 80
2 REM PROJECTILE PARAMETERS
10 PRINT""
20 INPUT"  MUZZLE VELOCITY (M/S) ";V
30 INPUT"  ANGLE OF ELEVATION (DEGS) ";A
40 LET G=9.81:A=A*3.1415927/180
50 LET TF=2*V*SIN(A/G)
60 LET R=2*V*V*SIN(A)*COS(A)/G
65 LET H=V*V*SIN(A)*SIN(A)/(2*G)
70 PRINT:PRINT:PRINT"   MUZZLE VELOCITY=
";V;" M/S"
80 PRINT:PRINT"   ELEVATION ANGLE=";A*18
0/3.1415927;" DEGS"
90 PRINT:PRINT"   TIME OF FLIGHT=";TF;"
SECS"
100 PRINT:PRINT"   RANGE=";R;" METRES"
110 PRINT:PRINT"   MAX. HEIGHT=";H;" MET
RES"
```

Notice that program 79 has the value 1Ø for g, whereas program 8Ø uses the considerably more accurate value of 9.81 m/s^2. This is known as the acceleration due to gravity and may be regarded as a constant for anywhere on the surface of the Earth. (On Mars of course it is different – 3.6 m/s^2.) It's a function of the mass of the planet and how far you are from the planet's centre, and like all accelerations is measured in the metric system in metres per second squared (m/s^2) also written (ms^{-2}).

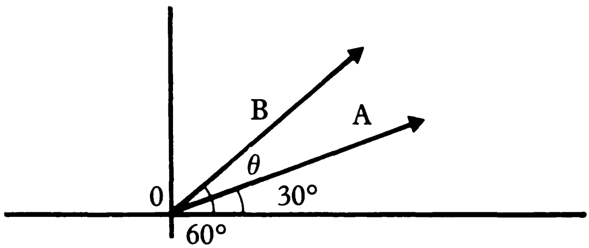
All this theory has a big box of applications attached to it, and if you want to follow this subject up more closely, get a book on the A level applied maths syllabus.

Back to looking at vector arithmetic: let's look at vector multiplication. There are two types of multiplication that can happen to vectors. If A and B are our two vectors, then we can have the *scalar product* or the *vector product*.

Dealing with the scalar product first, it is often called the *dot* product, because in maths books they show the product with a dot. The definition is,

A.B=A*B*Cosθ

where θ is the angle between the vectors. It's called the scalar product because the result is a scalar quantity. Here is an example:



If A has a magnitude of 10, B has magnitude 5, A is 30° up from the horizontal, and B is 60° up. The angle between them is therefore 30°, so from our definition, the dot product is,

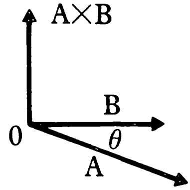
A.B=10*5*Cos 30°
=50*0.866
=43.3

Notice that Cos 90° is zero, so the dot product of two vectors which are at right angles is zero.

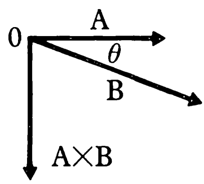
The *vector product* is defined as a vector, and is often called the *cross* product, because it is written with a cross instead of a dot. We define

AxB=A*B*Sinθ

The cross product is not as simple as it might seem, because although the two vectors form a plane, the cross product is at right angles to that plane. neither does AxB=BxA, which is strange enough. In fact, AxB=−(BxA), as we can see from the diagram



If you imagine vectors A and B drawn on a sheet of paper, and to get from A to B you have to rotate anticlockwise, then the vector AxB points upwards out of the paper. If you'd had to rotate clockwise to get from A to B, as below,



then your product vector would point downwards.

The rule turns up, for example, in elementary electricity, when you talk about the magnetic fields that occur when current is put through a solenoid. It's known as the *corkscrew rule* and helps in understanding the cross product. If you take your right hand and curl your fingers round and stick your themb out, then if your fingers are rotating A to B the cross product points in the direction of your thumb. Compare it with the diagrams above and play with it until it sinks in.

In the next program, remember that a minus sign in the answer to a cross product means a downward pointing vector instead of an upward pointing one.

Program 81 VECTOR PRODUCTS

```
1 REM PROGRAM 81
2 REM VECTOR PRODUCTS
10 PRINT""
20 INPUT"  VECTOR A";A:PRINT
30 INPUT"  VECTOR B";B:PRINT
40 INPUT"  ANGLE BETWEEN";TH:PRINT:PRINT
:PRINT
```

```
50 PRINT"   MAGNATUDE OF A =";A
60 PRINT"   MAGNATUDE OF B =";B
70 PRINT"   ANGLE BETWEEN A&B=";TH
75 TR=TH*3.1415927/180
80 PRINT:PRINT"   DOT PRODUCT A.B=";A*B*
COS(TR)
90 PRINT:PRINT"   CROSS PRODUCT AXB=";A*
B*SIN(TR)
100 PRINT:PRINT"   CROSS PRODUCT BXA=";-
A*B*SIN(TR)
```

Remember that the dot product gives a scalar, so that in our program when it says

DOT PRODUCT A.B=7Ø.71Ø678

the number is a magnitude only. In the cross products, the numbers show the magnitudes of the product vectors, but the direction is at right angles to the plane of the vectors A and B, up if it's a positive number and down if it's negative.

2 Matrix Algebra

This ties in with what we've learned, and also illustrates the mathematics of the computer's own array system.

You use the command DIM to dimension an array, and what it does is just the same as what it does in a LET statement. Suppose you've got a variable called x, and you want x to hold the value 1Ø, then you would command

LET x=1Ø

Inside the computer a location called x is created and the value 1Ø is stored in it.

As we have learned, there are numbers which are not just one number at a time, but many at a time. A vector can be thought of as a list of numbers, so that a four vector means a vector composed of four numbers, and a twelve vector means one with twelve separate elements. The computer allows us to work with such vectors.

The way to get the computer to accept a four vector is to tell it

1Ø DIM A(4)

at the beginning of the program. Then the computer will set aside a region of its memory to hold the data for a four vector called A.

Then we can use a FOR loop to read the data into A like this:

```
2Ø FOR n=1 TO 4
3Ø READ A(n)
4Ø NEXT n
5Ø DATA 1Ø,8,14,2Ø
```

If we had another four vector B, so that

B=(12,16,9,5)

you could do the sum

A+B

because you'd know that

A+B=(1Ø+12,8+16,14+9,2Ø+5)
=(22,24,23,25)

All the elements of A combine with the corresponding elements of B and you get a sum which is a four vector.

Now let's suppose a greengrocer wanted to computerise his business, and that one of his stock inventories was four vector A, and another was vector B. Both of these are quantity vectors.

Now suppose he has a price list like this:

apples	1Øp each
oranges	12p each
pears	14p each
bananas	8p each

then he could make a *column* vector (call it P for price), so that

$$P=\begin{pmatrix} 1Ø \\ 12 \\ 14 \\ 8 \end{pmatrix}$$

If our greengrocer wanted to work out the value of his inventory A he would have to multiply the corresponding row vector elements from A with the column vector elements from P, then add all together, so that

$$A{\star}P=(10\ ,8,14,\ 20){\star}\begin{pmatrix}10\\12\\14\\8\end{pmatrix}$$

$$=(10{\star}10)+(8{\star}12)+(14{\star}14)+(20{\star}8)$$
$$=100+96+196+160$$
$$=552 \text{ pence}$$

Notice how the multiplying together of two four vectors has given a single number, 552.

We could also model a scalar multiplication of a vector, by supposing that the greengrocer wanted to carry three times his present stock levels in inventory B. We would have

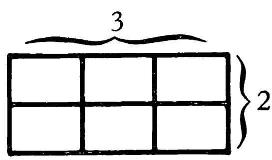
$$3{\star}B=3{\star}(12,16,9,5)$$
$$=(36,48,27,15)$$

This is a quick review of one dimensional number groups, but it's possible on your Commodore to have arrays of more than just one dimension.

If you said,

```
10 DIM A(2,3)
```

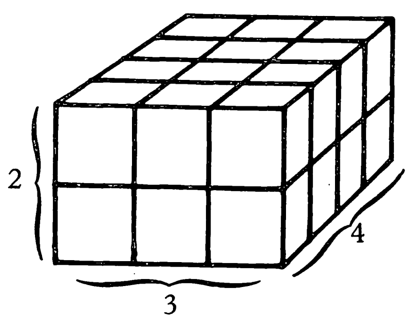
the computer would reserve a block of memory in two dimensions like this:



and if you'd said,

```
10 DIM A(2,3,4)
```

it would make a three dimensional array:



It would be possible to have a four dimensional array or even a ten dimensional one if you liked, but let's confine ourselves to the two dimensional variety.

If a vector is a list of numbers, then a matrix is a bunch of vectors such that you could have a *three by three* matrix, A,

$$A= \begin{pmatrix} 4 & 5 & 1 \\ 2 & \emptyset & 9 \\ 8 & 4 & 3 \end{pmatrix}$$

and this is how to get that inside the computer.

Program 82 THE MATRIX

```
1 REM PROGRAM 82
2 REM THE MATRIX
10 PRINT"":REM CLEAR THE SCREEN
20 DIM A(3,3)
30 FOR R=1 TO 3
40 FOR C=1 TO 3
50 READ A(R,C)
60 PRINT TAB(R*40+C*5+1);A(R,C)""
70 NEXT C
80 NEXT R
90 PRINT TAB(240)"SCALAR MULTIPLIER ";:I
NPUT X
95 PRINT""
100 FOR R=1 TO 3
110 FOR C=1 TO 3
120 LET A(R,C)=A(R,C)*X
130 PRINT TAB(R*40+C*5+20);A(R,C)""
140 NEXT C
150 NEXT R
200 DATA4,5,1,2,0,9,8,4,3
```

The program prints a copy of your chosen matrix, so you can see it has been done properly. It uses r to represent row and c to represent column, and because it's a 2-dimensional array, we have to nest two FOR loops to fill it up. And when it is up on the screen it can be pictured as three column vectors

4	5	1
2	Ø	9
8	4	3

or as three row vectors

4,5,1   2,Ø,9   8,4,3

or as nine independent elements.

If we wanted to add two three-by-three matrices we would just add corresponding elements, so that

$$\begin{pmatrix} 2 & 5 & 1 \\ 1 & 6 & 8 \\ 9 & 3 & 5 \end{pmatrix} + \begin{pmatrix} 4 & Ø & 8 \\ 5 & 5 & 1 \\ Ø & 4 & 6 \end{pmatrix} = \begin{pmatrix} 6 & 5 & 9 \\ 6 & 11 & 9 \\ 9 & 3 & 5 \end{pmatrix}$$

and subtraction is the same – you just subtract corresponding elements.

To achieve a scalar multiplication you would multiply each element of the matrix by a number, so that

$$3\star \begin{pmatrix} 4 & 5 & 1 \\ 2 & Ø & 9 \\ 8 & 4 & 3 \end{pmatrix} = \begin{pmatrix} 12 & 15 & 3 \\ 6 & Ø & 27 \\ 24 & 12 & 9 \end{pmatrix}$$

and a small routine could be written for the end of Program 82 that would allow us to multiply our chosen matrix by any scalar.

Run the whole program, and when it stops enter CONTINUE and it will methodically go through the matrix and multiply each element out for you by whatever scalar you like.

You could adapt program 82 to allow you to handle matrices bigger than 3 by 3. If you didn't insist on displaying the matrix you can handle very big arrays on the Commodore – but you would have to get it to print out the elements in sequence rather than all at once.

Multiplying two matrices together can also be done. First they have to be *conformable*. If you want to multiply matrix A by matrix B you must make sure that *A has the same number of columns as B has rows.*

If it doesn't you multiply each element of the first row by each element of the first column of the other. Then you take the next row of the first matrix and multiply each element of the next column of the other – you add these multiplications up to get the elements of the product, and keep it up until you get to the end.

Here's an example:

$$\begin{pmatrix} a & b & c \\ d & e & f \end{pmatrix}\begin{pmatrix} a & d \\ b & e \\ c & f \end{pmatrix} = \begin{pmatrix} (a*a+b*b+c*c) & (a*d+b*e+c*f) \\ (d*a+e*b+f*c) & (d*d+e*e+f*f) \end{pmatrix}$$

Here's a numerical example:

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{pmatrix}\begin{pmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{pmatrix} = \begin{pmatrix} (1*1+2*2+3*3) & (1*4+2*5+3*6) \\ (4*1+5*2+6*3) & (4*4+5*5+6*6) \end{pmatrix}$$
$$= \begin{pmatrix} (1+4+9) & (4+10+18) \\ (4+10+18) & (16+25+36) \end{pmatrix}$$
$$= \begin{pmatrix} 14 & 32 \\ 32 & 77 \end{pmatrix}$$

The result is a 2 by 2 matrix. Note that the product element formed by the first row and first column goes in the position 1,1 of the final matrix, and so on.

Here is a program that multiplies two two-by-three matrices like those above:

Program 83 PRODUCT OF 2×3 MATRICES

```
1 REM PROGRAM 83
2 REM PRODUCT OF 2X3 MATRICES
10 PRINT""
20 DIM A(2,3):FOR R=1 TO 2:FOR C=1 TO 3:
INPUTA(R,C):PRINTTAB(R*40+C*4);A(R,C)
25 PRINT"":NEXT C:NEXT R
30 DIM B(3,2):FOR R=1 TO 3:FOR C=1TO2:IN
PUTB(R,C):PRINTTAB(R*40+C*4+13);B(R,C)
35 PRINT"":NEXT C:NEXT R
40 C1=(A(1,1)*B(1,1))+(A(1,2)*B(2,1))+(A
(1,3)*B(3,1))
50 C2=(A(1,1)*B(1,2))+(A(1,2)*B(2,2))+(A
(1,3)*B(3,2))
60 C3=(A(2,1)*B(1,1))+(A(2,2)*B(2,1))+(A
(2,3)*B(3,1))
70 C4=(A(2,1)*B(1,2))+(A(2,2)*B(2,2))+(A
(2,3)*B(3,2))
```

```
80 PRINT"":PRINT:PRINT""
90 PRINT TAB(27);C1;C2
100 PRINT TAB(27);C3;C4
```

If you have two square matrices, they are conformable and you can multiply them together. The result is a matrix of the same order. By this I mean that if you multiply two 3X3 matrices you get a 3X3 matrix as the result.

But is there a matrix that you can multiply another matrix by so that you get a result identical with the one you started with? Considering ordinary numbers, any number multiplied by 1 is itself. The matrix version of one, such that any matrix multiplied by it is itself, is known as the *identity* matrix, and for a 3X3 matrix looks like this:

$$\begin{pmatrix} 1 & \emptyset & \emptyset \\ \emptyset & 1 & \emptyset \\ \emptyset & \emptyset & 1 \end{pmatrix}$$

Notice how the elements are all zero except the ones located on the top-left-to-bottom-right diagonal, which are all ones.

For any n*n matrix, there exists an identity matrix such that AI=IA=A, where A is an nxn matrix and I is the identity matrix. And there are also *inverse* matrices. With ordinary numbers you know that

$$a^{-1}*a=1$$

so with matrices

$$A^{-1}*A=I$$

But finding an inverse of a matrix is very complicated, and beyond the scope of this book.

3 Determinants

Remember that simultaneous equations are equations that you solve not one at a time but several at a time. You use them when you can't avoid it. If you've an equation in a variable x, for example,

$$3x-9=\emptyset$$

it can be solved quite simply (the solution is $x=3$). But if an equation has two unknowns, x and y, then it can't unless it has two equations:
Suppose we have

$$3x-5y=6$$

and

$$4x+3y=37$$

The first thing to do is try to eliminate the y term from the pair, and this is done by making the coefficients of y the same. This can be achieved in this particular case by multiplying the top equation by 3 on both sides and the bottom equation by 5 on both sides, giving:

$$9x-15y=18$$

and

$$20x+15y=185$$

If the two equations are then added together we get rid of the y terms and the result reduces to an equation in x alone:

$$29x=203$$

therefore

$$x=203/29=7$$

Armed with this, we can go back to one of our original equations and substitute the value of x we have just found:

$$3*7-5y=6$$

therefore

$$5y=21-6=15$$

so

$$y=3$$

Using two equations in two unknowns and a bit of technique, a solution is found, and it can be checked by substituting $y=3$ and $x=7$ back into the two original equations. But remember that the two equations are independent. That means that they are distinct equations, and that one is not made out of the other by some method. To illustrate, consider this:

$$4x+6y+2=0$$
$$2x+3y+1=0$$

These two equations are *not* independent because the bottom one has been concocted by halving all the coefficients.

In general, n independent simultaneous equations can be solved in n unknowns, so that equations in x, y and z call for three independent equations, and so on.

We can easily do two at a time by computer.

Program 84 SIMULTANEOUS EQUATIONS

```
1 REM PROGRAM 84
2 REM SIMULTANEOUS EQUATIONS
10 PRINT"":REM CLEAR SCREEN &WRITE YELLO
W
20 PRINTTAB(4);"SOLVING SIMULTANEOUS EQU
ATIONS"
30 PRINT:PRINT"          ***FIRST EQUATION*
**"
40 PRINT:PRINT"               Y=AX+B"
50 PRINT:INPUT"             ENTER A";A
60 PRINT:INPUT"             ENTER B";B
70 PRINT:PRINT"          ***SECOND EQUATION
***"
80 PRINT:PRINT"               Y=CX+D"
90 PRINT:INPUT"             ENTER C";C
100 PRINT:INPUT"             ENTER D";D
110 LET X=(D-B)/(A-C)
120 LET Y=A*(D-B)/(A-C)
130 PRINT:PRINT"              X=";X
140 PRINT:PRINT"              Y=";Y
```

To enter the coefficients into program 84, get the equations into the correct form. The example above becomes

$3x-5y=6$ therefore $y=\frac{3x}{5}-\frac{6}{5}$

and

$$4x+3y=37 \text{ therefore } y=\frac{-4x}{3}+37/3$$

you would consequently enter the following:

when asked for coefficient a, enter 3/5
when asked for coefficient b, enter −6/5
when asked for coefficient c, enter −4/3
and when asked for coefficient d, enter 37/3

Enter these fractions and the computer will work out the answers. For example, in the above case, coefficient a doesn't have to be converted into Ø.6 before entering it. Also, it doesn't matter which equation is chosen to be first or second; the program handles either equation first.

This could be extended to many equations, but it would also be tedious to work through several simultaneous equations.

The determinant is a way to automate the process. It looks like a matrix but is more fun. Whereas a matrix is usually written enclosed in ordinary curved brackets, determinants are enclosed in straight lines, like this:

$$\begin{vmatrix} 3 & 5 \\ 4 & 2 \end{vmatrix}$$

But for the moment the trouble is that program 84 requires so much algebra before it can be used. Let's look for a version that will deal with the coefficients *as they are presented* without the need to convert them.

The usual form of sumultaneous equations is

$$1_1x+b_1y+d_1=\emptyset$$
$$a_2x+b_2y+d_2=\emptyset$$

so that

$$x=\frac{b_1d_2-b_2d_1}{a_1b_2-a_2b_1}$$

and

$$y=\frac{-a_1d_2-a_2d_1}{a_1b_2-a_2b_2}$$

Notice that the denominator in these two equations is now the same, being $a_1b_2-a_2b_1$, and so long as this does not equal zero the program will work. Look out for our 'warhead' in equation form in lines 13Ø and 14Ø, and see how line 12Ø intercepts a zero denominator.

Program 85 SIMULTANEOUS EQUATIONS AGAIN

```
1 REM PROGRAM 85
2 REM SIMULTANEOUS EQUATIONS AGAIN
10 PRINT"":REM CLEAR SCREEN &WRITE YELLO
W
20 PRINTTAB(4);"SOLVING SIMULTANEOUS EQU
ATIONS"
30 PRINT:PRINT"          ***FIRST EQUATION*
**"
40 PRINT:PRINT"              A1X+B1Y+D1=0":
PRINT
50 INPUT"          ENTER A1";A1
60 INPUT"          ENTER B1";B1
65 INPUT"          ENTER D1";D1
70 PRINT:PRINT"          ***SECOND EQUATION
***"
80 PRINT:PRINT"              A2X+B2Y+D2=0":
PRINT
90 INPUT"          ENTER A2";A2
100 INPUT"          ENTER B2";B2
105 INPUT"          ENTER D2";D2
110 LET D=A1*B2-A2*B1:IF D<>0 THEN 150
120 PRINTTAB(08)"DENOMINATOR IS ZERO"
130 PRINTTAB(05)"TRY DIFFERENT COEFFICIE
NTS"
140 FOR T=0 TO 3000:NEXT T:GOTO 10
150 LET X=(B1*D2-B2*D1)/D
160 LET Y=(A1*D2-A2*D1)/D
170 PRINT:PRINTTAB(12);"X=";X
180 PRINT:PRINTTAB(12);"Y=";Y
```

Try it with our original pair of equations, so that

a1= 3 and a2= 4
b1=−5 and b2= 3
d1=−6 and d2=−37

These are the forms dealt with in this subject; now we can define a determinant. Instead of writing 'something times something minus something times something', a simple determinant is shorthand for it, so that

$$\begin{vmatrix} a & b \\ c & d \end{vmatrix}$$

is equivalent to writing a*d−c*b. The solution to our simultaneous equations is determinant form looks like this:

$$x=\frac{\begin{vmatrix} b1 & d1 \\ b2 & d2 \end{vmatrix}}{\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix}}$$

and

$$y=-\frac{\begin{vmatrix} a1 & d1 \\ a2 & d2 \end{vmatrix}}{\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix}}$$

The Greek letter D is delta (Δ) and we use it to show a determinant.

So we can write

$$x=\frac{\Delta 1}{\Delta 0}$$

$$y=-\frac{\Delta 2}{\Delta 0}$$

where $\Delta 1=\begin{vmatrix} b1 & d1 \\ b2 & d2 \end{vmatrix}$

$\Delta 2=\begin{vmatrix} a1 & d1 \\ a2 & d2 \end{vmatrix}$

$\Delta 0=\begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix}$

To solve this,

$$6+3y-33=0 \quad a1=6 \quad b1=3 \quad d1=-33$$
$$13x-4y-19=0 \quad a2=13 \quad b2=-4 \quad d2=-19$$

we can use determinants. program 85 or even regular school methods will show that x=3 and y=5, and using determinants we can show the same thing:

$$\frac{x}{\Delta 1} = -\frac{y}{\Delta 2} = \frac{1}{\Delta 0}$$

and since

$$\Delta 1 = \begin{vmatrix} b1 & d1 \\ b2 & d2 \end{vmatrix} = \begin{vmatrix} 3 & -33 \\ -4 & -19 \end{vmatrix} = -189$$

$$\Delta 2 = \begin{vmatrix} a1 & d1 \\ a2 & d2 \end{vmatrix} = \begin{vmatrix} 6 & -33 \\ 13 & -19 \end{vmatrix} = 315$$

$$\Delta 0 = \begin{vmatrix} a1 & b1 \\ a2 & b2 \end{vmatrix} = \begin{vmatrix} 6 & 3 \\ 13 & -4 \end{vmatrix} = -63$$

therefore

$$x = \frac{189}{63} \text{ and } y = \frac{315}{63}$$

Determinants come into their own when dealing with large numbers of equations at once. Apart from being a pwerful tool in the solution of linear equations, they can be employed in the manipulation of matrices, and the determinant of any square matrix can be taken. It has the same elements and is the same size, and the value of it tells us things about the matrix. For example, if the determinant of a matrix turns out to be zero then we say that the matrix is *singular*.

This program evaluates determinants of order two.

Program 86 DETERMINANT EVALUATION

```
1 REM PROGRAM 86
2 REM DETERMINANT EVALUATION
10 PRINT""
20 DIM A(2,2)
30 FOR R=1 TO 2
```

```
40 FOR C=1 TO 2
50 PRINT"":PRINT"
        ":PRINT""
52 INPUT" ENTER NEXT ELEMENT ";A(R,C)
60 PRINTTAB(R*80+C*4+11);A(R,C)
70 NEXT C
80 NEXT R
90 LET D=A(1,1)*A(2,2)-A(2,1)*A(1,2)
100 PRINT:PRINT:PRINT:PRINT:PRINT
110 PRINTTAB(5)"VALUE OF DETERMINANT IS
";D
```

Plug in the values row by row, element by element, and it gives back the numerical equivalent.

If there are three simultaneous equations, like these,

$$a_1x=b_1y+c_1z+d_1=\emptyset$$
$$a_2x+b_2y+c_2z+d_2=\emptyset$$
$$a_3x+b_3y+c_3z+d_3=\emptyset$$

determinants or order three are used. As before, the solution boils down to this:

$$\frac{x}{\Delta 1}=\frac{-y}{\Delta 2}=\frac{z}{\Delta 3}=\frac{-1}{\Delta\emptyset}$$

$$\Delta 1=\begin{vmatrix} b_1 & c_1 & d_1 \\ b_2 & c_2 & d_2 \\ b_3 & c_3 & d_3 \end{vmatrix}\Delta 2=\begin{vmatrix} a_1 & c_1 & d_1 \\ a_2 & c_2 & d_2 \\ a_3 & c_3 & d_3 \end{vmatrix}\Delta 3=\begin{vmatrix} a_1 & b_1 & d_1 \\ a_2 & b_2 & d_2 \\ a_3 & b_3 & d_3 \end{vmatrix}\Delta 0=\begin{vmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{vmatrix}$$

It's easier to remember how to arrive at these results than it seems, because for example Δ_1, which is associated with the variable x, has elements of all the coefficients except those of the x terms in our original equations – in other words, Δ_1 consists of b's c's and d's but not a's.

To solve a three matrix you have to use a special method. If our determinant is

$$\begin{vmatrix} A & B & C \\ D & E & F \\ G & H & I \end{vmatrix}$$

then the value of it is

A times the determinant left after removing the terms in the row and column occupied by A.
minus B times the determinant left after removing the terms in the row and column occupied by B.
plus C times the determinant left after removing the terms in the row and column occupied by C.

which is,

$$A\star \begin{vmatrix} E & F \\ H & I \end{vmatrix} - B\star \begin{vmatrix} D & F \\ G & I \end{vmatrix} + C\star \begin{vmatrix} D & E \\ G & H \end{vmatrix}$$

which is solvable since we can evaluate these order two type determinants.

Now try program 87, which does it all. Notice that it incorporates a subroutine that does the work of program 86.

Program 87 ORDER 3 DETERMINANT

```
1 REM PROGRAM 86
2 REM THIRD ORDER DETERMINANT
10 PRINT""
20 DIM A(3,3)
30 FOR R=1 TO 3
40 FOR C=1 TO 3
50 PRINT"":PRINT"
        ":PRINT""
52 INPUT" ENTER NEXT ELEMENT ";A(R,C)
60 PRINTTAB(R*40+C*4+08);A(R,C)
70 NEXT C
80 NEXT R
90 LET D1=A(2,1)*A(3,3)-A(3,1)*A(2,3)
100 LET D2=A(1,1)*A(3,3)-A(3,1)*A(1,3)
110 LET D3=A(1,1)*A(2,3)-A(2,1)*A(1,3)
120 LET D=-A(1,2)*D1+A(2,2)*D2-A(3,2)*D3
130 PRINT:PRINT:PRINT:PRINT:PRINT
140 PRINTTAB(5)"VALUE OF DETERMINANT IS
";D
```

To test it out on some order three determinants, try these:

$$\begin{vmatrix} 2 & 2 & 2 \\ 3 & 1 & 1 \\ 4 & 2 & 1 \end{vmatrix} = 4; \quad \begin{vmatrix} 4 & 1 & 5 \\ 8 & 2 & 6 \\ 12 & 3 & 7 \end{vmatrix} = \emptyset; \quad \begin{vmatrix} 4 & 3 & 5 \\ 1 & 6 & 1 \\ 7 & -2 & 8 \end{vmatrix} = -23$$

And if you're really adventurous, you could devise a program that evaluates a determinant of order four, by extending with rigid logic the ideas already discussed.

12

DIFFERENTIATION

1 Rates of Change

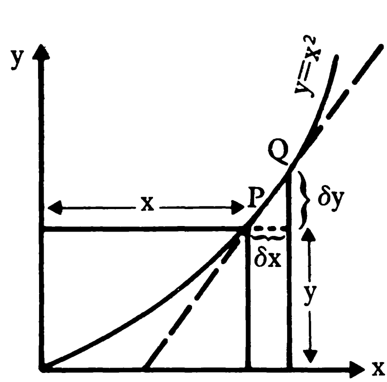
In Latin, the word *calx* means 'stone'. And it has a diminutive, a word for a little stone, or a pebble: *calculus*. Calculus is a major branch of mathematics, and the name is derived from the use of stones or pebbles as counters.

Discovered independently by Newton and Leibnitz in the 17th century, calculus is generally regarded as being of two kinds: *integral calculus* (which deals with a process called *integration*) and *differential calculus* (which deals with the process known as *differentiation*). In this chapter we will have a look at differentiation, and in the next, integration.

This section is entitled 'Rates of Change', and that's really what differentiation is about. The easiest way of looking at it is by considering the familiar ideas of distance, velocity and acceleration.

Consider this notion, which is at the very root of calculus. Exploring the concept of series, we discovered quantities that got smaller and smaller, tending to zero, so that some function of that quantity tended to a given value. The idea is called finding the limit.

Now imagine one of the simplest functions, with its attendant graph, $y=x^2$



Think of two points on the curve, P and Q, and connect them together by a line. Suppose that the horizontal separation of P and Q is called δx (using the small Greek d and read as 'delta x'. The small delta is used in mathematics to mean a small amount so delta x means a little bit of x. Mathematicians will call it a small *increment* in x).

Suppose also that the vertical separation is called δy. Then it follows that the slope of the line from P to Q is $\delta y/\delta x$. If we want to find the slope of the curve at any one point (which is the same as the slope of the tangent to the curve at that point), P and Q can be considered sliding along the curve until they merge. Then the slope of the curve at the point occupied by both P and Q will be the solution.

This is done mathematically by sliding Q towards P so that δy and δx both get smaller, until *in the limit* P and Q coincide. Instead of $\delta y/\delta x$ the limiting slope is written dy/dx (with English instead of Greek letters). In fact, it is no longer a ratio like $\delta y/\delta x$, but a limit of a ratio, which is a single thing with the meaning of 'the slope of the curve at that point'.

In our particular example we can work out what that slope is. If y is a function of x, then dy/dx will be a function of x too:

At P, $y=x^2$
At Q, $y+\delta y=(x+\delta x)^2=x^2+2x\delta x+(\delta x)^2$

If we take y off the LHS and y off the RHS (remembering that $y=x^2$), then we get

$\delta=2x\delta x+(\delta x)^2$

and to get our slope ratio, we divide both sides by δx

$$\delta y/\delta x=2x+\delta x$$

If we start sliding P and Q so that they coincide, we have to take the limit as δx tends to zero, and as δx tends to zero, $2x+\delta x$ tends to 2x. We can therefore write

$$dy/dx=2x$$

We can find the slope of $y=x^2$ at any point by just substituting into the equation

$$\text{slope}=dy/dy=2x$$

This can be extended to other functions too. For example, if we have any polynomial in x, or any power of x, we can get the slope dy/dx by taking each term of the polynomial separately, multiplying the term by the power of x, and reducing the power by one. The equation $y=x^2$ obeys this.

We call dy/dx the *differential* or the *derivative* of y, and we call the process of finding the derivative *differentiation*. The branch of maths dealing with differentials is called *differential calculus*, and equations containing terms that are derivatives are called *differential equations*.

Let's look at that formula for finding the differential of any function:

If $y=a \star x^n$
then $dy/dx=n \star a \star x^{(n-1)}$

If we try it with $y=x^2$,

$n=2$ and $a=1$,
therefore $dy/dx=2 \star x^1=2x$

and if we try it with $y=5x^3$,

$n=3$ and $a=5$,
therefore $dy/dx=3 \star 5 \star x^{(3-1)}=15x^2$

Here is a program that does it:

Program 88 DIFFERENTIATION OF POLYNOMIAL TERMS

```
1 REM PROGRAM 88
2 REM DIFFERENTIATION OF POLYNOMIALS
10 PRINT""
20 PRINTTAB(4)"DIFFERENTIATION OF POLYNO
MIALS"
30 PRINT:PRINT
40 PRINT"                    N
N-1"
50 PRINT"       IF Y=A*X   THEN DY/DX=N*A*X
"
60 PRINT:PRINT:PRINTTAB(6)"FOR EXAMPLE:
Y=";:P=POS(0)
70 PRINT:PRINT:PRINT:INPUT"   CHOOSE CON
STANT A ";A
80 GOSUB500:PRINTTAB(P);A;"*X"
90 PRINT:PRINT:PRINT:PRINT:INPUT"   CHOO
SE POWER N ";N
100 GOSUB500:PRINTTAB(P);A;"*X";:P=POS(0
)
110 GOSUB500:PRINTTAB(P-1);"";N
120 FOR T=0 TO 2000:NEXT T:REM DELAY
130 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:
PRINT:PRINT:PRINT:PRINT
135 IF N=0 THEN PRINTTAB(14);"SO DY/DX=0
"
140 IF N<0 THEN PRINTTAB(8);"SO DY/DX=";
N*A;"/X";"";-(N-1)
150 IF N>0 THEN PRINTTAB(8);"SO DY/DX=";
N*A;"*X";"";N-1
160 PRINT:PRINT:PRINT:PRINT"  DY/DX SHOW
S THE SLOPE OF THE CURVE"
499 END
500 PRINT"":REM HOMES CURSOR TO TOP LEFT
510 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT:
PRINT:REM PUTS CURSOR TO EQUATION LINE
520 RETURN
```

This program will do the polynomial terms. Get the right number of spaces in the PRINT statements so that it reads well on the screen.

Suppose you have

$y=3x^3+8x^2+4x+3$

take each term separately and do program 88 on it, and you get

$dy/dx=9x^2+16x+4+0$

(remembering that x^1 is x, x^0 is 1 and zero times anything is zero).

When it comes down to physical meaning, remember that dy/dx is the slope, but it is also *the rate of change of y with respect to x*.

This is important when we consider sports cars and motor bikes, because the rate of change of distance is velocity, and the rate of change of velocity is acceleration, so that if the letter s represents distance, then ds/dt represents velocity, which we can call v. And it follows that the differential of w is acceleration, so that a=dv/dt:

$v=ds/dt$
$a=dv/dt$

Note that we have equations of s in terms of t, and that we are differentiating with respect to time, t. Note also that what we have achieved is to get acceleration from distance by differentiating twice. It follows that,

$$a=dv/dt=\frac{d}{dt}(ds/dt)=\frac{d^2s}{dt^2}$$

This symbol is read 'dee-too-ess-by-dee-tee-squared' (and like dy/dx or ds/dt is usually regarded as a single symbol). It represents the *second differential* of a quantity. If you differentiate it again, you would get the rate of change of acceleration (the *third* differential) and we would write it

d^3s/dt^3

Similarly, looking at our original example,

$y=3x^3+8x^2+4x+3$

and

$dy/dx=9x^2+16x+4$

and

$$\frac{d^2y}{dx^2}=18x+16$$

This works as long as the function is a polynomial in x, but what if it has other functions in it like Sin x or e^x or $\log_e x$? It is possible to calculate from first principles the differentials of y=Sin x, $y=e^x$, or $y=\log_e x$, just as the $y=x^2$ case was worked out. Program 89 will yield a page full of derivatives:

Program 89 STANDARD DERIVATIVES

```
1 REM PROGRAM 89
5 PRINT""
10 PRINT"     SOME STANDARD DERIVATIVES"
20 PRINT:PRINT"        Y              DY/DX"
25 PRINT
30 PRINT"      SIN X ...... COS X"
40 PRINT"      COS X ......-SIN X"
50 PRINT"      TAN X ......(SEC X)^2"
60 PRINT"      COT X .....-(COSEC X)^2"
70 PRINT"      SEC X ...... SIN X/(COS X)
^2
80 PRINT"    COSEC X ......-COS X/(SIN X)
^2
90 PRINT"      E^A.X ...... AE^AX"
100 PRINT"  ARCSIN X ......1/(+OR-)SQR(1
-X^2)
110 PRINT"  ARCCOS X .....-1/(+OR-)SQR(1
-X^2)
120 PRINT"  ARCTAN X ......1/SQR(1+X^2)
```

But, back to the maths. If

$y=e^{2x}$

then

$dy/dx=2e^{2x}$

and so forth.

The next area of differentiation to conquer is the differentiating of a product, so that if

$$y=x^2 * \text{Sin } x$$

can be found.

There is a rule, called the *Product Rule*, so that if

$$y=u*v$$

there is a product of two functions of x multiplied together. Then

$$dy/dx=u*\frac{dv}{dx}+v*\frac{du}{dx}$$

In words, the derivative of a product is equal to one function times the differential of the other function plus the other function times the differential of the first function. So in our example, if we let $u=x^2$ and $v=\text{Sin } x$,

$$dy/dx=x^2*\text{Cos } x+2x*\text{Sin } x$$

Obviously, it doesn't matter which function is chosen to be the u and which to be the v. Using programs 88 and 89 the elements can be found and just plugged into the product rule. Also, it can be done with quotients (two functions where one is divided by the other), using the *Quotient Rule*:

If $y=u/v$

then

$$dy/dx=\frac{v\frac{du}{dx}-u\frac{dv}{dx}}{v^2}$$

Look at it carefully. This time the expression for the derivative is no longer symmetrical, so it does matter which function is chosen for u and which for v. This is because choosing the right way makes simplifying the function much simpler, whereas choosing the wrong way can make it more difficult.

As an example of this in operation, consider how to get the derivative of $y=\text{Tan } x$. It is true that

$$y=\text{Tan } x=\frac{\text{Sin } x}{\text{Cos } x}$$

Let u=Sin x and v=Cos x, so that by using the Quotient Rule:

$$dy/dx = \frac{\text{Cos } x.\frac{d(\text{Sin } x)}{dx} - \text{Sin } x.\frac{d(\text{Cos } x)}{dx}}{\text{Cos}^2 x}$$

$$= \frac{\text{Cos } x(\text{Cos } x) - \text{Sin } x(-\text{Sin } x)}{\text{Cos}^2 x}$$

$$= \frac{\text{Cos}^2 x - \text{Sin}^2 x}{\text{Cos}^2 x}$$

We know from our trig identities that $\text{Cos}^2 x - \text{Sin}^2 x$ equals 1, so that

$dy/dx = 1/\text{Cos}^2 x = \text{Sec}^2 x$

Of course, the derivatives of Sin x and Cos x have to be known to work it through. How are they found from first principles? Remember an earlier lesson:

$$\text{Sin } x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \ldots$$

and this:

$$\text{Cos } x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \frac{x^8}{8!} - \ldots$$

Take this to as many terms as you wish, and do a program 88 on it, to see clearly what's going on.

2 Motion

Another of Sir Isaac Newton's discoveries was the laws of motion. Consider this:

> Let s represent the distance travelled by a rock dropped over a cliff. And let a be the acceleration due to gravity. Then this equation relates the variables to time elapsed, t:
> s=u*t+½*a*t²
> Here, u (which represents the initial speed of the rock) is zero.

This equation can be used in different ways; for example, the value for the acceleration due to gravity on the Earth can be plugged in, and a number of seconds after dropping the rock chosen, and it will give the distance the rock has fallen.

Let $a=10m/s^2$ and choose 10 seconds. Then do

PRINT Ø*1Ø+1Ø*1Ø↑2/2

and the answer that comes back is 5ØØ which is in metres since the acceleration due to gravity was specified in metres/second^2. To get it in feet, use the value for a of

$a=32$ ft/s^2

By differentiating the original equation we get

$$s=ut+\frac{at^2}{2}$$

which becomes

$$\frac{ds}{dt}=v=u+at$$

so that after 10 seconds, not only has our rock fallen 500 metres but also its velocity has increased to:

PRINT Ø+1Ø*1Ø

which is 1ØØ metres/second.

If we differentiate again,

$$\frac{d^2s}{dt^2}=\frac{dv}{dt}=a=a$$

which says that the acceleration is the acceleration, which is known in maths as an identity (or truism).

This all holds true under any conditions so long as a constant acceleration is applied to the object (which is reasonably true of a rock dropped over a cliff). We can make a small program to use this result ...

Program 90 FALLING OBJECTS

```
1 REM PROGRAM 90
2 REM FALLING OBJECTS
10 PRINT""
20 PRINTTAB(10)"FALLING OBJECTS"
30 PRINT:PRINT:PRINT
```

```
40 INPUT"  ENTER INITIAL FALLING VELOCIT
Y ";U
50 PRINT:INPUT"  ENTER TIME ELLAPSED ";T
60 LET S=U*T+9.81*T*T/2:PRINT:PRINT:PRIN
T
70 PRINT" DISTANCE FALLEN IS ";S; "METRE
S"
75 LET V=U+9.81*T:PRINT:PRINT
80 PRINT" VEL. AFTER ";T;" SECONDS IS ";
V;"M/S"
```

You could construct a program that can be used in other situations where the acceleration is constant, but not necessarily acceleration due to gravity. Invent a drag racer, or a rocket ship.

With falling objects, this equation is strictly true for all velocities not close to the speed of light, if the body is falling in a vacuum. If it is not falling in a vacuum, but through air, then there comes a point where the air-brake counteracts the pull of gravity and you reach a terminal velocity. This is true if you leap out of a plane; before you use your parachute you will accelerate until the rush of air against your body brakes your fall, and your terminal velocity will be about 120 miles per hour. The big surface area of your parachute brings down your terminal velocity to a reasonable figure. Unless of course you go parachuting on the Moon, which has no air. Although the acceleration due to gravity on the Moon is only 1/6 that of the Earth, you would quickly reach a velocity that would terminate *you* on impact.

Alter program 90 for lunar conditions, by changing 9.81 to 1.6, and run it. You will see that after only 10 seconds you are travelling at 80 m/s which is about 180 miles per hour – and on the Moon your parachute doesn't work!

3 Taylor's & Maclaurin's Theorems

Now that differentiation has been defined, it is possible to go back to series and see the basis for it. First an alternative notation which is useful for derivatives, and very useful in our particular application:

If $y = f(x)$ (read "eff of eks")

then $\frac{dy}{dx} = f'(x)$ (read "eff dash of eks")

and $d^2y = f''(x)$ (read "eff double dash of eks")

and so on.

Maclaurin says that if you assume you can represent the function by a series of powers of f, so that

$$f(x)=A_0+A_1x+A_2x^2+A_3x^3+A_4x^4+\ldots$$

then you can find the values of the A coefficients by successive differentiations of the polynomial

$$f'(x)=A_1+2A_2x+3A_3x^2+4A_4x^3+5A_5x^4+\ldots$$

and

$$f''(x)=2A_2+2{*}3A_3x+3{*}4A_4x^2+\ldots$$

and

$$f'''(x)=2{*}3A_3+2{*}3{*}4A_4x+\ldots$$

etc.

Which must all hold true when $x=0$, so when we substitute $x=0$ into the successive derivatives, we get

$$A_0=f(0)$$
$$A_1=f'(0)$$
$$A_2=\frac{1}{2!}f''(0)$$
$$A3=\frac{1}{3!}f'''(0)$$

etc.

and then we just substitute the A coefficients back into the expansion we assumed, to get

$$f(x)=f(0)+xf'(0)+\frac{x^2}{2!}f''(0)+\frac{x^3}{3!}f'''(0)+\ldots$$

which is known as Maclaurin's Theorem, and it can be used to get any of the expansions seen in the chapter on series.

Let's try e^x. Assume that $e^x=A_0+A_1x+A_2x^2+A_3x^3+\ldots$ If you differentiate e^x with respect to x, then you get e^x, so that $f(x)=f'(x)=f''(x)=$etc. So we have

$e^x=A_1+2A_2x+3A_3x^2+4A_4x^3+\dots$

and

$e^x=2A_2+2*3A_3x+3*4A_4x^2+\dots$

and

$e^x=2*3A_3+2*3*4A_4x+\dots$
etc.

Putting x=0 in each of the above expansions, we get

$A_0=1$, $A_1=1$, $A_2=1/2!$, $A_3=1/3!$ etc

so that

$e^x=1+x+x^2/2!+x^3/3!+\dots$

Taylor's Theorem is concerned with the identical process except that it starts with f(a+x) instead of f(x), so Maclaurin's theorem is a particular case of Taylor's theorem where a=0.
The result for Taylor's theorem is:

$$f(a+x)=f(a)+xf'(a)+\frac{x^2f''(a)}{2!}+\frac{x^3f'''(a)}{3!}+\dots$$

One application already seen is that expanding $(a+x)^n$ results in the binomial theorem.

4 Differentials and Differential Equations

Remember that the symbol dy/dx has a meaning as a whole (the limit of $\delta y/\delta x$ as δx tends to zero), but cannot be split in two to result in two separate meanings, any more than a number 8 can be cut in half and have a meaning for the two separate hoops. However, you will often find what amounts to dy/dx=2x, written as

dy=2x.dx

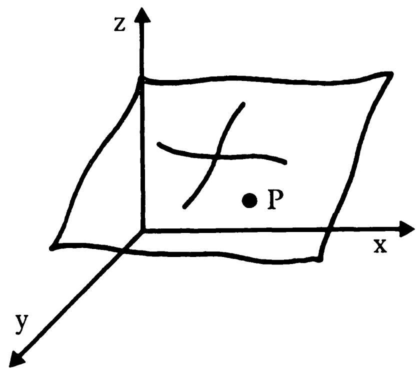
This is because we define things clled differentials so that the ratio of them *equals* dy/dx. And we write them dy and dx, making sure that we have

$$\frac{dy}{dx}=dy/dx$$

We will see this in the next chapter, which concerns integration. And on the subject of notation, you may someday see an unusual dy/dx symbol in which the d's are not quite d's nor Greek deltas, but something peculiar and halfway between. This is called a *partial* derivative, and looks like this:

$\frac{\partial y}{\partial x}$ (read as "partial-dee-why-by-dee eks")

It turns up in a quantity that varies with two quantities. For example, if a surface is defined in a three dimensional coordinate system, like this:



then z is a function of both x and y, because as any point P roams about on the surface, z takes on a variety of different values as x and y change. The two lines drawn on the surface are parallel to the x and y axes, though, and if the point moves along one of them x will change but y will remain constant, whereas if the point moves along the other line, then y changes as x remains constant, so the partial differential can be considered with respect to x or with respect to y.

In practice, suppose we have a surface with an equation

$$z=3x^2+4y^2$$

Then to find $\partial x/\partial x$ we differentiate with respect to x, treating y as a constant

$$\frac{\partial z}{\partial x}=6x$$

and to find z/y we differentiate with respect to y, treating x as a constant

$$\frac{\partial z}{\partial y}=8y$$

A differential equation is an equation with ordinary terms except that one of the terms (at least) is a derivative of some kind. As with determinants, a differential equation has an *order*, so that if its highest derivative is a first derivative it will be of order one, if there's a second derivative but none higher it will be of order two, and so on. An example of a differential equation of the *second* order is:

$$\frac{d^2y}{dx^2}-3\frac{dy}{dx}+2y=0$$

In general, a differential equation can't be solved without using integration, which we haven't yet looked at, but we can have a look at rabbits and foxes.

Imagine an island inhabited only by rabbits and foxes. The foxes eat the rabbits and the rabbits eat grass, but since it's an island, if there are plenty of rabbits then the number of foxes will increase (because of plentiful food supply) and soon there will be so many foxes that the number of rabbits begins to decrease. Soon, the rabbits decline so much that they are difficult to find, and the island can't support so many foxes and they decline too. Pretty soon, the fox population is so low that hardly any rabbit eating is being done, so the rabbit population begins to pick up again, and so it goes see-sawing back and forth. This is a classic example of a model being made of reality by a scientist (in this case a very simple ecological model).

Let the rabbit population be r, and the fox population be f. The *rate of change* of the rabbit population is the derivative of r with respect to time, which can be written as

$$\frac{dr}{dt}$$

And similarly, the rate of change of the fox population is

$$\frac{df}{dt}$$

We suppose that the rate of change of the rabbit population depends on two factors: 1) The number of rabbits (since the more rabbits there are the more baby bunnies will be born); and 2) The number of rabbits times the number of foxes. Also, we suppose the fox population depends upon two factors: 1) The number of foxes (making more baby foxes); and 2) The number of foxes per rabbit, f/r.

Now if something is *proportional to* something else, then we can write that it equals a constant times that factor, so we can write two differential equations, like this:

dr/dt=a*r−b*r*f

and

df/dt=c*f−g*f/r

where the letters a, b, c and g are constants. After an increment of time (small period) called dt, the rabbit population will be incremented by dr and the fox population by df. By using a FOR loop to increment the time by a small amount, and resorting to a number of different graphical display techniques, the whole operation can be followed through.

Clearly, the solution of a pair of differential equations like this depends on the initial conditions - in this case, how many rabbits and foxes you start with.

As with ordinary equations, differential equations come in a variety of different forms, and there are standard methods you can resort to to solve them, but this is getting into the realm of much higher maths.

13

INTEGRATION

1 Anti-differentiation

We have learned that the raising of e to a power can be reversed by taking logs (to base e), and other inverse (or reverse) operations; the reverse process of differentiation is important in maths, and it is called integration. If you have

$y=x^2$

then

$dy/dx=2x$

Similarly, if you have $y=2x$, then if you integrate it you will get something equal to x^2. That something is called the integral of y with respect to x, and the word integral is represented by a kind of drawn-out letter S that looks like this:

$$\int$$

We represent the 'with respect to x' by the symbol:

dx

So that we write

If $y=2x$

then

$\int y\, dx=x^2$

One very important point: if you differentiate a constant you get zero. This means that there is no way of knowing what it was in the first place. For example,

If $y=x^2+C$ (where C is a constant)

then

$dy/dx=2x$

To take account of this, we should have added that possible constant in when we integrated it back. It could be zero, or any numerical value, and without further information we have no way of telling, but we have to add it in like this:

If $y=2x$

then

$\int y\,dx=x^2+C$

Of course if you are supplied with more information you can find the value of C, so that if for example we know that the LHS of the equation is 10 when x is 3, then

$10=3^2+C$

so that

$C=10-9=1$

In the same way that there is a general method of going from y to dy/dx, there is also a reverse method to go from dy/dx to y and from y to $\int y$ dx. Compare the two methods:

$$\begin{array}{ccc} y & \text{differentiation} & dy/dx \\ x^n & \longrightarrow & n*x^{n-1} \end{array}$$

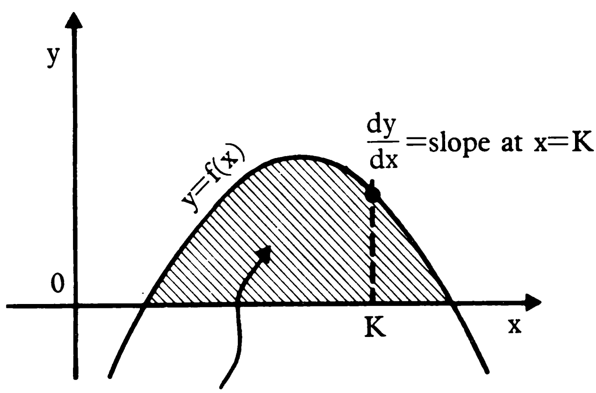
$$\begin{array}{ccc} dy/dx & \text{integration} & y \\ x^n & \longrightarrow & \dfrac{x^{n+1}+C}{n+1} \end{array}$$

Notice that to differentiate we multiplied by the index and then subtracted one from the index. Now we add one to the index and divide by the new index when integrating. And we *do* *not forget* to add in C (which is called the constant of integration).

Before looking at a program, examine the *meaning* of an integral. Remember that if y equals some function of x,

$y=f(x)$

a graph can be drawn of it, and the derivative (differentiated form) dy/dx gives you the slope of the curve, and by substituting in a value of x, the slope of the curve can be found at any given point. The integral $\int y\,dx$ also has a meaning: it represents *the area under the curve.*

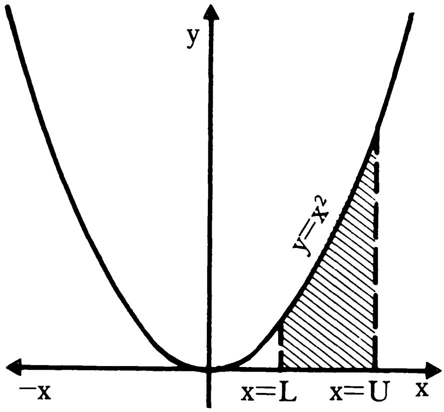


$\int y\,dx$=Area between curve and x-axis

In the world of calculus, the integration sign often gets written with a couple of small numbers against it, one at the top and one at the bottom, like this:

$$\int_0^5 y\,dx$$

These are called *limits*, and are values of x between which you are considering the integral. Not all functions obligingly curve over the below the x axis again, as in the diagram, so that the area under the curve would be infinite. Take the x^2 function for example:



The shaded area is bounded by the curve and the x-axis, and by the lines x=L and x=U, and here L is the lower limit and U the upper limit, so that if we want to write it out as an equation,

$$\text{Shaded Area}= \int y\,dx$$

$$= \int_L^U x^2\,dx = \left[\frac{x^3}{3}\right]_L^U$$

These limits can be regarded as meaning the area from x=0 to U, *minus* the area from x=0 to L, so if the limit values are substituted into the equation they can be subtracted to get a value for the area. Suppose we'd said that U=5 and L=3, then we'd have

$$\left[\frac{x^3}{3}\right]_3^5 = \frac{5^3}{3} - \frac{3^3}{3} = 41\tfrac{2}{3} - 9 = 32\tfrac{2}{3}$$

This integrating between definite limits is called *definite integration*. The constant of integration does not appear here, because the limits make up for it.

All this allows us to find areas of quite sophisticated shapes. The curve $y=x^2$ is by no means a simple shape (it's a parabola).

This program sorts out the polynomials, and is in effect just the reverse of program 88.

Program 91 POLYNOMIAL INTEGRATION

```
1 REM PROGRAM 91
2 REM POLYNOMIAL INTEGRATION
10 PRINT""
20 PRINT "        INTEGRATION OF POLYNOMIA
LS"
25 PRINT:PRINTTAB(9)"IF:             Y=A*X"
30 PRINT:PRINT:PRINTTAB(9)"THEN:    BUK";"
Y DX = A*X+C"
40 PRINT:PRINT:PRINTTAB(9)"FOR EXAMPLE:"
50 PRINT:PRINT:PRINTTAB(9)"Y=";:P=POS(0)
60 PRINT:PRINT:INPUT"         CHOOSE CONST
ANT A ";A
70 PRINTTAB(P);"";A;"*X";:P=POS(0)
80 PRINT"":input"        choose power n "
;n
90 printtab(p-1);"";N
100 FOR T=0 TO 500:NEXT T
110 PRINTTAB(9);"";"so..."
120 if n=-1 then printtab(10);"BUK";"Y D
X =";A;"*LN X";:END
130 PRINTTAB(10);"BUK";"Y DX =";A/(N+1);
"*X";"";N+1;:END
```

In program 91, when a=1 and n=−1 this is being integrated:

$y=1/x$

and it is true that

$\int y dx=\log_e x+C$

so line 11Ø is there to take care of it.

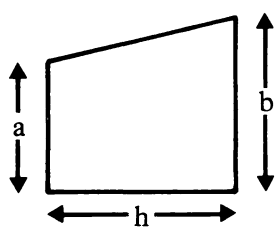
It is tempting to reproduce a version of program 89 which shows how to get standard derivatives of the common functions, but this time showing standard integrals. In fact, from a list of derivative conversions you can see what going in the other direction will give you (always remembering the constant C).

Most maths texts have a list of standard integrals for different functions, and the more comprehensive the book the longer the list.

Definite integrals are easier on the computer, because here we are dealing with numbers rather than manipulation of formulae. The principles of this aspect of integration need an explanatory word, so let's examine Simpson's Rule and the Trapezoidal Rule.

Taking the trapezoidal rule first, it is the same as drawing an irregular shape on some graph paper and finding the area by counting up all the tiny squares, except that the trapezoidal rule uses *strips*.

A trapezium is a shape like this:

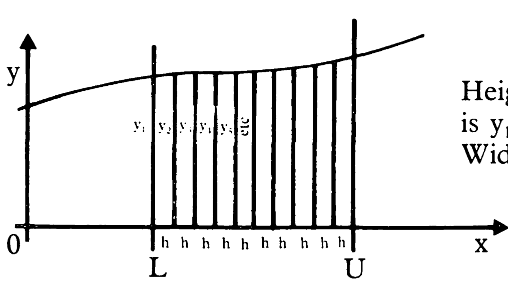


Trapezium
Area=½(a+b)*h

with *one* pair of opposite sides parallel. If a function is drawn and the area under it shown, strips of the area can be made so that each of the strips is a trapezium. (This means that the curve is considered to be a series of straight lines, but that is where the approximation comes in.)

Clearly, if the strips are exceedingly small and narrow, the results will be more accurate, but the price is dealing with more strips. We can take advantage of the accuracy and the computer will do the work.

Illustrate it with ten strips:



Heights of LHS of strips is $y_1, y_2, y_3, \ldots, y_{11}$
Width of each strip is h

Then the area will be the sum of the trapeziums:

$$\text{Area}=\frac{y_1+y_2}{2}h+\frac{y_2+y_3}{2}h+\ldots+\frac{y_{10}+y_{11}}{2}h$$

so,

$$\text{Area}=h\left(\frac{y_1+y_9}{2}+y_2+y_3+y_4+\ldots+y_{10}\right)$$

which in words amounts to

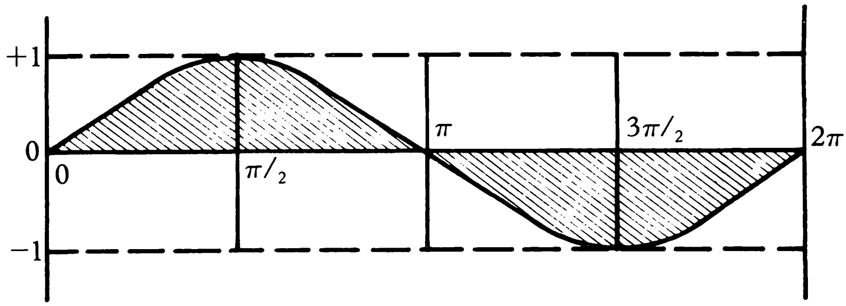
Area=width of strip *((first edge+last edge)/2+sum of others)

Program 92 TRAPEZOIDAL RULE

```
1 REM PROGRAM 92
2 REM TRAPEZOIDAL RULE
10 PRINT""
20 INPUT"   UPPER LIMIT ";U :PRINT
30 INPUT"   LOWER LIMIT ";L
40 DIM A(101):LET RANGE=U-L:LET H=RANGE/100:N=1:S=0
50 FOR X=L TO U STEP H
60 LET Y=SIN(X)
70 LET A(N)=Y
80 LET N=N+1
90 NEXT X
100 FOR N=2 TO 100
110 LET S=S+A(N)
120 NEXT N
130 LET AREA=H*(((A(1)+A(101))/2)+S)
140 PRINT:PRINT:PRINT AREA
```

Here we are looking at y=Sin x. The function can be put in at line 6Ø to your own taste. The integration of the Sin x function is interesting. Remember we are using radians on the computer, so you could set upper limit to PI and lower limit to zero. The answer is approximately 2 units. The area over a full revolution however is approximately zero (from 2*PI to zero), because the area below the x axis between PI and 2*PI is counted negative;

beware of that when using this rule with the repetitive functions.



Simpson's Rule is a refinement that tries to lay pieces of parabola against the curve instead of straight lines.

$$\text{Area}=\frac{h}{3}(y_1+2(y_{odd})+4(y_{even})+y_{11})$$

or in words,

Area=h/3*((first edge+last edge)+2 (odd edges)+4 (even edges))

It is a refinement that is generally more accurate, and the two programs can be run on a given integration problem to see how they both compare with the *real* answer!

Program 93. SIMPSON'S RULE

```
1 REM PROGRAM 93
2 REM SIMPSONS  RULE
10 PRINT""
20 INPUT"    UPPER LIMIT ";U :PRINT
30 INPUT"    LOWER LIMIT ";L
40 DIM A(101):LET RANGE=U-L:LET H=RANGE/
100:N=1
50 FOR X=L TO U STEP H
60 LET Y=SIN(X)
70 LET A(N)=Y
80 LET N=N+1
90 NEXT X:EVEN=0:ODD=0
100 FOR N=2 TO 100 STEP 2
```

```
110 LET EVEN=EVEN+A(N)
115 LET ODD=ODD+A(N+1)
120 NEXT N
130 LET AREA=(H/3)*(A(1)+A(101)+4*EVEN+2
*ODD)
140 PRINT:PRINT:PRINT AREA
```

Notice that in both program 92 and 93 1ØØ strips are being used, which takes a little time to calculate.

In the same way that differentiating successively will give you

dy/dx, d^2y/dx^2, d^3y/dx^3, etc

you can get successive integrals too. Like this:

$$y=\int \int \int f(x)\ dx\ dx\ dx$$

One at a time is integrated, like stripping off coats of paint. It's like a ladder, where you differentiate to go down a rung and integrate to go up. But not every time you see integration signs ganging up together will they be straightforward *successive* integrations. Sometimes they're *multiple* integrals, so that the first integration is with respect to x, the second might be with respect to y and the third with respect to z,

$$\int \int \int f(x,y,z)dx\ dy\ dz$$

for example,

$$\int \int \int (x^2+y^2+z^2)\ dx\ dy\ dz$$

And the result is the same whichever order you integrate in.

A short list of standard integrals:

y	$\int y\ dx$
$1/x$	$\log_e x+C$
a^x	$a^x * \log_a e+C$
e^x	e^x+C
Sin x	−Cos x+C
Cos x	Sin x+C
Tan x	−log Cos x+C
Sinh x	Cosh x+C
Cosh x	Sinh x+C
Tanh x	log Cosh x+C

Since the integral of Sin x is listed here, it can be checked with the Trapezoidal and Simpson's Rules.

If y=Sin x

$$\int_0^{\pi} y\,dx = \Big[-\text{Cos } x\Big]_0^{\pi}$$

$$=(-\text{Cos } \pi)-(-\text{Cos } 0)$$

And we can show that Cos π is -1 and Cos 0 is 1, so our area is:

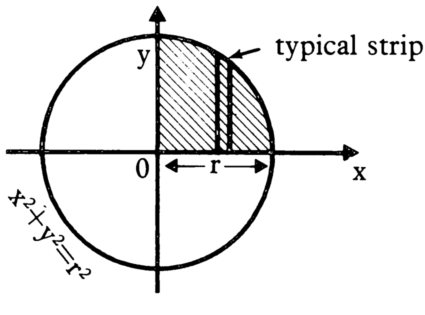
$$-(-1)-(-1)=1+1=2$$

It is not easy to find a standard integral identical to any given expression you want to integrate, so how do you proceed when faced with a complicated expression? It takes ingenuity and practice to figure out how to integrate some expressions. There are a few standard methods for products and so on, but it is mostly a question of experience. Generally speaking, the expression must be prepared so that it fits one of the standard forms, and that can be done in a variety of different ways, such as 'trigonometric substitution' or using 'partial fractions'.

We have seen integration as 'anti-differentiation', and that the process is essentially the finding of areas under curves. We should look closer at the numbers aspect of it.

Starting with y=f(x) (y is a function of x), a table can be drawn up of corresponding values as if to draw a graph. Cut the area into strips by choosing a suitable interval for x, then *sum* the strips together to get the whole area. The important point to note is that in integration the width of the strips goes from small, δx, to vanishingly small, dx.

To demonstrate the method for areas, the circle is as good a figure as any. First, the most convenient way to carve up the circle into strips is like this:



The equation of the circle is $x^2+y^2=r^2$ where r is the radius of the circle. The area to look at is the shaded quadrant, and the representative strip goes from limits of 0 to r. So,

$$\text{Area of quadrant}=\int_0^r y\,dx$$

and we know that

$$x^2+y^2=r^2$$

so that

$$y=\sqrt{r^2-x^2}$$

Therefore

$$\text{Area}=\int_0^r \sqrt{r^2-x^2}\,dx$$

Without worrying about how to integrate that expression, looked up in a reference book as a standard integral it would be

$$\frac{r^2*\mathrm{Sin}^{-1}(x/r)+\tfrac{1}{2}x*\sqrt{r^2-x^2}}{2}$$

so that our area is that expression with limits of 0 and r, and we substitute values of r and 0 wherever we find x.

When $x=r$, $\mathrm{Sin}^{-1}(x/r)=\mathrm{Sin}^{\Delta 1}(1)$ (the angle whose sine is 1) which in radians is PI/2.

And by similar reasoning, $\mathrm{Sin}^{-1}(0/r)=\mathrm{Sin}^{-1}0=0$ so we have

$$\text{Area}=(\tfrac{1}{2}r\sqrt{r^2-r^2}+\tfrac{1}{2}r^2*PI/2)-0$$

which reduces to

$$\text{Area}=\frac{PI*r^2}{4}$$

But remember that this is the area of our quadrant which is only a quarter of the full circle, so the area of the full circle is four times this:

$$\text{Area of circle}=PI*r^2$$

The method can be used for any shape whose equation is known. If you had a plane area made up of strips, you could also make up a solid figure from slices, and use a method involving integration to find *volumes*. It's also possible to find the lengths of curves, using an equation that uses both differentiation and integration.

14

THE RANDOM FACTOR

1 The Random Factor

The last chapter deals with one of the most interesting aspects of the microcomputer, and also serves as an introduction to one of the most fascinating branches of mathematics – the mathematics of chance.

From tiny electrons to vast clouds of intergalactic gas, our universe is one where matter and energy move and interact with apparent randomness – unpredictability – yet there *are* rules, and the discoveries that have been made in 20th-century physics have had a lot to do with probability theory.

In Tom Stoppard's play *Rosenkrantz and Guildenstern are Dead*, the two principal characters open by tossing a coin which, no matter how many times they toss it, *always comes up heads*. This leads them to conclude that something has happened to their world (they are in fact *dead*!), and serves to remind us that, paradoxically, out of unpredictability comes clues to the world we live in.

Examine the computer's function RND. It stands for random, and if you, for example,

```
PRINT RND(1)
```

the computer will give you back a number between Ø and 1, and that number will be chosen by the computer at random. Try it a few times, or better still, get a page full of random numbers by running this:

```
1Ø FOR n=1 TO 21: PRINT RND(1): NEXT n
```

The numbers are indeed between Ø and 1, and don't look like any well-defined sequence or series. They appear to be independent of one another and completely random, but unfortunately they are not, quite.

The RND function is derived from a function. It would be extremely difficult to get a thing so intensely ordered and predictable as a microchip to behave in a really random fashion so it has to be simulated. The RND function receives a number from the computer's memory and uses it to calculate the random number it gives back, and although it is not truly random, it's good enough for most games, and some statistical modelling.

For a random number between Ø and 1Ø, try this:

```
1Ø FOR n=1 TO 21: PRINT 1Ø*RND(1): NEXT
```

And if you want random *integers* between Ø and 1Ø, slip in an INT function:

```
1Ø FOR n=1 TO 21: PRINT INT(1Ø*RND(1): NEXT
```

If you want a random integer between 5 and −5, you have to subtract five:

```
1Ø FOR n=1 TO 21: PRINT INT(1Ø*RND(1))−5: NEXT
```

You choose the range of your random number. But beware: the number you multiply RND by is an absolute upper limit of that range, so that if you have 1Ø*RND(1) you can never get 1Ø, only 9.999999.

Now we will examine probabilities, look at permutations and combinations, and model some physical situations on the computer using what we have learned in theory.

2 Permutations and Combinations

A *permutation* is a way of ordering things. For example, if we have three letters, A, B, and C, how many permutations are there? There are six, and that's *all.*

A B C	B A C	C A B
A C B	B C A	C B A

If you have two objects, then you have only two permutations:

A B
B A

But if you have four:

A B C D	B A C D	C A B D	D A B C
A B D C	B A D C	C A D B	D A C B
A C B D	B C A D	C B A D	D B A C
A C D B	B C D A	C B D A	D B C A
A D B C	B D A C	C D A B	D C A B
A D C B	B D C A	C D B A	D C B A

There are 24 permutations. Maybe we can think of a way to automate this. Let's look at the facts so far:

No. of objects:	1	2	3	4	...
No. of permutations:	1	2	6	24	...

If you do not recognise this, go back and try program 69. The *factorial* function is the key to it.

Imagine now a bag of four letters, A, B, C and D. How many ways are there of taking just two of the letters out of the bag?

We could have:

1) a and B
2) A and C
3) A and D
4) B and A
5) B and C
6) B and D
7) C and A
8) C and B
9) C and D
10) D and A
11) D and B
12) D and C

There's a way of representing twelve options, using a letter P (for permutation), and when choosing two from four as above, we would write,

$^4P_2$

The top number is the total to pick from and the bottom number is the number being picked. In our case,

$^4P_2=12$

and there is a general formula for choosing so many items from a total number:

$^rP_r=n!/(n-r)!$

so that in our case, we'd have

$^4P_2=4!/(4-2)!=24/2=12$

Now look again at the exercises. So long as notice is taken of the *order* that the letters are drawn out of the bag, there will be twelve permutations. But if the order doesn't matter, and drawing A and B is the same as drawing B and A, (options 1 and 4 in the list), then there are only six outcomes. These outcomes are called *combinations*, and we represent this mathematically as

$^nC_r=6$

and it so happens that

$^nC_r={}^nP_r/r!=n!/(n-r)!*r!$

This is so because each permutation of ^nP_r consists of r objects which can be arranged amonst themselves in r! ways.

You may wish to review the generation of factorials at this point by looking back over program 69.

Program 94 PERMUTATIONS

```
1 REM PROGRAM 94
2 REM PERMUTATIONS
10 PRINT""
20 PRINT"  PERMUTATION IS THE NUMBER OF
WAYS"
30 PRINT" OF CHOOSING R OBJECTS FROM A T
OTAL N."
35 PRINT:PRINT"****************************
***************"
40 PRINT:PRINT:PRINT
45 INPUT"  TOTAL NUMBER OF OBJECTS ";T:P
RINT
```

```
50 INPUT"  NUMBER OF CHOSEN OBJECTS ";R
60 LET X=T:GOSUB200:NFAC=X
70 LET X=T-R:GOSUB200:NMRFAC=X
75 PRINT:PRINT:PRINT:PRINT
80 PRINT"   PERMUTATIONS=";NFAC/NMRFAC:E
ND
200 F=1
210 FOR N=X TO 2 STEP-1
220 LET F=F*(N-1)
230 NEXT N:IF X=0 THEN F=1
240 X=F:RETURN
```

The subroutine in lines 21Ø–24Ø is the bit of code that works out the factorial. (This has the same restriction on it as the original factorial program so far as maximum number size is concerned; that is, it will work with numbers up to 33, but beyond that it will return an overflow error report.)

Here is a combinations routine:

Program 95 COMBINATIONS

```
1 REM PROGRAM 95
2 REM COMBINATIONS
10 PRINT""
20 PRINT"  COMBINATION IS THE NUMBER OF
WAYS"
30 PRINT" OF CHOOSING R OBJECTS FROM A T
OTAL N."
32 PRINT" WHEN THE ORDER OF SELECTION DO
ES NOT"
34 PRINT" COUNT."
35 PRINT:PRINT"*************************
***************"
40 PRINT:PRINT:PRINT
45 INPUT"  TOTAL NUMBER OF OBJECTS ";T:P
RINT
50 INPUT"  NUMBER OF CHOSEN OBJECTS ";R
60 LET X=T:GOSUB200:NFAC=X
70 LET X=T-R:GOSUB200:NMRFAC=X
73 LET X=R:GOSUB200:RFAC=X
```

```
75 PRINT:PRINT:PRINT:PRINT
80 PRINT"    COMBINATIONS=";NFAC/(NMRFAC*
RFAC):END
200 F=1
210 FOR N=X TO 2 STEP-1
220 LET F=F*(N-1)
230 NEXT N:IF X=0 THEN F=1
240 X=F:RETURN
```

Take a look at this for a bit of mathematical connection: recalling Pascal's Triangle, you can use the combinations program to get a line of it. If you want the sixth line, start by writing down one, then the next term is ${}^6C_1=6$, then the next term is ${}^6C_2=15$, then ${}^6C_3=2\emptyset$, then ${}^6C_4=15$, then ${}^6C_5=6$, and lastly a one. You can write out the line in full:

1 6 15 20 15 6 1

It works for any line, and notice that it's symmetrical, so that it follows that

$${}^nC_r={}^nC_{n-r}$$

It also follows that in the binomial expansion

$(a+x)^n$ is a series of terms like ${}^nC_r{\star}a^{n-r}{\star}x^r$

which leads on to Recurrence or Recursion formulae, but we're not going to look at those here.

3 Probability

This is important if you are going to take up gambling, or engage in making predictions and the way we approach it is via those tools of chance, the coin and the dice.

Consider a coin, and ignoring the remote possibility that it can come down on its edge (or not come down at all), you can say that it has two possible states: it can be 'heads' or it can be 'tails', and either is an equal chance. The probability of heads is Ø.5 and the probability of tails is Ø.5, which we write like this:

p(heads)=Ø.5
p(tails) =Ø.5

Note that p(tails) OR p(heads)=Ø.5+Ø.5=1, and that a probability of Ø means 'impossible' and a probabiltiy of 1 means 'certain'.

This does not predict that if a coin is tossed ten times you will get five heads and five tails. What it does mean is that if the coin is tossed a large number of times, it will come up half heads and half tails (more or less), and that if it could be tossed an *infinite* number of times you would get half heads and half tails. As the number of trials tends to infinity the results tend to their predicted probability. Program 69 simulates this.

Program 96 PROBABILITY TRIALS

```
1 REM PROGRAM 96
2 REM PROBABILITY TRIALS
10 PRINT""
20 INPUT "   NUMBER OF TRIALS ";N
30 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT
40 PRINT"  N         H         T  "
50 H=0:T=0
60 FOR X=1 TO N
65 R=RND(1)
70 IF R<= 0.5 THEN T=T+1:PRINTTAB(0);X,H
,T
75 IF R>0.5 THEN H=H+1:PRINTTAB(0);X,H,T
77 PRINT""
80 IF T<>0 THEN PRINT"h/t=";h/t;""
90 NEXT X
99 PRINT""
100 end
```

While we are not dealing with a random phenomenon here, just with the computer's pseudo-random generator, it is sufficiently good to demonstrate the notion. The program prints the number of the trial at the top of the screen, the number of tails to left of centre, the number of heads to right of centre and finally the ratio of heads to tails at the bottom. If you run the program time and again with larger and larger numbers of trials, the bottom number will get closer to one.

The dice program simulates the throwing of a pair of dice 500 times and builds up what is known as a frequency histogram of the results. The results can be any integer from 2 to 12 (adding the totals of the two dice) and if waiting four seconds between throws begins to bore you, just hold a key down to speed things up. The program freezes after 5ØØ goes and you have to break it by pressing CAPS SHIFT and the BREAK keys simultaneously.

Program 97 DICE

```
1 REM PROGRAM 97
2 REM DICE
10 PRINT""
20 PRINT"    N       THROW      DIE 1
 DIE 2  "
21 GOSUB 400
22 DIM X(12)
30 FOR N=1 TO 500
40 LET D1=INT(RND(1)*6+1)
50 LET D2=INT(RND(1)*6+1)
60 LET S=D1+D2
70 LET X(S)=X(S)+1
80 GOSUB 200:REM UPDATE FIGRES
90 GOSUB 300:REM UPDATE CHART
100 NEXT N
110 PRINT""
120 END
200 PRINT""
210 PRINT" ";N;"      ";S;"       ";D1;"
"      ";D2
300 PRINT"";:REM REF POINT
310 FOR C=1 TO S
320 PRINT"";
330 NEXT C
340 FOR P=0 TO 150 STEP 5
350 IF X(S)>P THEN PRINT "§";
360 NEXT P
370 RETURN
400 PRINT"";:REM REF POINT
410 FOR C=2 TO 12:REM WRITE NUMBERS
```

```
420 PRINT C
430 NEXT C
440 RETURN
```

Sooner or later, although the throwing of dice is random the histogram at the bottom of the screen builds up in a characteristic way. There are generally taller strips in the middle and shorter strips at the edges. Why should this hump shape occur? Fortunately throwing dice is a simple system to look into.

First let's examine how a score is arrived at. Die number 1 has possible scores 1,2,3,4,5 and 6, and the same for die number 2, so for a total score of 2, there must be a one on die 1, and a one on die 2.

To get a score of three, there can be either a two on die 1 and a one on die 2, or vice versa, i.e. two separate ways of scoring three.

To get a score of four, there can be a two on each die, a three on die 1 and one on die 2, or vice versa, which is *three* ways ... and so it goes. To summarise:

Score (n)	*Number of ways* to get it	p(n)
2	1	1/36
3	2	2/36
4	3	3/36
5	4	4/36
6	5	5/36
7	6	6/36
8	5	5/36
9	4	4/36
10	3	3/36
11	2	2/36
12	1	1/36

There are six ways to get a score of seven, making seven the most popular result. Obviously, if the dice are not loaded, each of the 36 different score combinations are equally likely, so we would expect the probability of a seven score to be six times as likely as a score or a twelve score. The absolute probabilities will be, for example, 6/36 for a score of seven, which is 1/6, or Ø..1666667, and for a score of say three 3/36=1/12, which is Ø.Ø833333. If all the p(n)s were converted into decimals and added together the result would be one. This must be so, since the score for one throw must be between 2 and 12 inclusive.

Before you open your own casino, run the dice program a few rounds and compare the frequency with which the combinations turn up to the predicted probabilities. Just divide the occurence numbers for each score (at the top of the screen) by 5ØØ, and compare them with the p(n) predictions.

4 Probability Distributions

In the 17th and 18th centuries there flourished a family that turned out several excellent mathematicians and scientists, and they were called Bernoulli. One of them, probably Jacques Bernoulli (1654–1705), came up with the Bernoulli distribution, otherwise known as the binomial distribution, and it goes like this:

If the probability of something happening is p, and the probability of it not happening is q (where q=1−p, since p=q must equal one), then what is the probability of it happening x times out of n trials?

The formula that tells you is:

$$p(x)={}^{n}C_{x}\star p^{x}\star q^{(n-x)}$$

so for example, the probability of getting two heads in six tosses of a coin is,

$$p(2)={}^{6}C_{2}\star(\emptyset.5)_{2}\star(\emptyset.5)^{6-2}=\frac{6!}{4!2!}\star(\emptyset.5)^{6}=\emptyset.234375$$

Program 98 extends this idea, asking you how many trials you want, how many successes you want, and the probability of one success. Try it with the above example, putting in6,2 and Ø.5 respectively. Or you could find the chance of getting two sixes with three rolls of a dice, where the number of trials will be three, the number of successes is two, and the chance of throwing one six is 1/6. You ought to get Ø.Ø694.

Program 98 BINOMIAL DISTRIBUTION

```
1 REM PROGRAM 98
2 REM BINOMIAL DISTRIBUTION
```

```
10 PRINT""
20 INPUT"   NUMBER OF TRIALS ";N:PRINT
30 INPUT"   NUMBER OF SUCCESSES ";X:PRINT
40 INPUT"   PROBABILITY OF ONE SUCCESS ";
P:PRINT
50 Y=N:GOSUB200:FT=Y
60 Y=N-X:GOSUB200:RT=Y
70 Y=X:GOSUB200:FC=Y
80 PRINT:PRINT:PRINT
90 PRINT" PROBABILITY OF ";X; " SUCCESSE
S"
100 PRINT" IN ";N;" TRIALS IS "; (FT/(RT*
FC))*(P^X)*((1-P)^(N-X)):END
200 F=1
210 FOR A=Y TO 2 STEP -1
220 F=F*A
230 NEXT A:IF Y=0 THEN F=1
240 Y=F:RETURN
```

The next idea concerns collecting data. If you start making a list of how tall people are, you will get raw data on heights. If you wanted to draw some conclusions about that data, you need to process it and condense it into two numbers, called the *mean* and the *standard deviation.*

The first, the mean, is just the average, obtained by adding all the values together and dividing by the number of them. If there are values 2,3,4,5 and 6, they are added together to get 2Ø and divided by the number of values (five) giving a mean of 4. The mean, then, is the value about which the values cluster.

The standard deviation is what is known as a measure of central tendency, and shows you how closely the values are clustering about the mean. If the SD is large, then the clustering is loose and the values are all over the place, and if the SD is small then the cluster is tight and the values are mostly close to the mean.

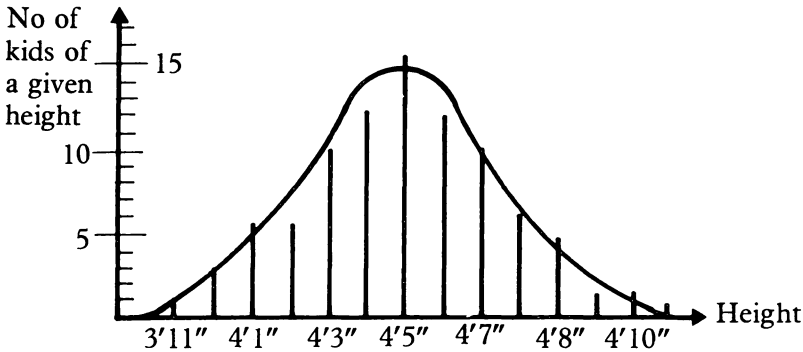
Get hold of some raw data (count the number of seconds between successive cars on the road, or something else liable to be reasonably random), then plug the values into program 99. It will give you back the mean and the standard deviation. The way it works out the standard deviation, incidentally, is by taking each of the values away from the mean (to see how far away from the central value it is), squaring it, adding all those squared differences together and dividing the result by the number of values, and finally taking the square root.

Program 99 MEAN & STANDARD DEVIATION

```
1 REM PROGRAM 99
2 REM MEAN & STANDARD DEVIATION
10 PRINT""
20 INPUT"  NUMBER OF VALUES ";N:PRINT: D
IMA(N):S=0
30 FOR X=1 TO N:INPUT" NEXT VALUE ";A(X)
:PRINT A(X):NEXT X
40 FOR X=1 TO N:S=S+A(X):NEXT X:PRINT:PR
INT:PRINT"SUM OF VALUES IS ";S
50 PRINT:PRINT:PRINT"MEAN OF VALUES IS "
;S/N
60 SD=0:FOR X=1 TO N:D=((S/N)-A(X))*((S/
N)-A(X)):SD=SD+D:NEXT X
70 PRINT:PRINT:PRINT"ST. DEV. IS ";SQR(S
D/N)
```

Notice that if *all* the values were the same, there would be no deviation from the central value, and SD would equal Ø.

The thing about heights of people, or weights of people, or lengths of manufactured components or anything else of a randomly-arrived-at nature, is that there are usually very few from the sample that are very much bigger than the mean, and similarly few that are very much smaller. If you were to take the heights of a class full of kids, measuring to the nearest inch, you might have a mean of 4 ft 5 inches. Then if you plotted out the number of kids of a given height against the height on a graph, then you'd get a graph that was a hump, with the centre of the hump coinciding with the mean, like this:



The standard deviation is the measure of the sharpness of the hump in an inverse sort of way, so that if the standard deviation is small there will be a sharp curve, and if the standard deviation is large a flatter curve. Once the mean and the standard deviation of a sample are known, an idealised curve can be prepared that the real results can be expected to follow more or less (given a big enough sample). This curve has an equation first developed by a German mathematician called Johann Gauss (1777–1855), and the distribution is called the *normal* or *Gaussian distribution*.

The equation is,

$$Y=\frac{1}{SD\sqrt{2*PI}} *EXP-(X-MEAN)^2/2*SD^2$$

One useful utility is to be able to draw a graph of data points collected. Suppose the profits of a smal business over 18 months go like this:

JAN	£320	JUL	£530	JAN	£1106
FEB	£310	AUG	£300	FEB	£ 995
MAR	£405	SEP	£495	MAR	£1047
APR	£312	OCT	£615	APR	£1272
MAY	£378	NOV	£700	MAY	£1320
JUN	£440	DEC	£950	JUN	£1471

The figures would be much easier to visualise if they were put on a graph.

Program 100 GRAPH

```
2 REM HISTOGRAM
7
10 PRINT""
20 GOSUB500:REM PRINT HEADING
25 INPUT  "input title of your histogram
";t$
30 input"how many data points (<11)";n
33 rem input values
35 dim x(n)
40 for c=1 to n
50 input"next value ";x(c)
60 next c
70 print"    ";:rem ref point
80 for c=1 to n:printtab(c*3+3);c;:next
c
85 m=0
90 for c=1 to n
97 rem find max value
100 if x(c)>m then m=x(c)
110 next c
115 rem rescale other values
120 for c=1 to n
130 let x(c)=x(c)*16/m
140 next c
150 for h=0 to 16
152 rem plot histogram
155 print ""
160 FOR C=1 TO N
170 IF X(C)>=H THEN PRINT TAB(C*3+4);"&"
;
180 NEXT C
190 NEXT H
195 REM WRITE HEADING
200 PRINT"";spc((40-len(t$))/2-1);"";t$
210 print""
```

```
220 for c=16 to 0 step -1
230 print int(m*c/16*10+.5)/10
240 next c
499 end
500 print"";"                    histogram "
510 return
```

There are times, however, when the order of data is wanted not according to time (as with the business profits) but in order of magnitude the kids' heights). If there is a lot of data to put into ascending order (so that values get bigger) program 101 will do it. Just tell the computer how many values there are, enter them one at a time, and the computer will print them out in ascending order.

Program 101 ORDERED DATA

```
1 REM PROGRAM 101
2 REM ORDERED DATA
10 PRINT""
20 INPUT"  NUMBER OF VALUES ";N:DIM D(N)
:FOR X=1 TO N:INPUT" NEXT VAL.";D(X)
25 NEXT X
30 FOR X=1 TO N-1:F=0:FOR Y=1 TO N-1:IF
D(Y)<=D(Y+1) THEN 50
40 Q=D(Y):D(Y)=D(Y+1):D(Y+1)=Q:F=1
50 NEXT Y:IF F=0 THEN END
60 FOR Z=1 TO N:PRINT;D(Z);" ";:NEXT Z:P
RINT:PRINT:NEXT X
```

And that is the end of the short maths course. You should have learned a lot about your Commodore as well as about maths; as you carry on playing with your machine, thinking of new ways to use it and writing your own programs, you will be surprised how often this apparently theoretical stuff will be of use. Good luck!

Appendix 1
USER-DEFINED GRAPHICS

There follows some lists of data that can be used to define sets of useful characters that do not already exist on the micro.

LOWER CASE GREEK LETTERS

alpha	Ø,	Ø,	18,	44,	68,	74,	5Ø,	Ø
beta	56,	36,	56,	36,	36,	6Ø,	32,	Ø
gamma	Ø,	49,	74,	12,	2Ø,	2Ø,	8,	Ø
delta	28,	32,	16,	24,	36,	36,	24,	Ø
epsilon	Ø,	Ø,	28,	32,	56,	32,	28,	Ø
zeta	22b	8,	8,	16,	56,	4,	24,	Ø
eta	Ø,	88,	36,	36,	36,	4,	4,	Ø
theta	56,	68,	68,	124,	68,	68,	56,	Ø
iota	8,	Ø,	16,	16,	16,	2Ø,	8,	Ø
kappa	Ø,	Ø,	72,	8Ø,	96,	8Ø,	72,	Ø
lambda	Ø,	32,	16,	8,	8,	2Ø,	34,	Ø
mu	Ø,	36,	1ØØ,	36,	6Ø,	34,	32,	Ø
nu	Ø,	4,	2,	54,	2Ø,	8,	8,	Ø
xi	16,	28,	16,	28,	32,	24,	4,	56
omicron	Ø,	Ø,	56,	68,	68,	68,	56,	Ø
pi	Ø,	Ø,	62,	84,	2Ø,	36,	34,	Ø
rho	Ø,	12,	18,	18,	44,	32,	64,	Ø
sigma	Ø,	Ø,	Ø,	62,	72,	72,	48,	Ø
tau	Ø,	Ø,	62,	72,	8,	16,	16,	Ø
upsilon	Ø,	Ø,	1ØØ,	36,	36,	24,	Ø,	Ø
phi	8,	8,	28,	42,	28,	8,	8,	Ø
chi	Ø,	34,	84,	8,	2Ø,	37,	66,	Ø
psi	8,	8,	1Ø6,	42,	28,	8,	8,	Ø
omega	Ø,	Ø,	32,	66,	82,	44,	Ø,	Ø

UPPER CASE GREEK LETTERS

ALPHA	Ø,	24,	36,	36,	6Ø,	36,	11Ø,	Ø
BETA	12Ø,	36,	36,	56,	36,	36,	124,	Ø
GAMMA	126,	34,	32,	32,	32,	32,	112,	Ø
DELTA	8,	8,	2Ø,	2Ø,	34,	34,	127,	Ø
EPSILON	126,	34,	4Ø,	56,	4Ø,	34,	126,	Ø
ZETA	126,	7Ø,	12,	24,	48,	96,	126,	Ø
ETA	119,	34,	34,	62,	34,	34,	119,	Ø
THETA	62,	65,	85,	93	85,	65,	62,	Ø
IOTA	56,	16,	16,	16,	16,	16,	56,	Ø
KAPPA	11Ø,	36,	4Ø,	48,	4Ø,	36,	118,	Ø
LAMBDA	8,	8,	2Ø,	2Ø,	34,	34,	119,	Ø
MU	99,	54,	62,	42,	42,	34,	119,	Ø
NU	1Ø3,	34,	5Ø	42,	38,	34,	115,	Ø
XI	66,	126,	Ø,	126,	Ø,	126,	66,	Ø
OMICRON	56,	68,	68,	68,	68,	68,	56,	Ø
PI	119,	34,	34,	34,	34,	34,	119,	Ø
RHO	12Ø,	36,	36,	56,	32,	32,	112,	Ø
SIGMA	126,	34,	16,	8,	16,	34,	126,	Ø
TAU	127,	73,	8,	8,	8,	8,	28,	Ø
UPSILON	99,	34,	2Ø,	8,	8,	8,	28,	Ø
PHI	28,	8,	62,	42,	62,	8,	28,	Ø
CHI	119,	34,	2Ø,	8,	2Ø,	34,	119,	Ø
PSI	28,	73,	1Ø7,	42,	62,	8,	28,	Ø
OMEGA	28,	34,	65,	65,	54,	85,	119,	Ø

MISCELLANEOUS MATHEMATICAL SYMBOLS
Powers and Superscripts

squared	112,	16,	112,	64,	112,	Ø,	Ø,	Ø
cubed	112,	16,	112,	16,	112,	Ø,	Ø,	Ø
4	64,	8Ø,	12Ø,	16,	16,	Ø,	Ø,	Ø
5	112,	64,	112,	16,	112,	Ø,	Ø,	Ø
−2	24,	8,	1Ø4,	8,	8,	Ø,	Ø,	Ø
−2	14,	2,	11Ø,	8,	14,	Ø,	Ø,	Ø
−3	14,	2,	11Ø,	2,	14,	Ø,	Ø,	Ø

One character square will take two of these small numbers across, and small subscripts can be defined by moving the symbol to the bottom of the square instead of the top.

Other symbols useful in maths programs

½	66,	68,	72,	87,	33,	71,	132,	7
dx	Ø,	Ø,	16,	16,	117,	82,	117,	Ø
∂	96,	16,	8,	6Ø,	68,	68,	56,	Ø
∇	Ø,	127,	34,	34,	2Ø,	2Ø,	8,	8
∫	Ø,	Ø,	Ø,	Ø,	Ø,	Ø,	12,	18,
	16,	16,	16,	16,	8,	8,	8,	8
	72,	48,	Ø,	Ø,	Ø,	Ø,	Ø,	Ø

Appendix 2
A SERIES FOR PI

There exists a series (an infinite series) whose sum is PI, and which (once we know about it) will allow us to calculate PI to as many places as we please ... if we're patient enough!

It goes like this:

PI=4(1/1−1/3+1/5−1/7+1/9−1/11+1/13−...)

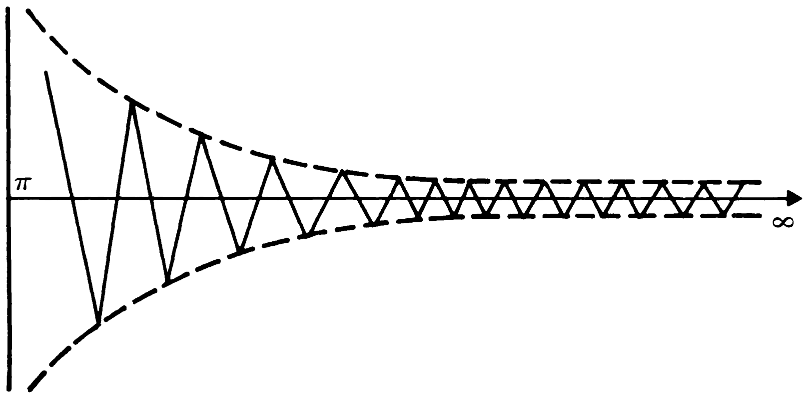
Notice that it's basically the series of odd fractions, alternating positive and negative, and the whole show multiplied by four.

Try this:

Appendix Program 4   PI SERIES

```
1 REM APPENDIX PROGRAM
2 REM PI SERIES
10 PRINT""
20 PRINT"";"pi=";3.1415927:s=0:for n=0 t
o 10000:k=1
30 if n/2<>int(n/2) then k=-1
35 print "";"n=";n
40 q=k/((2*n)+1)
50 s=s+q:print "";"our approximation=";s
*4
55 print""
60 NEXT N
```

Don't be alarmed if the number oscillates for several minutes; it is trying to converge to PI, but even with ten thousand terms

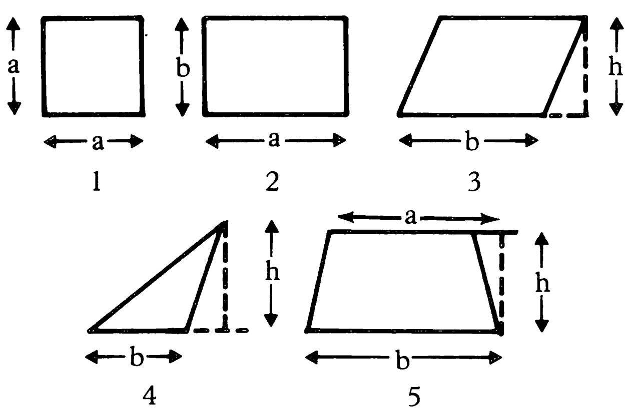


this series is so slow to converge that it takes a while for the oscillations to die out in the fourth decimal place! There are much faster converging series in existence, but you might like to see this one in action.

Appendix 3
GEOMETRIC SHAPES

Plane Figures

1. Square: Area=a^2 (where a is the length of a side)
2. Rectangle: Area=a*b (where a and b are sides)
3. parallelogram: Area=b*h (where b is the side and h is the vertical height)
4. Triangle: Area=½*b*h (where b=base and h=height)
5. Trapezium: Area=½*(a+b)*h (where a and b are the lengths of the parallel sides and h is the height)



6. Circle: Area=PI*r^2=PI*d^2/4 (where r is the radius Circumference=2*PI*r=PI*d and d is the diameter)
7. Annulus (Ring) Area=(PI/4)*(D*D–d*d) (where D is the diameter of the larger circle and d that of the smaller)

8. Ellipse Area=PI*d/4 (where D is the major axis length and d is the minor axis)

Solids

1. Sphere: Volume=(PI*d↑3)/6=$\frac{4}{3}$*PI*r↑3

   Surface Area=PI*d↑2=4*PI*r↑2
2. Right Cylinder: Volume=Area of base*vertical height
   Surface Area=Perimeter of base*height+2*Area of base
3. Oblique Cylinder: Area of base*perpendicular height=Volume
   S. Area=perimeter of perpendicular slice*length of side
4. Right Cone: Volume=area of base*⅓(perpendicular height)
   Surface area=perimeter of base*½ sloping length+area of base
5. Oblique cone: Volume=area of base*⅓(perpendicular height)

This book provides an introduction to mathematics for the non-mathematical person who has access to the Commodore 64. With the help of over 100 short programs, the book shows the ability of the micro to perform fast calculations of a laborious kind and to draw illustrative and animated graphs and diagrams. The style of the book is very informal and refreshingly different to the dry textbook/revision booklet approach.

The book will appeal, in addition to people of limited mathematical ability (or people whose maths are rusty), to 'O' and 'A' Level pupils and to teachers wishing to illustrate mathematical principles and techniques.

ISBN 0-7126-0612-2

9 780712 606127

£7.95

ISBN 0 7126 0612 2

CENTURY COMMUNICATIONS LTD

MATHS TUTOR FOR THE COMMODORE 64

ROBERT CARTER