DRAGON magic

Your First Programming Book

Richard Wadman

Dragon Magic

Your First Programming Book

by Richard Wadman

Contents

Introduction

Today computers are everywhere – in the schools, shops, offices and your own home. This book is to help you learn to use a computer.

With your DRAGON computer you will be able to play games, draw pictures and do your arithmetic.

Work through the book slowly, and make sure that you understand each section before moving on to the next. If there is anything you don't understand, ask somebody else – a teacher, a parent, or someone at the shop where you bought the computer. If you have any difficulties that they cannot answer write to us at:

Dragon Data Limited
Kenfig Industrial Estate
Margam
Port Talbot
SA13 2PE
West Glamorgan

MY DRAGON CHEATS AT TENNIS BUT IS GREAT AT ARITHMETIC!

Chapter 1
Computers, programs and programmers

Since the very beginning, one of the things that has made humans different from animals is the ability to invent tools. The first tools were used to assist the physical skills of man – to help him build shelters, make clothes and hunt his food.

Much later, machines were invented to help the thinking skills of man – first adding machines, then *computers*.

Computers cannot think or have bright ideas like us, but they can do arithmetic very much faster than we can.

Though computers can look as if they are very clever, they are still only tools. They cannot do anything at all unless they are given instructions. The instructions given to a computer are called a *program*.

The people who write these instructions and run the computer are called *programmers*. This book is to help you to become a programmer.

Programs are written in a special language understood by computers. Most small computers understand a language called BASIC. This language is very easy to learn, because it uses English words that you already know. This book will

USING BASIC.

help you to learn enough BASIC to get you started. The full BASIC language is explained in your Dragon manual.

At the end of each chapter we will give a list of things to remember from the chapter we have just finished. Here is the list for this chapter.

Things to remember from this chapter.

1. A computer is a tool to help people to think.

2. Computers do not have ideas of their own.

3. Computers can do arithmetic much faster than people, but they have to be told how to do it.

4. The instructions to a computer are called a program.

5. People who write programs for computers are called programmers.

6. Programs for small computers are usually written in a language called BASIC.
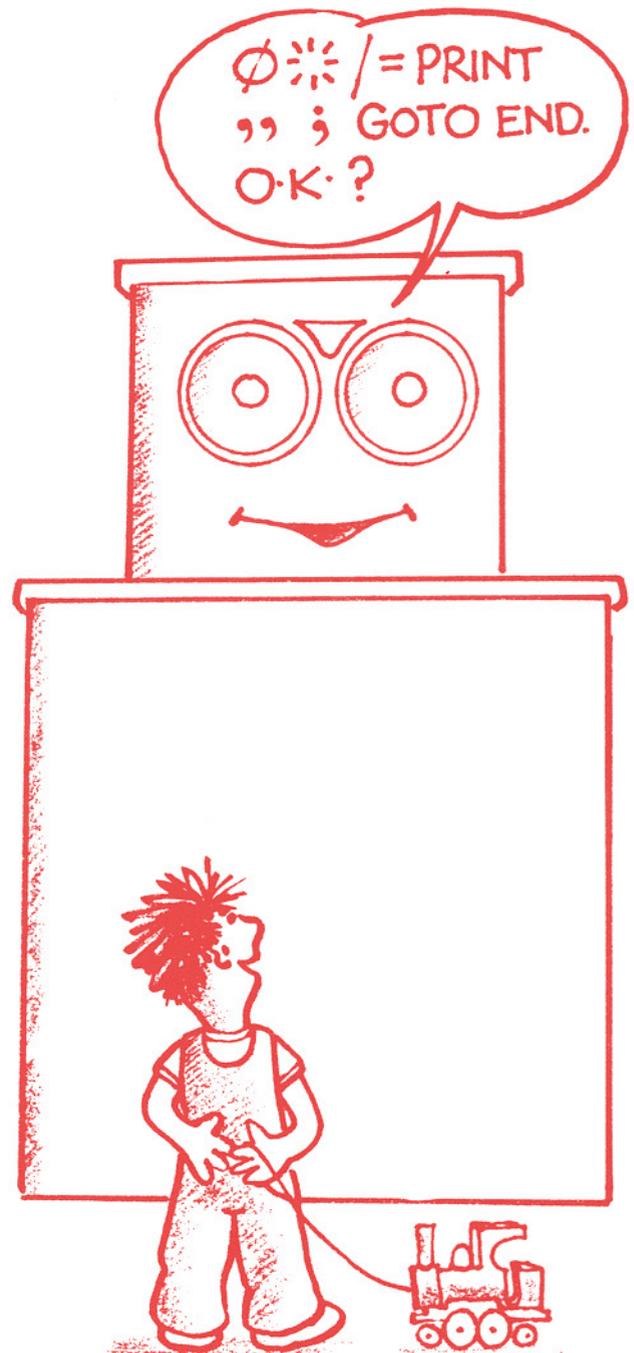
Chapter 2
Program order, flowcharts and loops

The computer can only do one thing at a time, so a program has to be made up of small steps. People sometimes use programs to help them to remember to do things properly. We all learn a program to help us cross the road. The most well known is the Green Cross Code, this tells us the steps we must follow to cross the road safely. It also gives us the order of each step, because order is important as well.

Imagine a program for getting dressed in the morning. One step may be to put on your shoes, another step can be to put on your socks. Your friends would think you were funny if you did those two steps in the wrong order. You may think that this is a silly example because everyone knows that you put your socks on first. Computers don't know – they don't really learn anything. They have to be told what to do and what order it must be done in. This is what a program does.

So a program is a list of steps in the right order to get the job done. When you start to write a program, you could just make a list of all the steps. A big job could have a program with many hundreds of steps and it can be very difficult to keep track of where you are.

One way of doing this is to draw a special type of picture called a *flowchart*. A flowchart shows the steps and the order in which they must be carried out. The picture uses different shapes to show what kind of step it is. You write the step inside the shape.
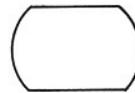
Oval
This shape is used to show where the program begins or ends. The word inside is START or END.
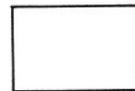
Rectangle
This shape contains one complete step.

Diamond
This shape means a decision has to be made. It contains a question. The answer to the question must be YES or NO. The answer can only be YES or NO. *Sometimes, maybe*, or *when I feel like it* are not allowed.

The shapes are joined by arrows showing you where to go next.

Here is a flow chart showing you how to cross the road.
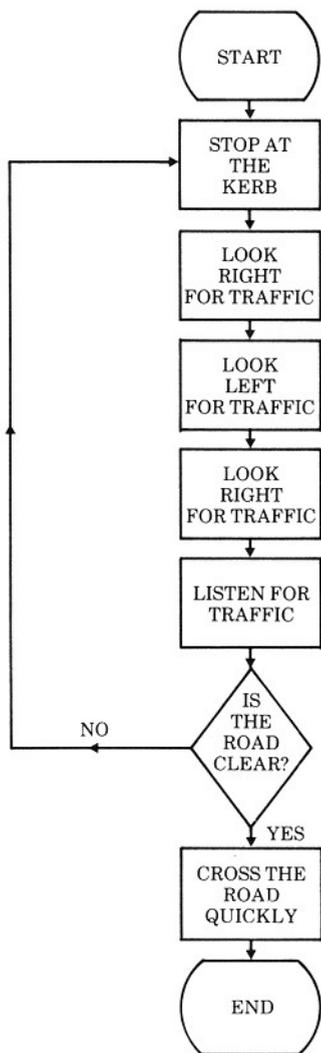
You can see that the diamond shape has two arrows coming out of it. One is for the YES answer, the other is for the NO answer. Each rectangle has one step inside telling you exactly what to do. You must do what a rectangle says, you have no choice.

A flow chart must always have a start and an end. There must always be an arrow showing you where to go next. You cannot have a 'dead end', a shape with an arrow going in, but no arrow coming out.

```
        ┌──────────┐
        │  START   │
        └──────────┘
             │
        ┌──────────┐
   ┌───▶│ STOP AT  │
   │    │   THE    │
   │    │  KERB    │
   │    └──────────┘
   │         │
   │    ┌──────────┐
   │    │  LOOK    │
   │    │  RIGHT   │
   │    │FOR TRAFFIC│
   │    └──────────┘
   │         │
   │    ┌──────────┐
   │    │  LOOK    │
   │    │  LEFT    │
   │    │FOR TRAFFIC│
   │    └──────────┘
   │         │
   │    ┌──────────┐
   │    │  LOOK    │
   │    │  RIGHT   │
   │    │FOR TRAFFIC│
   │    └──────────┘
   │         │
   │    ┌──────────┐
   │    │LISTEN FOR│
   │    │ TRAFFIC  │
   │    └──────────┘
   │         │
   │       ╱IS╲
   │      ╱ THE ╲
   NO◀───◀ ROAD  │
         ╲CLEAR?╱
          ╲   ╱
           │ YES
        ┌──────────┐
        │CROSS THE │
        │  ROAD    │
        │ QUICKLY  │
        └──────────┘
             │
        ┌──────────┐
        │   END    │
        └──────────┘
```

8
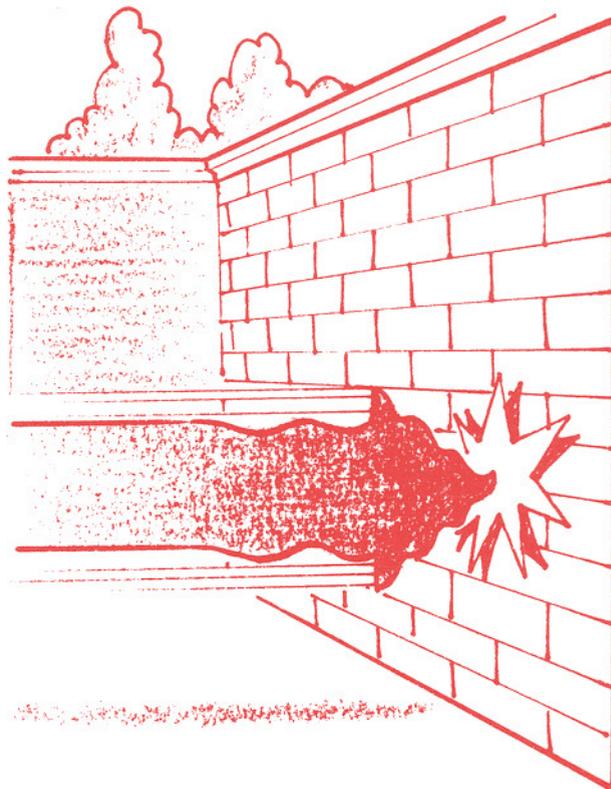
See if you can find the 'dead end' in the next flow chart. You should be able to follow the arrows through the flow chart until you find what is wrong.

START

FIND A PACKET OF SWEETS

IS IT EMPTY → NO → EAT A SWEET

YES

STILL HUNGRY? → YES

NO

END
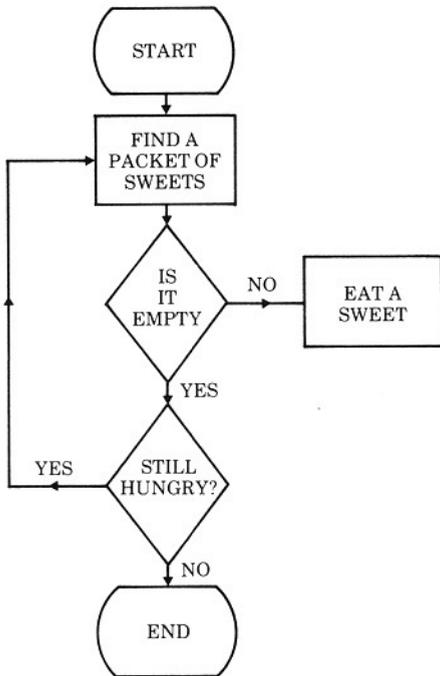
The answer is the EAT A rectangle.
                   SWEET
There should be an arrow coming out of it. Where should the arrow go to? See if you can work it out before turning the page.
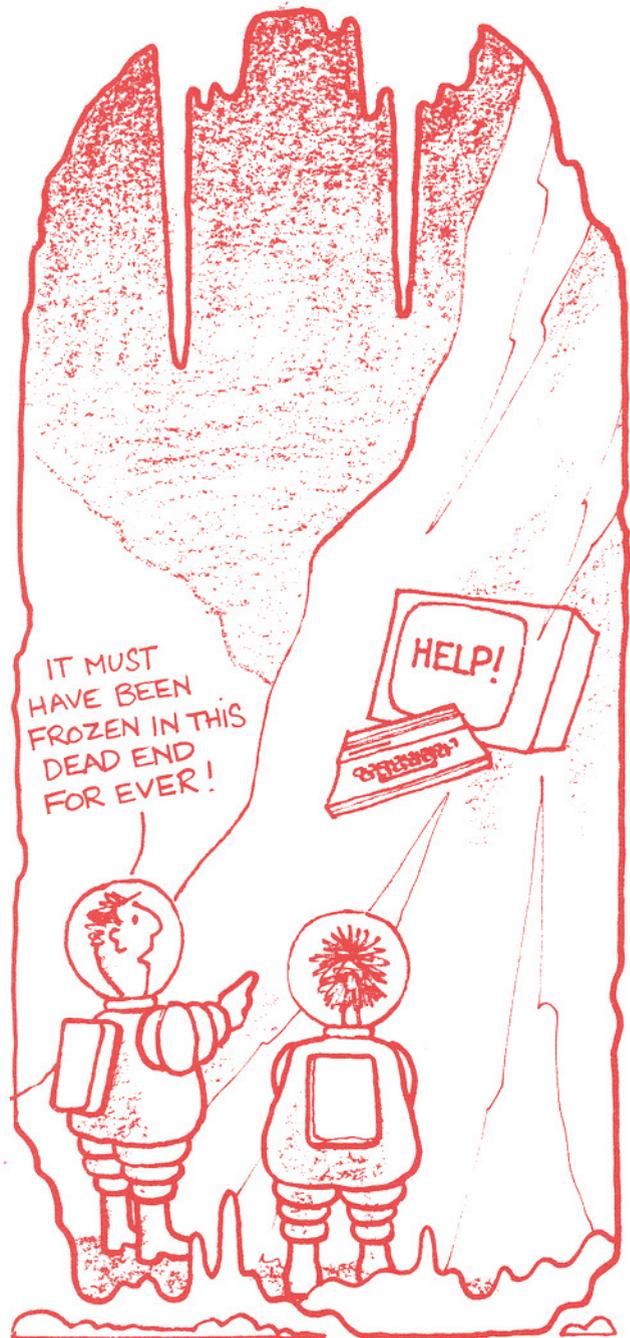
This is only part of the flowchart, but it shows where the arrow coming out of the 'dead end' should go.
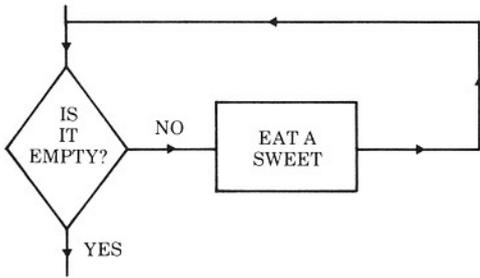
Try practising your own flowcharts. Pick a subject that you know well to start with. Then move onto subjects that you have to think about. Here are some problems you might like to try.

1. Getting dressed for school.
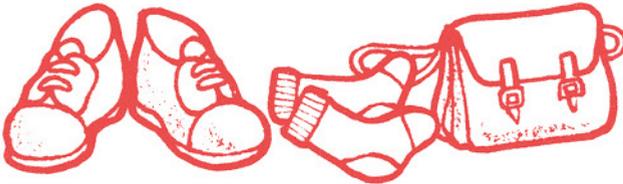
2. Taking the dog for a walk.
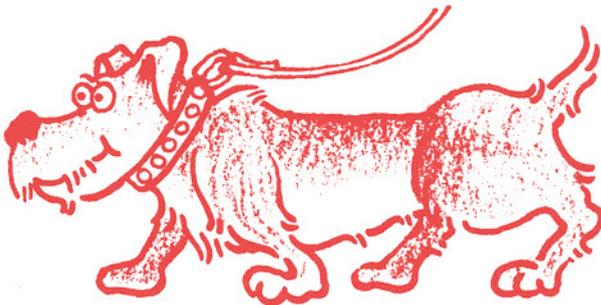
3. Baking a cake.

4. Buying a birthday card for mother.
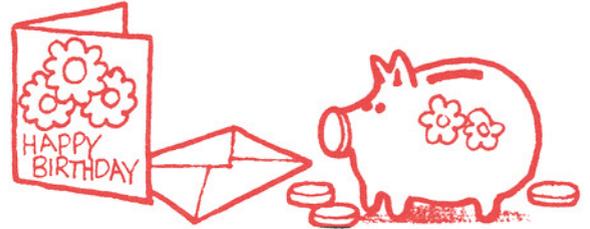
5. Mending a puncture on your bicycle.
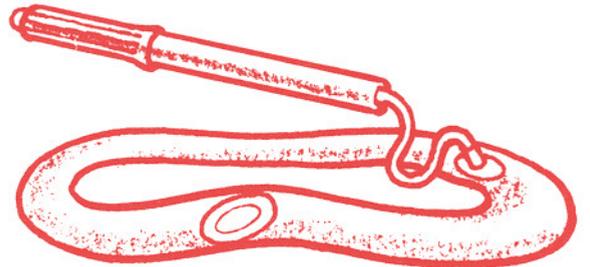
6. Getting a rhinoceros into the bath.
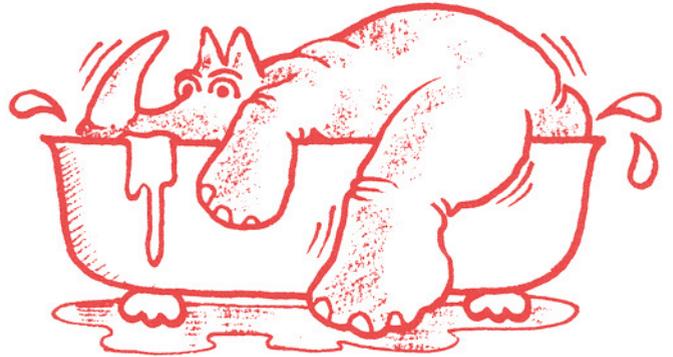
Here is one answer to problem 5, I hope you have a better one!
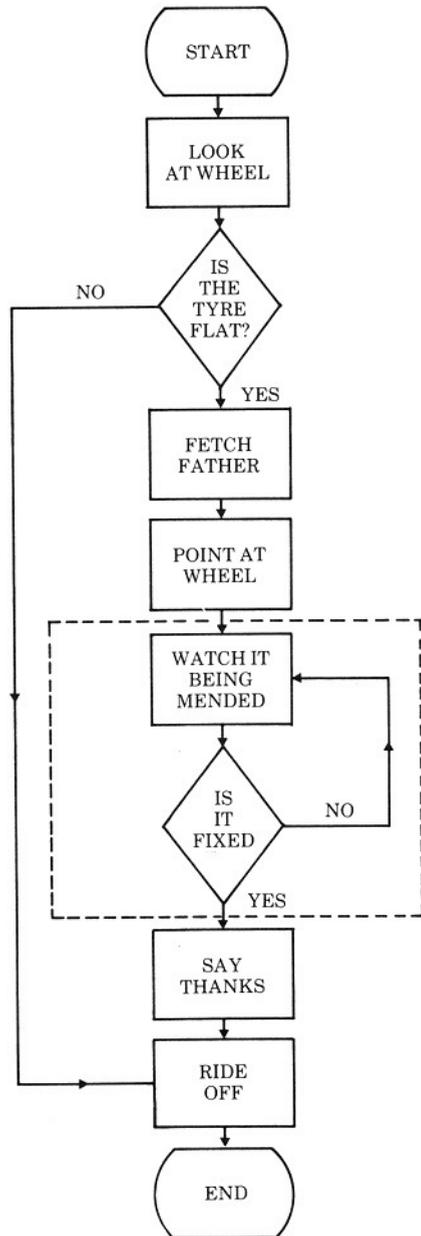
In this flow chart we have put a dotted line around a bit in the middle. This part is called a *loop*. This is because the arrows make you do something over and over again. You keep coming back to the questions asked until the answer is YES, in this case.
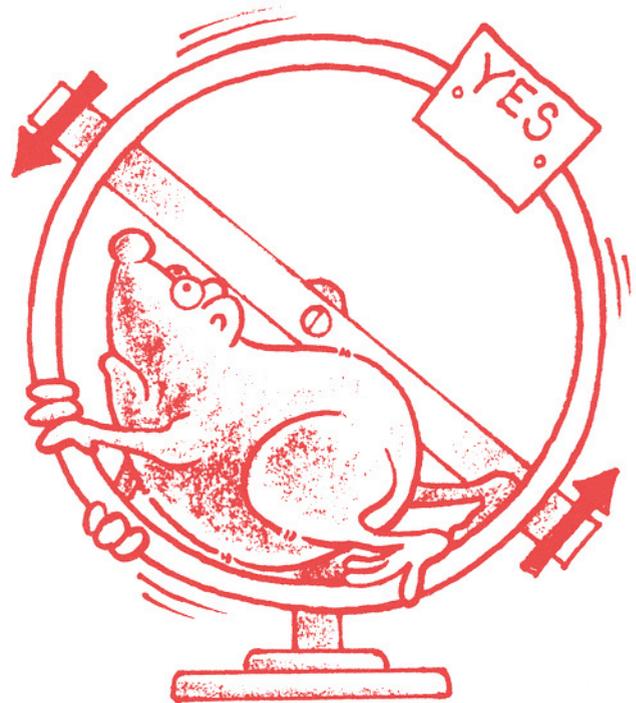
Try the problems, or some of your own, and make sure you understand the reason for flowcharts. Later you will see how to use them to help you write computer programs.

Things to remember from this chapter.

1. A program is a list of small steps to do a job, or solve a problem.

2. The order of the steps is important.

3. Flowcharts are pictures of how the program steps are ordered.

4. The oval shape is used to show where the program starts and ends.
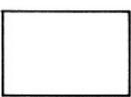
5. The rectangle shape is used for one complete step.

6. The diamond shape is for decisions. It contains a question which must have a YES or NO answer.

7. The shapes in a flowchart are joined by arrows.

8. There must be no 'dead end' in a flowchart. The arrow must always show where to go next.

9. When part of a flowchart is repeated over and over it is called a loop.

THE ARROW MUST ALWAYS SHOW WHERE TO GO NEXT

Chapter 3
Setting up your computer

Let us now take a look at your computer. There are three parts which make up the system.

The television set
This is just an ordinary television set. Your computer can draw pictures in colour, but if you want to see them in colour you must use a colour television.

The computer
This has a keyboard at the front just like a typewriter. The rest of the computer is inside. There is no need to take the top off at all. It is not a good idea, because you may damage yourself and the computer.

The cassette recorder
You don't need a cassette recorder to make the computer work. It is used to keep programs so you do not have to type them into the computer every time you want to use them. Any cassette recorder will do. It plugs into the computer using a special cable.

If you have a recorder that you can use with your computer, then read Chapter 4 in the manual which comes with your DRAGON computer.
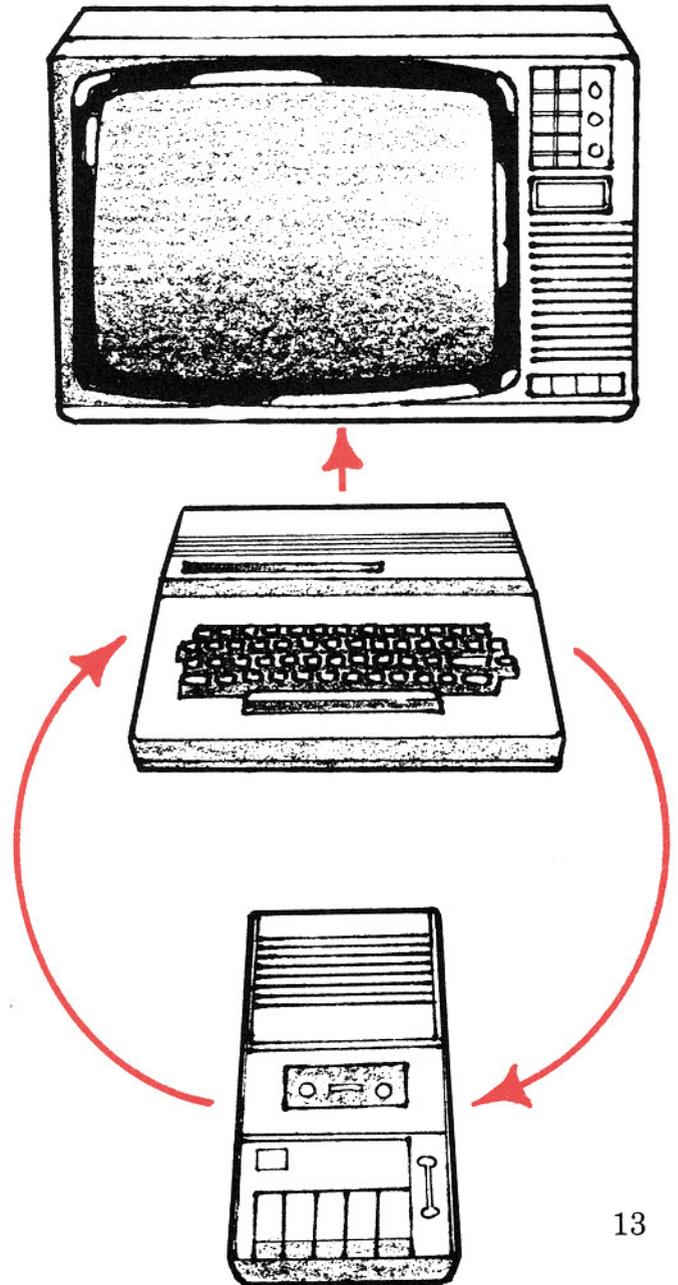
The first time you use the computer get an adult to help you connect all the plugs. It is not hard once you know how, and most of the plugs will only go in one way. Make sure the mains plug is out of the wall socket before making any connections. Keep all wires tidy, so that no one can trip over them or pull them out by accident. When you are sure that everything is correct, put the mains plug into the socket and switch on.

The TV screen should show a large green square with a black border, (or a light grey square with a black border if you are using a black and white TV set). On the green square you will see a message about the *interpreter* in the computer.

The interpreter is the part of the computer which translates your instructions into signals which the
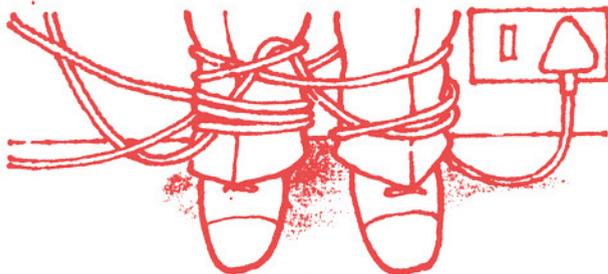
computer can understand. Underneath the message is the word OK and a flashing rectangle.


```
(c) 1982 DRAGON DATA LTD
16K BASIC INTERPRETER 1.0
(c) BY MICROSOFT

OK
■
```
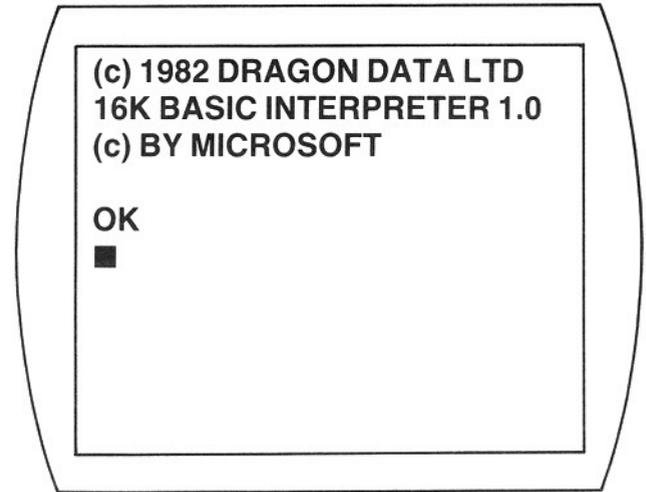
The OK is called the *prompt*. This is the computer's way of telling you it is ready to do something for you.

The flashing rectangle is called the *cursor*, this shows you where you will be typing next.

If you look on the left side of your computer, towards the back you will find a button. This is called the RESET button. If you press it, the computer will stop what it is doing, and wait for you to tell it what to do. You can then start all over again, it is very useful if things go wrong.

KEEP WIRES TIDY.

Now look at the keyboard. You will see that it is like a typewriter and has all the letters and numbers on it. Some of the keys have two things written on them like $"_2$ or $^+_;$. To get the top one on the key you have to hold down the key marked SHIFT. Try it, type some 2s and then hold down the SHIFT key and type the 2 again. You will see the 2s on the screen followed by the " symbol. This symbol is called the quotation mark, we shall be using it later.
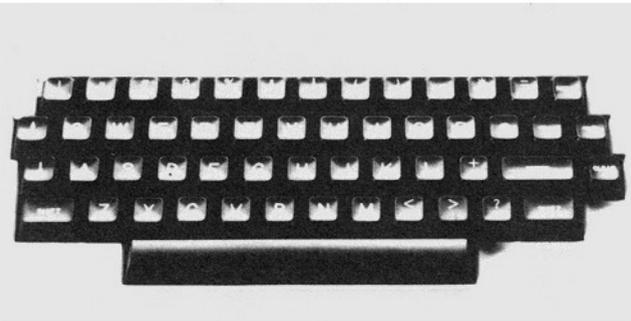
Throughout this book you will see the number zero, or nought, written like this: 0. This is because computers are very fussy about the difference between the letter O, and the number 0. On an ordinary typewriter you can use the letter O instead of the number if you wish, but on a computer you must always press the correct key. If you look on the keyboard at the end of the top row you will see the key for the number, 0.

Practice with the keyboard for a while. Type some letters then press the ← key. This is the backward arrow or backspace key. When you press this key the cursor moves to the left, rubbing out the letters. You can use this key to correct your mistakes. If you press the key marked CLEAR, then all the letters are rubbed out and you start at the top of the screen again.

If you make a typing mistake when writing a program, the computer will tell you there is an error. The message you are most likely to see is:

?SN ERROR

This means you have made a syntax or grammatical error, and the computer cannot understand what you mean. This is usually because you have spelt a command wrongly, or used O instead of 0. All the other error messages are listed at the back of the manual which came with the computer.

15

Things to remember from this chapter.

1. Do not poke around inside the computer or the television set.
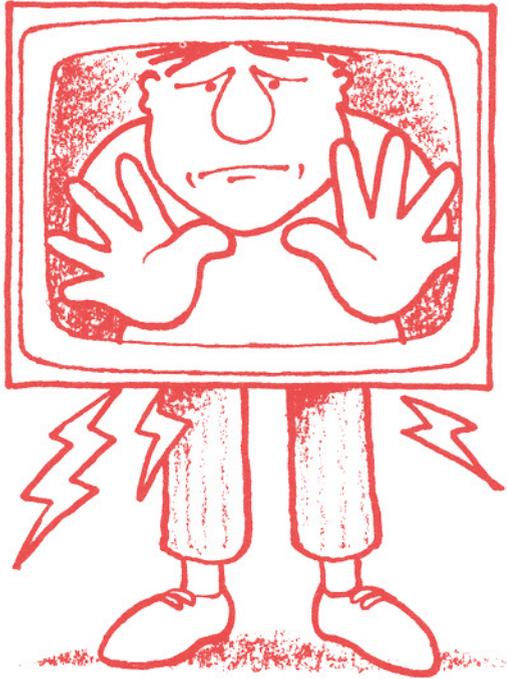
2. Make sure the mains plug is out of the wall socket before connecting the TV, cassette and computer together.

3. Keep all the wires tidy and out of the way.

4. The OK on the screen is called the prompt. It means the computer is ready to work.

5. The flashing rectangle is called the cursor. It shows you where you will be typing next.

6. The keyboard is like a typewriter. You will have to hold down the key marked SHIFT to type in the symbols on the top of the keys.

7. You can correct mistakes by using the backspace key ← to rub out the letters.

8. Pressing the CLEAR key rubs out all the writing on the screen.

9. One extra thing to remember all the time: switch off everything when you have finished using the computer.

Chapter 4
Printing numbers and words

Now we are ready to make the computer do some work. The first BASIC word we shall use is PRINT. This means print something on the TV screen. Use the CLEAR key to start on a new page, then type in:

PRINT 2 + 2

Remember you have to use the SHIFT key to get the + sign. Now press the key marked ENTER. The computer comes back with the answer straight away, writes it on the next line and then gives the OK prompt. Your screen should look like this.

```
PRINT 2 + 2
4
OK
■
```
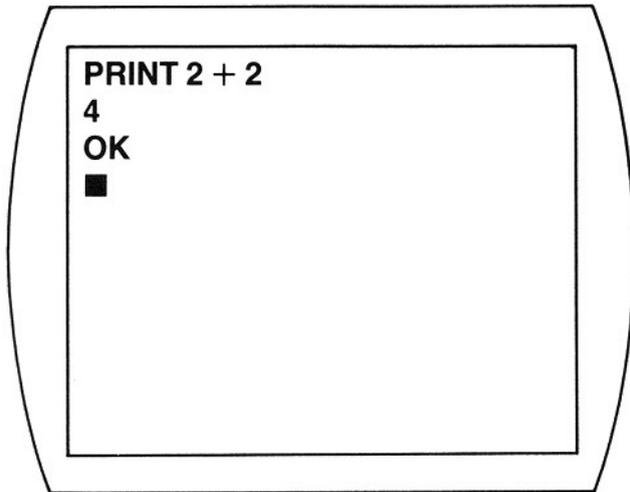
That was too easy, you don't need a computer to add two and two, try another one.

PRINT 6987 + 1498

and press the ENTER key. This time your screen will show:

```
PRINT 6987 + 1498
8485
OK
■
```
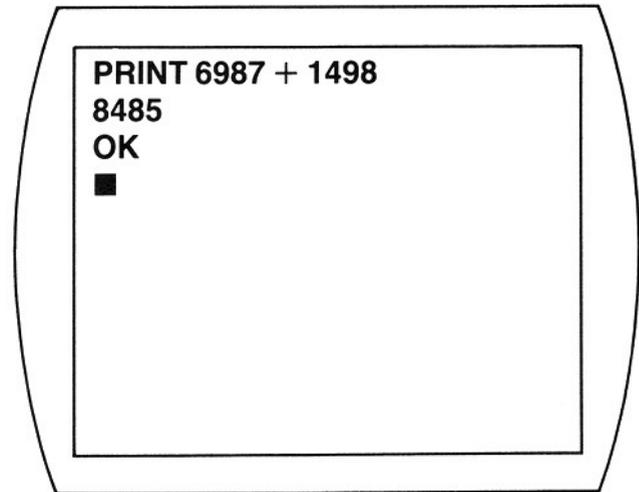
You are using the computer like a calculator. It will also do subtraction, multiplication and division. Try some of the following addition and subtraction sums and remember to press the ENTER key after each line.

17

PRINT 38 − 17
PRINT 4 − 2 + 6
PRINT 18 + 9 − 7

To do multiplication the computer uses the * sign, meaning times, so 5 times 4 is written 5 * 4. (You have to use the SHIFT key to get this symbol as well.)

To do division it uses the / sign, so 12 divided by 3 is written 12/3.

Here are some more sums to try:

PRINT 15/3
PRINT 8/4
PRINT 21 * 6
PRINT 4 + 8 * 3
PRINT 65783 * 27

You will notice that the computer does nothing until you press the ENTER key. When you press the ENTER key it tells the computer you want it to do something.

The OK prompt from the computer means that the computer is waiting for you to do something.

There is no need to hurry when you are working with your computer. The computer is very patient, and will wait until you are ready.
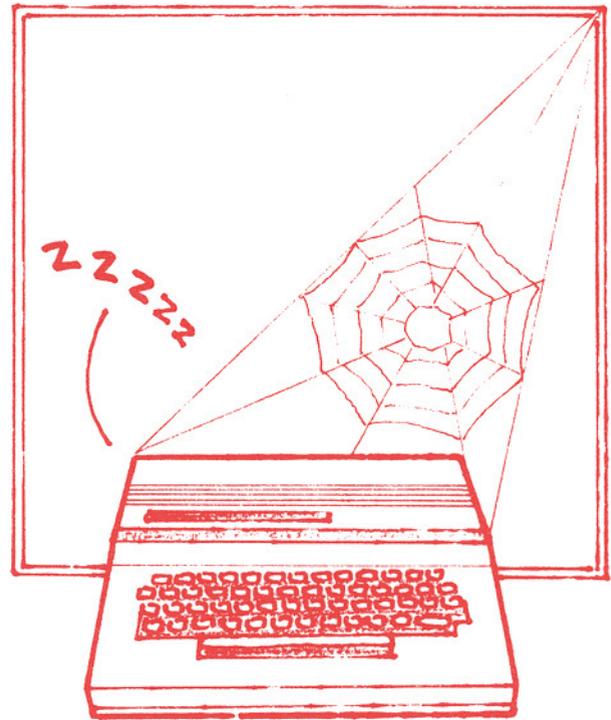
THE COMPUTER IS VERY PATIENT!

That's enough arithmetic for the moment, let us do something else.

You can print words as well as numbers. To tell the computer the difference between words and numbers we use the quotation mark ", which needs the SHIFT key again. Press the CLEAR key to start a new page, then copy this line.

PRINT "HELLO"

Don't forget to type the quotation marks and to press the ENTER key. Your screen should look like this.

```
PRINT "HELLO"
HELLO
OK
■
```
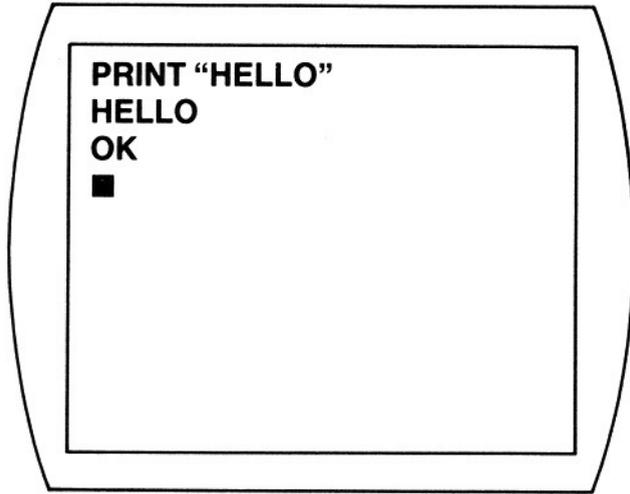
The computer will print out exactly what you put between the quotation marks, no matter what it is. If you spell the words incorrectly, the computer will not correct them for you.

Things to remember from this chapter.

1. The PRINT command prints on the screen. It can also be used to do simple arithmetic.

2. To add numbers together use the + sign.

3. To take away or subtract numbers use the − sign.

4. To multiply numbers use the * sign.

5. To divide numbers use the / sign.

6. The OK prompt means the computer is ready for you to type something in.

7. Pressing the ENTER key tells the computer you have finished typing.

8. The PRINT command can be used to print words or sentences.

9. So that the computer can tell that you are using words and not numbers, words must always be in quotation marks (").

Chapter 5
Entering and running a program

So far we have only used the computer as a calculator. It is time we started to write a program. You remember in Chapter 2, we said a program is a list of steps in the order they are to be done in.

The easiest way to keep the steps in order is to number them 1, 2, 3 and so on. This is what we do when we write a program using the BASIC language. Each step is a line in the program and we give every line a number to show which order it is to be done in.

You don't have to number the lines 1, 2, 3 because the computer will start with the lowest number and work upwards. So we can number the lines in tens if we want to: 10, 20, 30; or in fives: 5, 10, 15. (There is another good reason for this which you will see later.) Here is a small program:

10 PRINT "I AM YOUR DRAGON 32 COMPUTER"
20 PRINT "WHAT IS YOUR NAME?"
30 END

See how each line has a number? The numbers for each line must be different,

you cannot have two lines with the same numbers.

The first step in this program will be line number 10, because it is the lowest number. The next step will be line 20 and the last step is line 30.

The PRINT in this program is the same as the PRINT we were using in the last chapter, but this time it has a line number in front of it. This is to show the computer it is part of a program.

The last line of the program, line 30, has a new command word, END. This tells the computer the program has finished and there are no more steps to do.

Try typing the program into the computer. The first thing to do is to make sure that the computer's memory is empty. Do this by typing NEW and pressing the ENTER key. This tells the computer to get ready to store a new program. The NEW does not have a line number in front of it, because it is not part of the program.

When you have told the computer you are going to put in a new program, you then type in the program. Type each line as it is in our example and remember to press the ENTER key at the end of each line.

Do not worry if the line you are typing is too long to fit into one line on the screen, it will just continue on the next line. The computer expects the end of the line to be the pressing of the ENTER key.

After you press the ENTER key this time, the computer will not do anything. The number at the beginning of the line tells the computer it is part of a program, so it just stores it in its memory until you are ready.

To make the computer carry out the program, type RUN and press the ENTER key. The RUN is like the NEW command, it does not have a number in front of it.

DON'T WORRY IF THE LINE IS TOO LONG

Your screen should look like this.


```
10 PRINT "I AM YOUR DRAGON
   32 COMPUTER"
20 PRINT "WHAT IS YOUR
   NAME?"
30 END
RUN
I AM YOUR DRAGON 32
COMPUTER
WHAT IS YOUR NAME?
OK
```

If you press the CLEAR key now, the TV screen will be cleared, but the program will still be in the computer's memory.

To see the program again type LIST and press ENTER. The LIST command tells the computer to print out the program that is in its memory.

If you want to add another line to your program all you have to do is to type in the line with a number that will put it in the right place. Type in this line:

5 CLS

Remember to press the ENTER key at the end of the line.

21

As the number 5 is smaller than 10, this line will become the first step in the program. If you type LIST now you will see the computer has put the new line in the right order.

The new word CLS means clear the screen. It is like the CLEAR key, but this time is part of the program. If you RUN the program now it will clear the screen before printing the message.

By now you should be getting used to pressing the ENTER key after each line, so we won't tell you every time from now on. So far we have used two types of command words. Those without a line number are obeyed at once. Those with a number are stored in the memory.

Things to remember from this chapter.

1. A program is a list of steps the computer must do to solve the problem.

2. Each step is one program line.

3. Every line is numbered to show the order it must be done in.

4. Before you put a program into the computer type NEW to clear the memory.

5. To put a program into the computer's memory, you type the line number and the line, and then press the ENTER key.

6. The first line of the program is the one with the lowest number.

7. The last line of the program has the highest number and the word END.

8. To start a program type RUN. RUN does not have a line number.

9. To see the program in the computer's memory, type LIST. LIST does not have a line number.

10. The CLS word is used to clear the screen in a program.

Chapter 6
Variables and assignment statements

When we typed in the program in the last chapter it was stored in the computer's memory. We can also store numbers and words in the memory and call them back when we want.

The computer's memory is divided into very small sections. You could imagine the memory to be a lot of very small boxes, each one of which can hold only one number. To tell the computer which box you want to use you have to give it a label, or name. The name must begin with a letter. The full name can be two letters, or a letter and a number. In computer programs, the letter names you give to these boxes are called *variables*. They are called variables because their value can be changed. Here are some examples of names for *variables*.

A, D3, M4, XX, GB, K9, Z, Y1, Y2

To put a number into a variable you use an *assignment statement*.

20 A = 6

This means put the number 6 into a place in memory called A.

You can use variables on both sides of an assignment statement, but the left side of the equals sign (=) must be a variable name.

40 D = A + 2

This program line means look into the place in memory called A, take the number that is there, add 2 to it, and put the answer into the place called D. Of course you should have put something into A earlier in the program. If you haven't put anything there, then the number in A will be zero (0).

You can change the number in a variable at any time with another assignment statement. Each variable can only hold one number, so if you do change the number then the old number is lost.

Here is an example using assignment statements:

```
10 CLS
20 A = 2
30 B = 3
40 PRINT A + B
50 A = 6
60 PRINT A + B
70 END
```

Type the example into the computer, remember to type NEW and press ENTER before you start with the program. After you have typed in the program, type RUN to start it.

The first PRINT in line 40 will give 5 as the answer, the second PRINT in line 60 gives 9 as an answer. This is because the number in A was changed in line 50.

There is another way to put numbers into variables, this is from outside the program.

The INPUT word allows you to put a number into a variable while the program is running. This command allows you to 'talk' to the program by using the keyboard. The INPUT word is followed by the name of a variable. When the program reaches a line with the INPUT word, it stops and prints a question mark (?) on the screen. This means the computer is waiting for you to type in a number and press the ENTER key. The number you type in will be put into the variable after the INPUT word. The program will then carry on with the next line.

Type NEW and press ENTER, then put in this example:

```
10 CLS
20 PRINT "TYPE IN A NUMBER"
30 INPUT A
40 PRINT "THE NUMBER WAS"
50 PRINT A
60 END
```

Every time you RUN the program, it will stop and wait for you to type in a number. Type in any number and then press ENTER. The program will carry on and print out the number you have just entered.

24
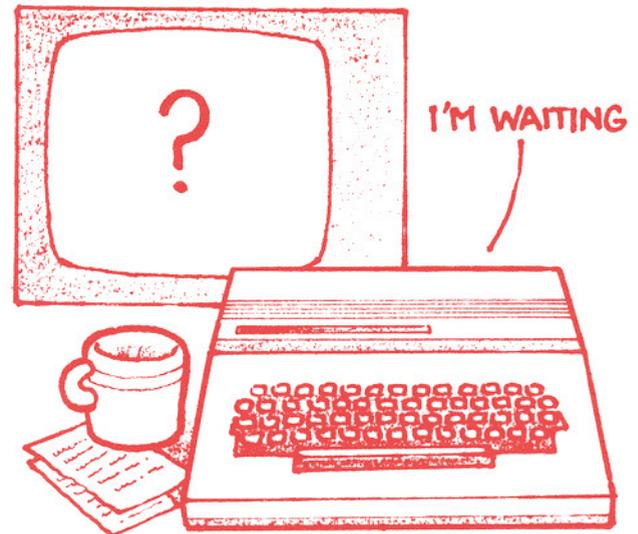
To save you typing RUN every time here is a new BASIC word, GOTO. GOTO tells the computer to go to the program line number which follows the GOTO word. Add another line to the last example, but don't type NEW first, because you will clear the program from memory if you do.

55 GOTO 10

This new line will fit between line 50 and line 60. When the program reaches line 55 it will go back to line 10 and start all over again. When you run the program this time it will keep asking you for another number, again and again. Every time the program reaches line 30 and a new number is typed in, it is put into the variable A. The number that was in variable A before is lost.

To stop the program you will either have to switch off (which means you will lose the program), or press the key marked BREAK. The BREAK key tells the computer to stop what it is doing. It stops the program and gives the OK prompt. The program is still in the memory.

Repeating part of a program like this is called a loop. We mentioned loops in chapter two, when we were talking about flowcharts. If you draw the flowchart for the last example, you will see why it

never stops, because there is no arrow to the END shape.

25
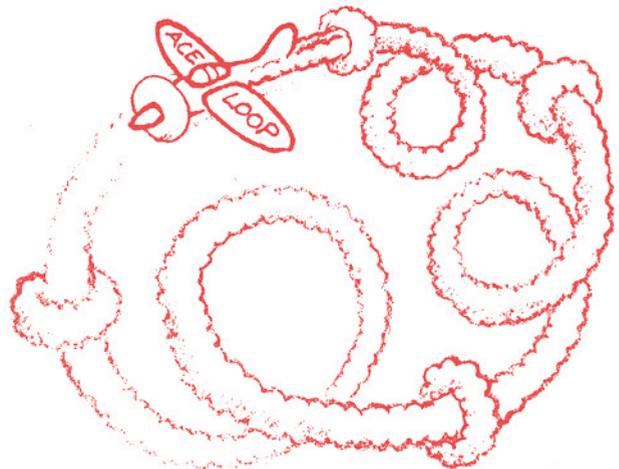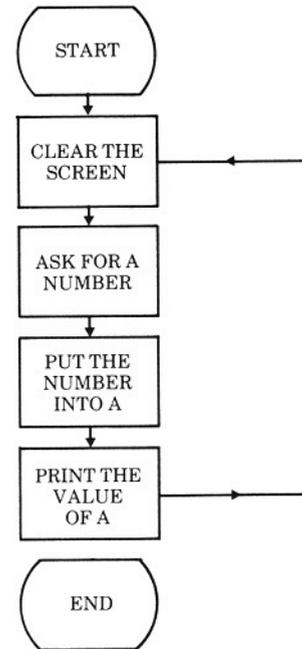
So far the only type of variable we have been using is for numbers, but there are also variables for words. The variables for words are kept in a different place in the computer's memory, because they must not be mixed with numbers.

The names for variables where we keep words are nearly the same as those for numbers. The difference is that the word variables have a dollar sign ($) at the end of the name. Here are some examples of variable names for keeping words:

A$, M6$, AB$, NA$, X$, W$

Variables that are used to keep numbers are called *numeric variables*. Variables that are used to keep words are called *string variables*, not because they are made of string, but because they are used to store strings of characters.

You can use string variables in assignment statements, but you must remember to use the quotation marks.

Here is an example using string variables.

```
10 B$ = "DRAGON 32"
20 PRINT "WHAT IS YOUR NAME"
30 INPUT A$
40 CLS
50 PRINT "HELLO"
60 PRINT A$
70 PRINT "MY NAME IS"
80 PRINT B$
90 END
```

See if you can work out what it is doing before running the program.

STRING VARIABLES

Things to remember from this chapter.

1. The computer's memory is like a lot of small boxes. Each box can hold only one number.

2. To tell the computer which box you are using, you must give the box a name.

3. The names must begin with a letter (from A to Z). The next part of the name can be a letter or a number.

4. The names you give to these boxes are called variables.

5. To put a number into a variable you use an assignment statement. An assignment statement is a variable name, followed by an equals sign (=), and then the number you want to put into the variable.

6. The left hand side of the assignment statement must be a variable name.

7. You can put a number into a variable while the program is running by using the INPUT command. This stops the program and waits for you to type the number.

8. The GOTO command tells the computer to go straight to the line number following the word GOTO.

9. You can stop a program at any time by pressing the BREAK key.

10. There are special variables for storing words or letters. These are called string variables.

11. The names for string variables must end with a dollar sign ($).

Chapter 7
Controlling loops

In the last chapter we made a program loop back and repeat the program again and again. The only trouble with using GOTO like this is that we cannot stop the program without using the BREAK key. There is another way of making a program loop without using GOTO. This uses two special lines which allow you to decide how many times you want to loop.

Type NEW, to clear out any program in the memory, and then type in this example.

```
10 CLS
20 FOR A = 1 TO 10
30 PRINT "HELLO"
40 NEXT A
50 END
```
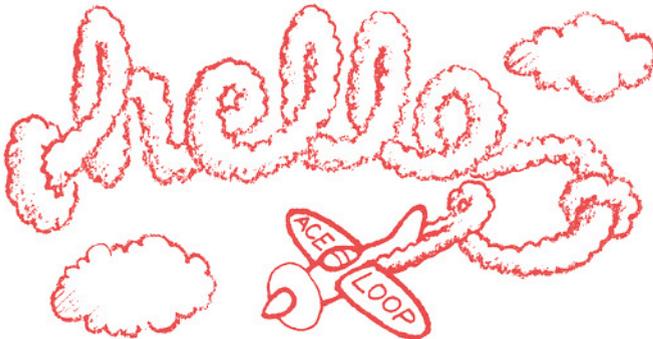
This little program will print HELLO on the screen ten times. The top of the loop is the special command FOR in line 20. The bottom of the loop is the special command NEXT in line 40. Here is a flowchart which shows you what is happening.
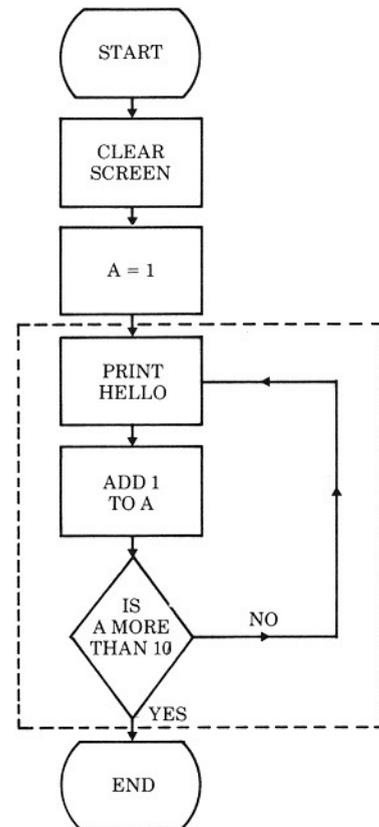
Lines 20 and 40 are the top and bottom of the loop. Line 20 means do all the lines from here to the line with a NEXT, ten times. The variable A is used to keep a count of how many times the loop has been done. If you change the program a little you can see it counting. Type in the line below:

30 PRINT A

You can't have two lines with the same number in a program, so the new line 30 replaces the old line 30. If you run the program now it will print the count in variable, A. Line 40 checks the counter to see if it has finished, if it hasn't it sends the program to line 20.

The variable in the FOR line must be the same as the variable in the NEXT line.

You can do many useful things with FOR NEXT loops, as they are called.

Here is an example which will add up as many numbers as you want.

```
10 PRINT "HOW MANY NUMBERS
   DO YOU WANT TO ADD"
20 INPUT N
30 S = 0
40 FOR A = 1 TO N
50 PRINT "TYPE IN A NUMBER"
60 INPUT X
70 S = S + X
80 NEXT A
90 PRINT "THE ANSWER"
100 PRINT S
110 END
```

This time, the number of times the loop is done will be decided by the number you put into variable N.

If you can't work out what line 70 is doing then type in the extra line:

75 PRINT S

and that should help you to understand. The variable S is being used to hold the total.

So far we have used the PRINT command to print out only one thing at a time. If we use a comma (,) to separate the list of numbers in a PRINT, then the numbers will be printed in two columns on the screen. The following example shows how the comma works.

```
 5 CLS
10 FOR I = 1 TO 10
20 PRINT I,2 * I
30 NEXT I
40 END
```

We can also use a semicolon (;) with the PRINT. This time instead of starting a new line or moving to the next column the printing will carry on after the last thing printed. Change line 20 in the last example to:

```
20 PRINT I; "TIMES 2 IS"; 2 * I
```

and run the changed program. You will see that the numbers and the words are all together in a sentence on one line.

If we put together the FOR NEXT loop and the new PRINT command we can write a program to do multiplication tables.

```
 5 CLS
10 PRINT "WHAT NUMBER TABLE
   DO YOU WANT"
20 INPUT N
30 CLS
40 PRINT "MULTIPLICATION
   TABLE"
50 FOR I = 1 TO 12
60 PRINT I; "TIMES"; N; "IS"; N * I
70 NEXT I
80 END
```

You can use one loop inside another, but they must not overlap. The second loop must open and close inside the first.

Things to remember from this chapter.

1. A program can be made to loop as many times as you want using FOR NEXT loops.

2. The FOR NEXT loop uses a numeric variable as a counter.

3. The variable in the FOR line of the loop must be the same as the variable in the NEXT part of the loop.

4. All the lines between the FOR and NEXT will be done each time.

5. If you use a comma (,) in a PRINT command the numbers or words will be printed in two columns on the screen.

6. A semicolon (;) in a PRINT command will cause the numbers or words to be printed next to each other.

Chapter 8
Making decisions

If you take another look at Chapter 2, where we talked about flowcharts, you will see we have now done all the shapes except one. The one we haven't done is the diamond shape. This is the one used to show that a decision has to be made. If you look at the flowchart on page 8 which we drew to show us how to cross the road, you will see the decision part looks like this:

```
        │
NO      ◇ IS
 ◄──────  THE
         ROAD
         CLEAR?
           │
           │ YES
     ┌───────────┐
     │ CROSS THE │
     │   ROAD    │
     │  QUICKLY  │
     └───────────┘
           │
```
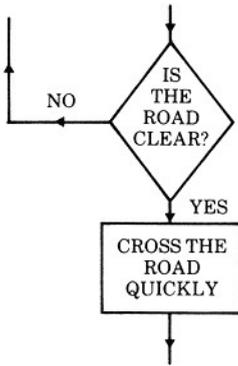
If you wrote this as an English sentence it would be something like this: '**If** there is no traffic on the road **then** cross quickly'. Your computer uses the same form to make its decisions. The BASIC language statement is called IF THEN. You translate it as follows: IF something is true THEN do something.

Here is a small example to show how IF THEN works. First a flowchart to show the steps in the right order.

```
        ┌─────────┐
        │  START  │
        └─────────┘
             │
     ┌───────────────┐
     │     PRINT     │
     │  INSTRUCTIONS │
     └───────────────┘
             │
     ┌───────────────┐
     │   ENTER TWO   │
     │    NUMBERS    │
     └───────────────┘
             │
           ◇ IS
            FIRST
           NUMBER        NO    ┌──────────────┐
          BIGGER THAN ────────►│ PRINT SECOND │
           SECOND             │   IS BIGGER  │
             │                │  THAN FIRST  │
             │ YES            └──────────────┘
     ┌───────────────┐               │
     │  PRINT FIRST  │        ┌─────────┐
     │   IS BIGGER   │───────►│   END   │
     │  THAN SECOND  │        └─────────┘
     └───────────────┘
```

31
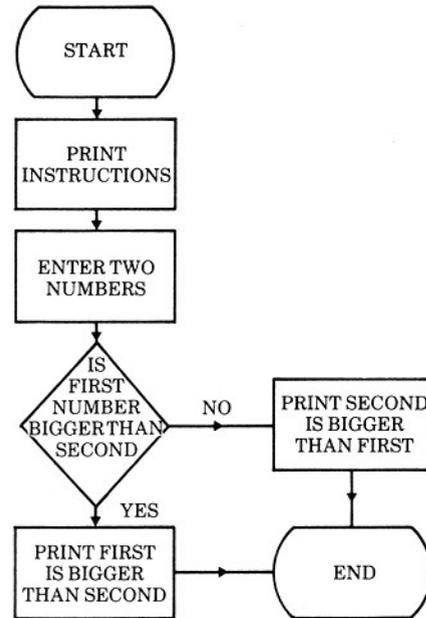
Here is the program.

```
  5 CLS
 10 PRINT "TYPE IN TWO
    NUMBERS"
 20 PRINT "SEPARATED BY A
    COMMA"
 30 INPUT A, B
 40 IF A > B THEN 70
 50 PRINT A; "IS SMALLER THAN";
    B
 60 GOTO 80
 70 PRINT A; "IS BIGGER THAN"; B
 80 END
```

There are some new things in this program. The first is the INPUT at line 30. This shows you how to enter more than one number at a time. You just have a list of variables and separate them by commas, the same way as we did before with the PRINT command.

When you run the program it will stop at the INPUT line and wait for you to put in the numbers, as before. This time it wants two numbers, so you can either type in each number and press ENTER after each one, or type in the two numbers separated by a comma.

The decision is made by the IF THEN at line 40. This line means, IF the number stored in variable A is bigger than the number stored in variable B THEN go to line 70. The program will

only go to line 70 if the A > B part is true, if it is not it will not do the rest of the IF line, it will carry on with the next line.

The > sign is the mathematical symbol meaning 'greater than'. Here are some others you can use in the decision part of the IF THEN statement.

$<$ less than or is smaller than
$=$ equal to or is the same
$<>$ not equal to or not the same

So in the example, line 40 tests to see if the number in A is greater than that in B. If it is the program continues at line 70. If A is not greater than B then it must be smaller so the test is not true and the program continues at line 50.

Can you work out why the program needs a GOTO in line 60? It is because you need to jump round line 70.

There is also a problem with this program. If you put in two numbers that are the same (like 8, 8), the answer will be that 8 is smaller than 8. This is silly, and it is because our program does not tell the computer what to do if the numbers are the same.

See if you can change the program so that this does not happen. You will need another IF THEN line, another PRINT line, and another GOTO line.

The IF THEN statement is very important, it is one of the things which make computers very different from calculators. IF THEN statements are used a lot in computer games to find out if the answer is right or wrong.

Before we give an example of how to use the IF THEN in a game, we will give you another new word that is also used a lot in games. This is the RND command. The RND tells the computer to pick a number at random. Picking a number at random is like putting a lot of pieces of paper with some numbers written on them into a hat, and pulling out one with your eyes closed.

To use RND you have to tell the computer which set of numbers to choose from. So RND(6) means it will only choose from the numbers 1 to 6. RND(10) means it will choose from the numbers 1 to 10. This example shows how RND works.

```
 5 CLS
10 FOR I = 1 TO 6
20 PRINT RND(10);
30 NEXT I
40 END
```

The semicolon (;) in the PRINT at line 20 causes the numbers to be printed on one line. When we ran the program it chose the numbers: 1 8 5 4 2 8. What

numbers did it choose when you ran the program? The numbers will be different each time.

Here is a program where you have to guess the number the computer has chosen. The program uses RND to 'think' of a number, and the IF THEN to check if your guess is right.

```
  5 CLS
 10 X = 10
 20 T = 0
 30 N = RND(X)
 40 PRINT "NUMBER GUESSING
    GAME"
 50 PRINT "I HAVE THOUGHT OF A
    NUMBER"
 60 PRINT "IT IS BETWEEN 1 AND";
    X
 70 PRINT "WHAT DO YOU THINK
    IT IS"
 80 INPUT A
 90 T = T + 1
100 IF N = A THEN 160
110 IF N < A THEN 140
120 PRINT "NO. IT IS BIGGER
    THAN"; A
130 GOTO 70
140 PRINT "NO. IT IS SMALLER
    THAN"; A
150 GOTO 70
160 PRINT "RIGHT!!"
170 PRINT "IT TOOK YOU"; T;
    "GOES"
180 END
```

This program only chooses numbers between 1 and 10. If you want to make it more difficult then change line 10. Try changing it to:

10 X = 100

and see how many guesses it takes you to find the number.
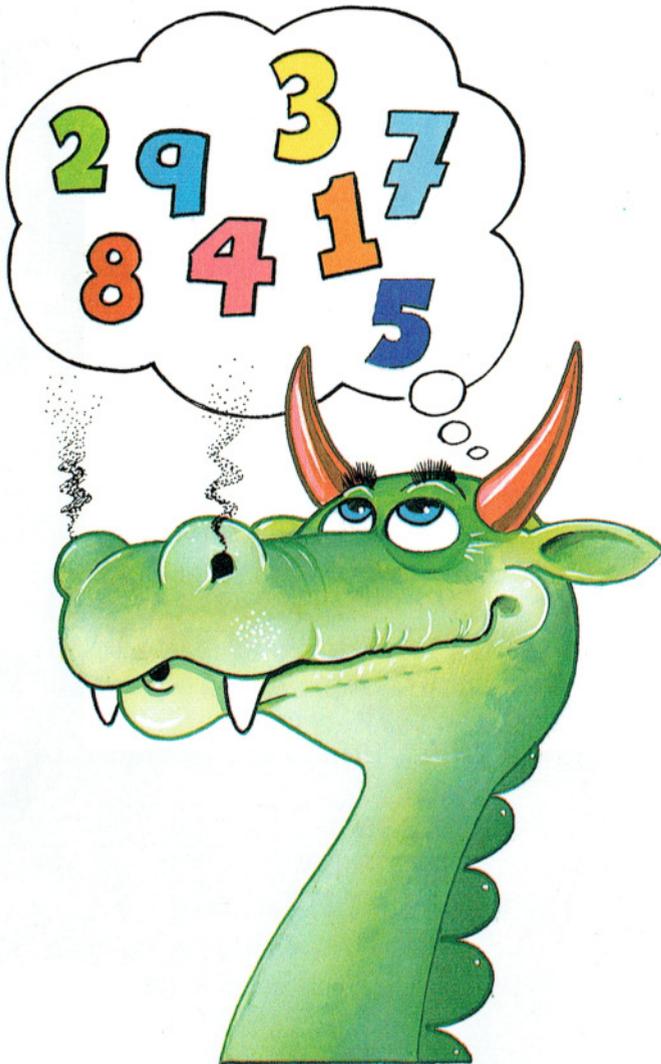
Things to remember from this chapter.

1. The IF THEN statement is used in a program to make decisions.

2. The IF THEN makes a test. IF the test is true THEN the rest of the statement is done.

3. If the test is not true the program carries on at the next program line.

4. The test uses these symbols.
   > greater than, or bigger than
   < less than, or smaller than
   = equal to, or same as
   < > not equal to, or not the same

5. You can put more than one number into the computer with the INPUT command. You have a list of variable names separated by commas. When you enter the numbers they are also separated by commas.

6. The RND word is used to choose a random number. The RND must have a number in brackets after it. This number tells the computer which numbers to choose from.

Chapter 9
Colour pictures

So far we have used words and numbers in our programs, now we will add some colour. Remember that you must be connected to a colour TV set. We have been using the CLS command to clear the screen to its normal colour, which is green. We can also use CLS to change the background to other colours. There are nine colours you can use, each one has its own number. These are the numbers for each colour:
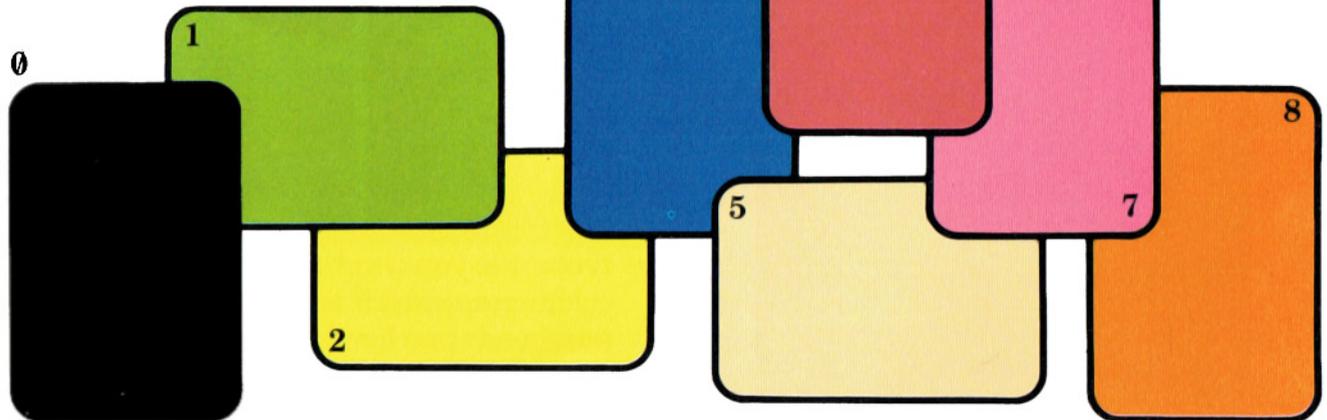
Here is a small program that will show you the background colours. If you do not put a number after CLS, the computer will use 1, green.

```
10 FOR C = 0 TO 8
20 CLS C
30 PRINT "THIS IS COLOUR"; C
40 FOR D = 1 TO 600
50 NEXT D
60 NEXT C
70 END
```

0 Black
1 Green
2 Yellow
3 Blue
4 Red
5 Buff
6 Cyan
7 Magenta
8 Orange

This program uses the FOR NEXT loop, which we met in Chapter 7, to go through all the colour numbers one at a time. The number is put into the variable named C. In line 20 CLS C means clear the screen and set the background colour to the number held in C. Line 30 prints a line at the top of the screen to tell you what the number is. Line 40 is a FOR NEXT loop which just counts from 1 to 600. This is because the computer works so fast you would not be able to see the colours properly – so we slow it down by making it count for a while.

The CLS can be used to change the whole screen to a different colour, but we can also change parts of the screen. To do this we have to imagine the screen is divided up into lots of small rectangles. There are 64 across the screen and 32 down, making it look like this.
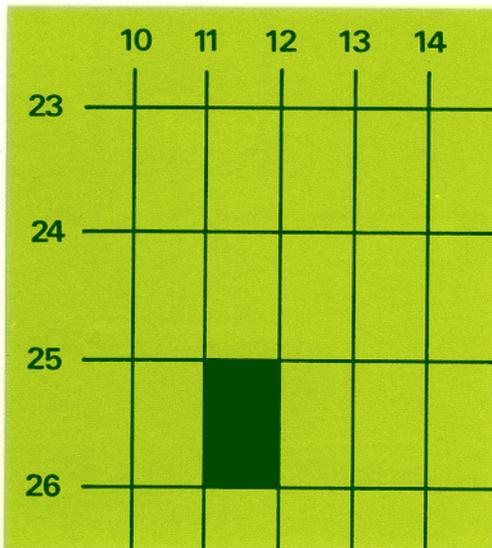
To tell the computer which rectangle you want to switch on you have to give the address of the rectangle. This is done by counting the number across the screen and the number down, just like a crossword. The computer always counts from the top left hand corner. This means the middle of the screen is 31 across and 15 down. The top left hand corner of the screen is 0 across and 0 down, and the bottom right hand corner is 63 across and 31 down.
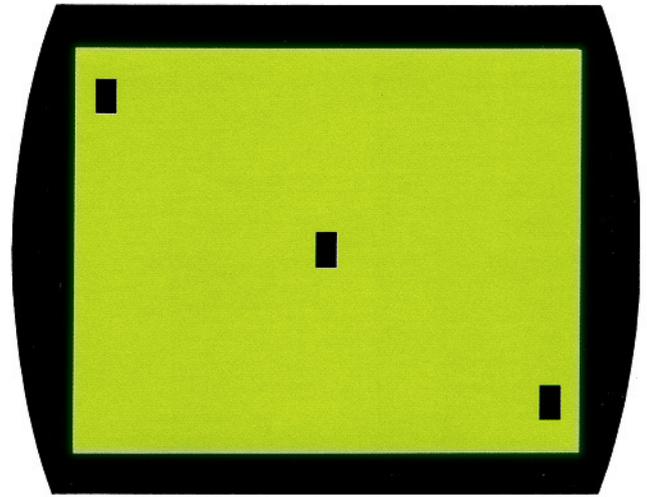

To make the rectangle the colour you want, use the SET command. The SET word is followed by the position of the rectangle you want to turn on, and the colour you want it to be. These three numbers, (two for the position and one for the colour), are always written in brackets ( ) in the same order.

This program will put an orange rectangle in the middle of the screen.

```
10 CLS0
20 SET (31, 15, 8)
30 GOTO 30
```

The first line clears the screen and sets the background to black. The next line, (20), will set the rectangle 31 across and 15 down to the colour number 8, which is orange. The last line (30) sends the program into an endless loop. It will keep going to line 30. This is to stop the OK prompt appearing on the screen. To stop the program, just press the BREAK key.
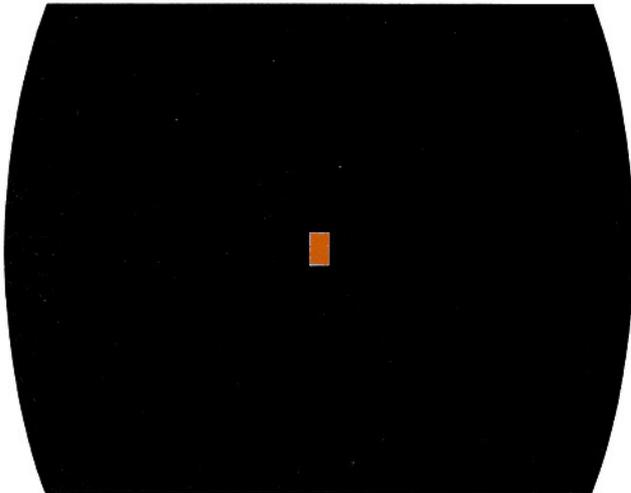
If you want to see what happens if you leave it out of the program, then type 30 and press the ENTER key. This is one way of taking a line out of a program. The

old line 30 is replaced by a blank line with the same number, and the computer ignores blank lines.

To switch the rectangle off again, we use the command RESET. By switching on and off, and choosing a different rectangle, we can make it look as though it is moving.

Here is an example of making the rectangle move. It uses the RND command to choose a rectangle.

```
10 CLS 0
20 X = RND(64)–1
30 Y = RND(32)–1
40 SET (X, Y, 8)
50 RESET (X, Y)
60 GOTO 20
```

We set the background to black in line 10 so that we can see the orange rectangle better. Use the BREAK key to stop the program.

By using the FOR NEXT loop we can draw lines on the screen. Here is a program that will draw a red cross on the screen, remember to type NEW before you enter it.

```
10 CLS 5
20 FOR Y = 0 TO 31
30 SET (31, Y, 4)
40 NEXT Y
50 FOR X = 0 TO 63
60 SET (X, 15, 4)
70 NEXT X
80 GOTO 80
```

```
25 SET (30, Y, 4)
35 SET (32, Y, 4)
55 SET (X, 14, 4)
65 SET (X, 16, 4)
```

Here is another flag, this time the French flag. Watch how your Dragon starts with a light background first, then draws the blue stripe, and then the red stripe. This time we use two loops to make the stripe thicker.
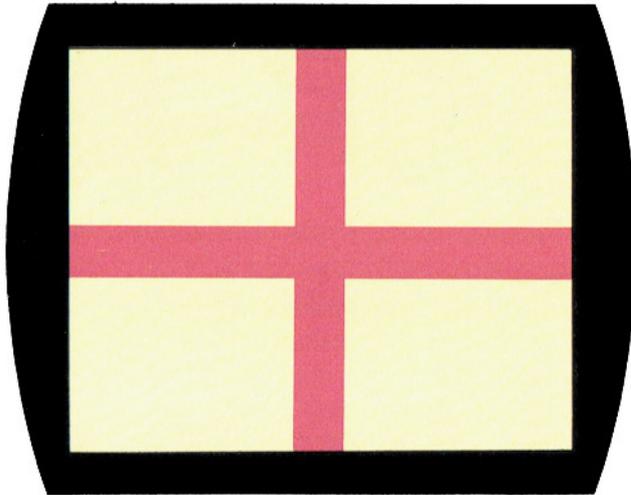
If you watch the screen carefully you can see how it works. The loop at lines 20 to 40 switches on the rectangles at 31 across for all the down positions, one at a time. This makes the red line down the middle of the screen. The next loop does the same, except this time across the middle of the screen.

The picture nearly looks like the English flag but the red lines are too thin. If you want to make them thicker, just add these extra lines to the program.
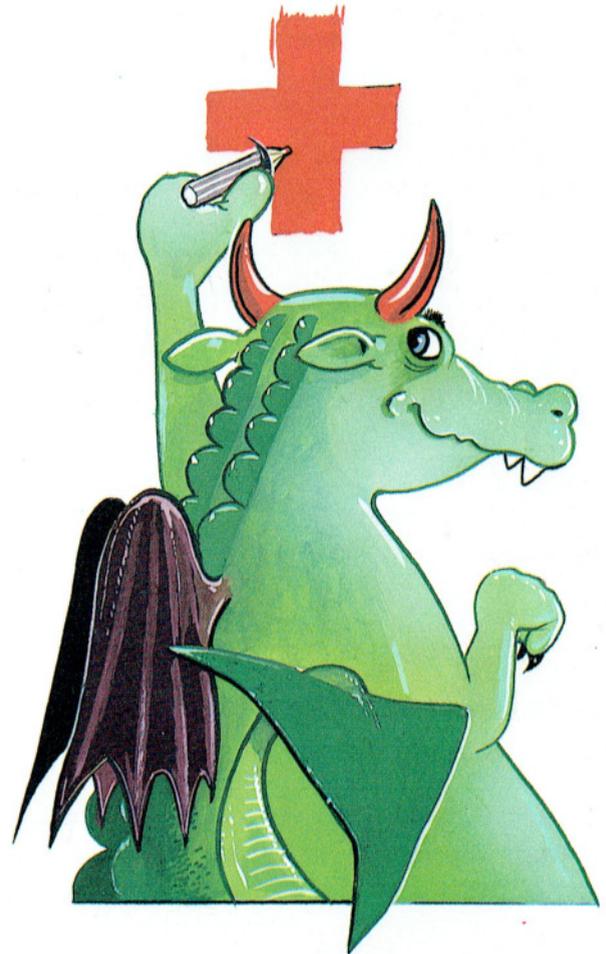
```
10 CLS 5
20 FOR X = 0 TO 19
30 FOR Y = 0 TO 31
40 SET (X, Y, 3)
50 NEXT Y
60 NEXT X
70 FOR X = 42 TO 63
80 FOR Y = 0 TO 31
```

```
90 SET (X, Y, 4)
100 NEXT Y
110 NEXT X
120 GOTO 120
```

Can you change this program so that the picture is the Italian flag? You only need to retype one line.

Things to remember from this chapter.

1. The CLS command can be used to change the background colour on the screen.

2. The colours all have a number. These numbers are:
   0 Black
   1 Green
   2 Yellow
   3 Blue
   4 Red
   5 Buff
   6 Cyan
   7 Magenta
   8 Orange

3. The SET command is used to switch on a small part of the screen.

4. To use the SET command, the screen is divided up into small rectangles, numbered 0 to 63 across the screen and 0 to 31 down.

5. When you use the SET command you must tell the computer which rectangle to switch on. This is done by how many across the screen and then how many down. You must tell the computer the colour number you want the rectangle to be.

6. To draw lines with the SET command it is easier to use a FOR NEXT loop.

7. The RESET command switches a point off.

39

In the last chapter we used the SET command to draw on the screen. This is all right for lines that go straight across the screen, or straight down, but is not very good for any other lines.

When you use the SET and RESET commands, the computer is working in what is called the low resolution mode. To draw better lines, it has to use the high resolution mode. The difference is like trying to draw with a thick crayon or a sharp pencil.

The high resolution mode is a lot more difficult to use, because you have to remember a lot more things.

The screen is divided into small rectangles, the same as before. This time, however, there are 256 rectangles across the screen and 192 down. This means the rectangles are much smaller than the ones we were using before. We can still switch rectangles on and off as before, but the commands are PSET and PRESET, instead of SET and RESET.
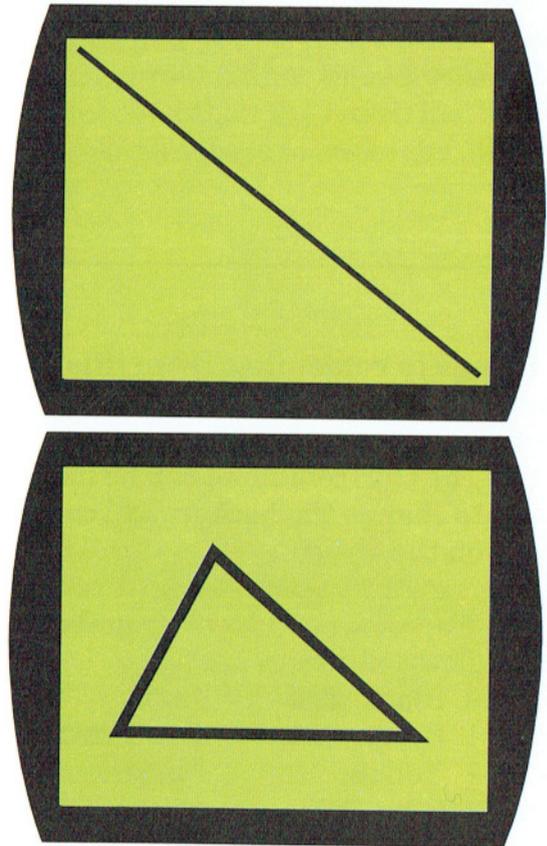
In the high resolution mode you do not need to use FOR NEXT loops to draw a line, as we had to in the low resolution mode. There is a special command for this: LINE. Of course, you have to tell the computer where you want the line to start and where it must finish.

Here is a program which will draw lines from corner to corner of the screen. The first line, (10), of the program sets the computer in the high resolution mode. There are a number of different mode and colour combinations available on your DRAGON computer. We shall only use this one in this book. To learn about the others, look in Chapter 8 of the manual that comes with the machine.

10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 LINE (0, 0) − (255, 191), PSET
**30 LINE (255, 0) − (0, 191), PSET**
40 GOTO 40

Line 20 draws a line from the top left hand corner of the screen to the bottom right hand corner, as in the illustration. The rectangle at top left is 0 across and 0 down, the bottom right is 255 across and 191 down. Look at line 20 carefully, and remember that the start of the line you want to draw and the end must be in brackets, and separated by a minus sign (−). The PSET must always be there as well. Line 30 draws the line from top right to bottom left.

Now that you can draw lines, you can make any shape you want, for instance a triangle.

10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 LINE (50, 150) − (200, 150), PSET
30 LINE − (100, 50), PSET
40 LINE − (50, 150), PSET
50 GOTO 50

This program draws a triangle in the middle of the screen. Line 30 is not wrong, it is a special form of the LINE command. If you miss out the start of the line then it will draw from the last position, but you must remember to put in the minus sign (−).

You can also draw squares and rectangles by using four lines, but there is an easier way. If you add , B onto the end of the LINE command it will draw a box. It uses the two points in the command as the positions of the opposite corners of the rectangle.

YOU MUST TELL THE COMPUTER WHERE THE LINE SHOULD FINISH

41

Now try this program:

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 LINE (50, 20) — (200, 150), PSET
30 GOTO 30
```

This will draw a diagonal line across the screen. Now change line 20 to:

```
20 LINE (50, 20) — (200, 150), PSET,
   B
```

and it will draw a rectangle instead of a line. Remember that all you have to do to change a line is to type in the new line and it will replace the old one.

There is one other thing you can do with the LINE command, you can fill the box with colour. Change line 20 again to:

```
20 LINE (50, 20) — (200, 150), PSET,
   BF
```

and the rectangle is now filled in with red.

NOW FILL THE RECTANGLE WITH COLOUR
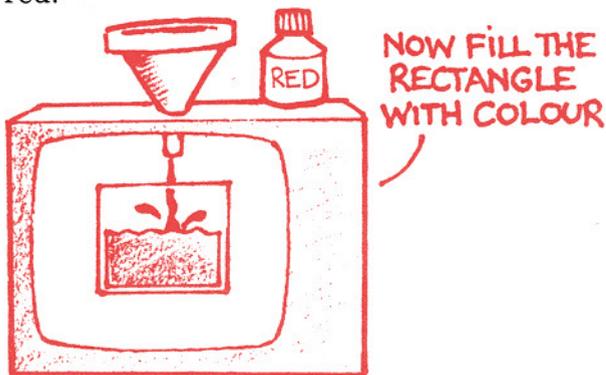
Here is a program that uses the LINE command to draw a picture.

```
10 PMODE 3, 1: SCREEN 1, 0:
   PCLS
20 LINE (250, 170) — (20, 170),
   PSET
30 LINE — (20, 190), PSET
40 LINE — (220, 190), PSET
50 LINE — (250, 170), PSET
60 LINE (180, 10) — (180, 170),
   PSET
70 LINE (176, 20) — (176, 165),
   PSET
80 LINE — (15, 165), PSET
90 LINE — (176, 20), PSET
100 LINE (184, 20) — (184, 165),
    PSET
110 LINE — (250, 165), PSET
120 LINE — (184, 20), PSET
160 GOTO 160
```

When you have put this program into the computer, then save it on the cassette, because we will add some more lines to it later.

Your computer makes it easy for you to draw pictures using the LINE command, but only for straight lines. There is also a command to draw circles for you: CIRCLE.

The CIRCLE command needs to know where the middle of the circle is, and how big you want it. The size of a circle depends on its *radius*.
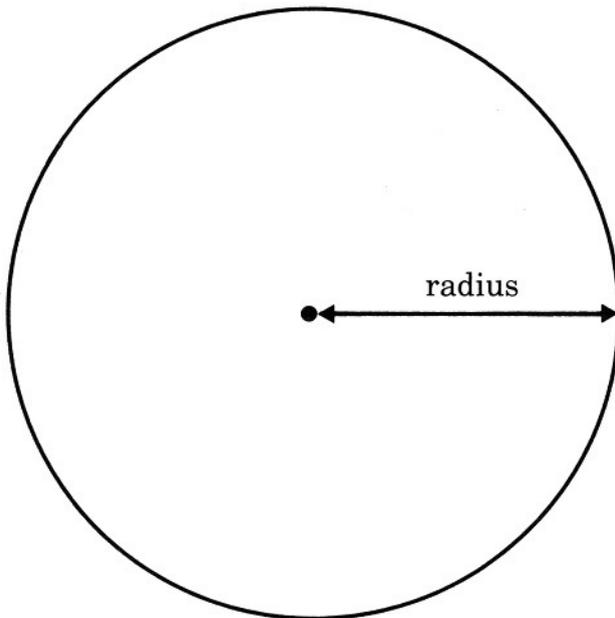
radius

This program will draw a circle in the middle of the screen.

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 CIRCLE (128, 96), 60
30 GOTO 30
```

The middle of the circle is 128 rectangles across and 96 down. This is the position in the brackets. The next number, 60, is the *radius* of the circle. This means the radius will be 60 rectangles in length.

You can use a variable name to replace any of the numbers in a LINE or CIRCLE command, this makes it easier to change them in a program.

Now try this program using the CIRCLE command.

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 FOR R = 20 TO 100 STEP 20
30 CIRCLE (128, 96), R
40 NEXT R
50 GOTO 50
```

This program will draw five circles, all with the same centre, but with each with a different radius. It uses a FOR NEXT loop to change the value of the radius. Line 20 is still the same FOR NEXT loop we have been using, but with an extra word, STEP. The STEP tells the computer how you want it to count through the loop. In this case, it will count on in steps of 20, (20, 40, 60, 80, 100).

43
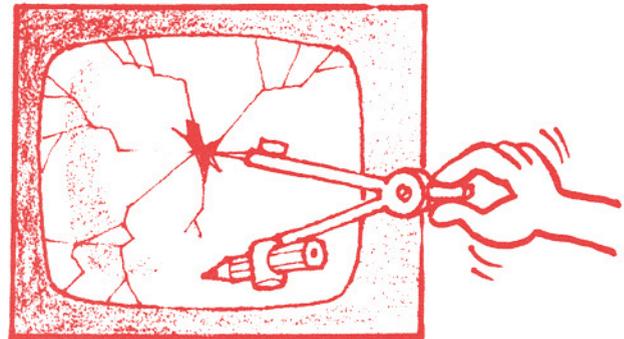
Now that we can draw pictures using LINE and CIRCLE, we may want to colour them in. To do that we use the PAINT command.

The PAINT command is just like colouring in a drawing. You start at one place and paint a colour until you reach a border, then you start in a different place with a different colour.
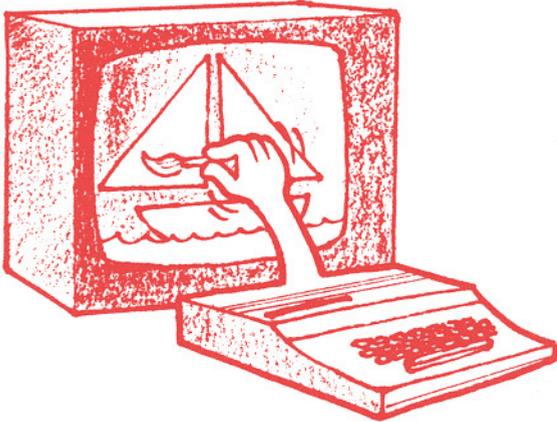
You choose the place to start painting by telling the computer the point (how many rectangles across and how many down). You also tell it the number of the colour to paint with, and the colour of the border it has to stop at.

In the high resolution mode we are using, all the borders are drawn in red, which is colour number 4. The only other colours we can use are green (1), yellow (2), and blue (3).

This program shows how the PAINT command works.

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 CIRCLE (128, 96), 80
30 LINE (100, 70) − (150, 120), PSET,
   B
40 PAINT (128, 96), 2, 4
50 PAINT (90, 65), 4, 4
60 PAINT (0, 0), 3, 4
70 GOTO 70
```

Lines 20 and 30 draw a circle with a square in the middle. Line 40 will paint starting at the middle of the screen, (128, 96), in the colour yellow, (2), until it reaches the red, (4), border. This line paints the square yellow. Line 50 will paint in red, (4), until it reaches a red border. This line paints the rest of the circle outside the square. Line 60 starts in the top left corner and paints the rest of the screen blue, (3).

We can now colour in the yacht that we drew earlier. If you load the program back into the computer, (CLOAD), and add these extra lines they will paint the yacht.

```
130 PAINT (25, 175), 3, 4
140 PAINT (175, 30), 2, 4
150 PAINT (200, 65), 4, 4
```

If you don't like these colours why not change them to a red yacht with blue sails?

We have now come to the end of this book. The commands we have used are enough for you to write your own programs, but there are many more commands available on your Dragon 32, and these can be found in the manual. Being a good programmer is like everything else – if you want to be good you have to practise.

Don't just try typing lines into the computer and hope that it will work. First think carefully about what it is you want to do, then draw your flowchart, or write the steps down. When you are sure that you have the right steps in the right order, then write out the BASIC program on paper. The last part is to copy the program into the machine and watch it run.

Apart from being useful, computing can be fun. We hope it will be fun for you.

PRACTICE MAKES PERFECT

45
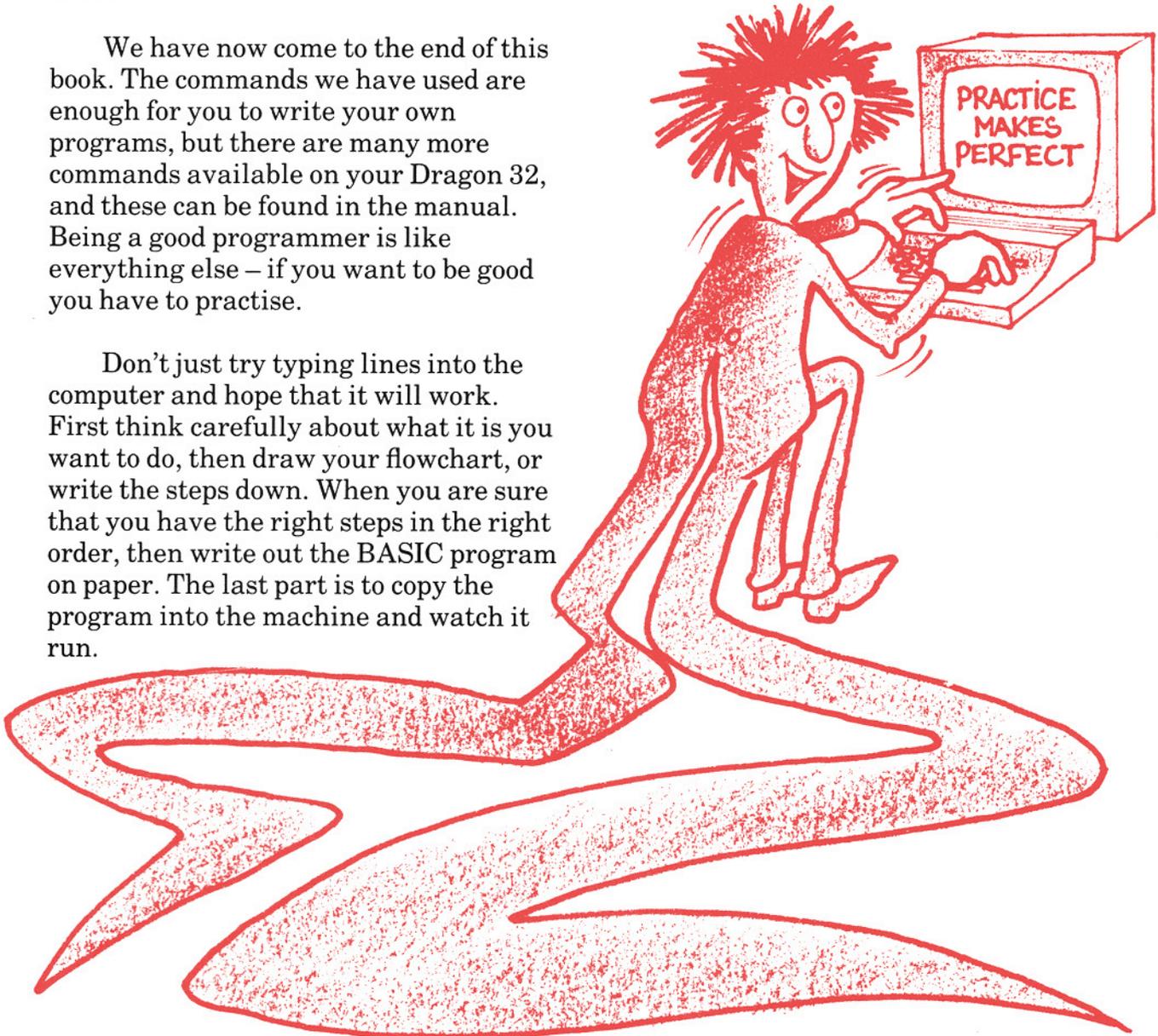
Things to remember from this chapter.

1. Your computer can draw in two different ways: low resolution and high resolution.

2. The low resolution mode is like drawing with a very thick crayon.

3. The high resolution mode is like drawing with a sharp pencil.

4. In the high resolution mode the screen is divided into 256 rectangles across and 192 down.

5. PSET and PRESET are the same as SET and RESET, but they can only be used in the high resolution mode.

6. The LINE command is used to draw lines. You must tell it where to start the line and where to end the line. Here is how the command should look.
LINE (10, 15) – (36, 40), PSET

7. The LINE command can be used to draw boxes or rectangles, by adding, B on the end.
LINE (10, 15) – (36, 40), PSET, B

8. If you add, BF to the LINE command it will fill the box with the drawing colour, red.
LINE (10, 15) – (36, 40), PSET, BF

9. The CIRCLE command will draw circles. You must tell it where the middle of the circle is, and the radius of the circle.
CIRCLE (128, 96), 55

10. The PAINT command is used to fill in colour. You must tell it where to start, what colour to paint with, and the colour of the border where it must stop.
PAINT (35, 62), 2, 4

Glossary of statements and commands

CIRCLE tells the computer to draw a circle. It must be used in the high resolution mode. You must give the centre of the circle and the radius.
25 CIRCLE (120, 50), 35

CLOAD loads a program from the cassette tape into the computer's memory. The name of the program must be in quotation marks. CLOAD does not need a line number.
CLOAD "MYPROG"

CLS clears the TV screen. If CLS is followed by a number, it will change the background colour. The numbers you may use are:

0	Black	5	Buff
1	Green	6	Cyan
2	Yellow	7	Magenta
3	Blue	8	Orange
4	Red		

10 CLS 8

CSAVE saves the program in the memory onto cassette tape. The name of the program must be in quotation marks and not more than eight letters long. CSAVE does not need a line number.
CSAVE "MYPROG"

END tells the computer the program is finished.
250 END

I'M A GIANT CLOAD

WE'RE BABY MYPROG'S

FOR...
NEXT   is a type of loop which
makes the computer repeat
the lines in between FOR
and NEXT a set number of
times.
10 FOR A = 1 TO 10
20 PRINT A
30 NEXT A

GOTO   tells the computer to jump
straight away to a certain
line number.
156 GOTO 20

IF...THEN   tells the computer to make a
decision. If the answer is
yes, it goes to the line
number after the THEN. If
the answer is no, it
continues on the next line in
the program.
45 IF A = 7 THEN 210

INPUT   stops the program and waits
for you to type something in.
The number you type in is
stored in the variable
following INPUT.
60 INPUT X

LINE   draws lines on the screen in
high resolution mode. It can
also be used to draw boxes,
and fill them with colour.
60 LINE (10, 20) — (40, 100),
   PSET
70 LINE — (35, 65), PSET
80 LINE (5, 5) — (250, 190),
   PSET, B
90 LINE (20, 20) — (120, 90),
   PSET, BF

LIST   prints the program in
memory onto the screen. It
does not need a line number.
LIST

NEW   clears the program from the
computer's memory. It does
not need a line number.
NEW

49

PAINT
is used in the high resolution mode to fill in colour on the screen. You must tell it where to start, what colour to paint with and the colour of the border where it is to stop.
110 PAINT (25, 80), 2, 4

PCLS
is used to clear the screen in the high resolution mode.
5 PCLS

PMODE
is used to tell the computer to draw in the high resolution mode.
10 PMODE 3, 1

PRESET
switches off a point that has been turned on by PSET. It can only be used in the high resolution mode.
115 PRESET (45, 62)

PRINT
tells the computer to write something onto the screen. If you use a comma (,) to separate items, it will print them in two columns. If you use a semicolon (;) to separate items, they will be printed next to each other.
20 PRINT "HELLO"
30 PRINT A, B
45 PRINT "THE ANSWER IS"; C

PSET
switches on a point to a chosen colour. It can only be used in high resolution mode.
15 PSET (115, 40, 3)

RESET
switches off a point that has been turned on by SET. It can only be used in the low resolution mode.
80 RESET (32, 12)

RND
selects a random number from the range given by the number following in the brackets.
25 A = RND (20)

RUN
starts the program in the memory. It does not have a line number.
RUN

SCREEN
is used to set the high resolution screen ready for drawing.
10 SCREEN 1, 0

SET
switches on a point to a chosen colour. It can only be used in low resolution mode.
140 SET (25, 6, 8)

More example programs

Toy train

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 LINE (20, 100) − (120, 140), PSET,
   BF
30 LINE (130, 100) − (250, 140), PSET,
   BF
40 LINE (180, 100) − (250, 75), PSET, B
50 PAINT (185, 80), 3, 4
60 LINE (230, 75) − (245, 50), PSET, B
70 PAINT (242, 60), 2, 4
80 LINE (179, 100) − (179, 40), PSET
90 LINE − (130, 40), PSET
100 CIRCLE (40, 145), 10
110 CIRCLE (100, 145), 10
120 CIRCLE (150, 145), 10
130 CIRCLE (190, 140), 20
140 CIRCLE (225, 140), 20
150 GOTO 150
```

Random painting

```
10 PMODE 3, 1: PCLS: SCREEN 1, 0
20 FOR I = 1 TO 4
30 LINE − (RND(255), RND(191)), PSET
40 CIRCLE (RND(255), RND(191)),
   RND(80)
50 NEXT I
60 FOR I = 1 TO 10
70 PAINT (RND(255), RND(191)),
   RND(4), 4
80 NEXT I
90 GOTO 90
```
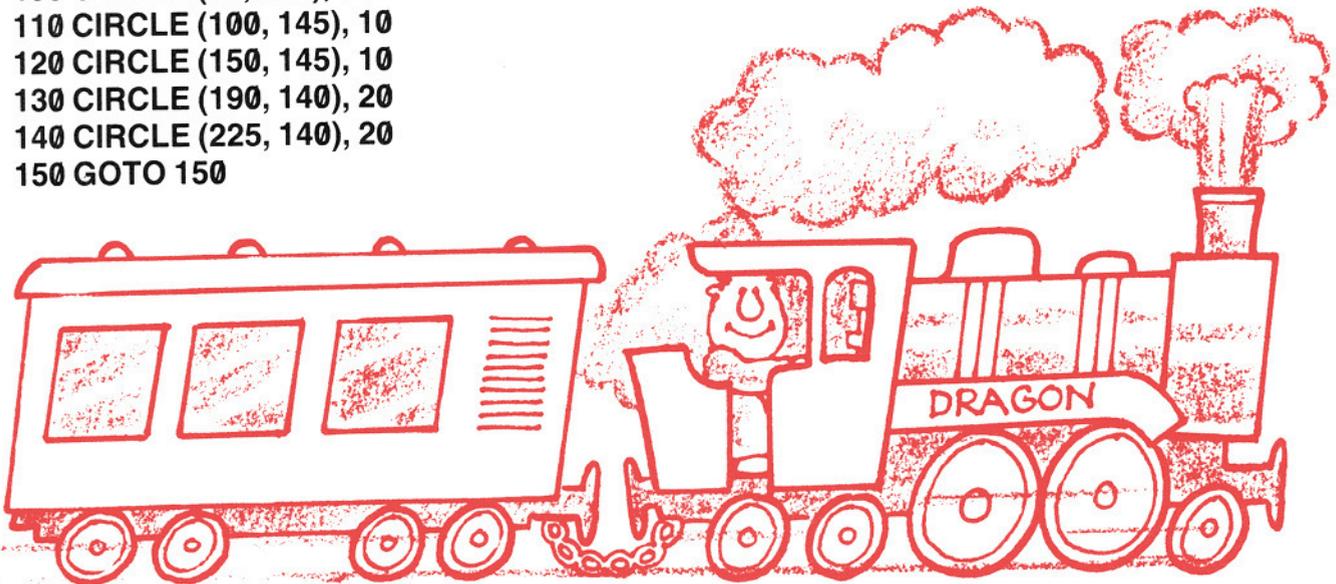
Funny face

```
 10 PMODE 3, 1: SCREEN 1, 0: PCLS
 20 CIRCLE (128, 96), 80
 30 PAINT (128, 96), 2, 4
 40 LINE (75, 50) − (90, 65), PSET, B
 50 PAINT (80, 60), 3, 4
 60 LINE (170, 50) − (185, 65), PSET, B
 70 PAINT (180, 60), 3, 4
 80 LINE (116, 70) − (140, 110), PSET,
    BF
 90 LINE (80, 130) − (180, 135), PSET,
    BF
100 GOTO 100
```

Kaleidoscope

```
10 CLS 0
20 X = RND(31)
30 Y = RND(15)
40 C = RND(8)
50 SET (31 + X, 15 + Y, C)
60 SET (31 + X, 15 − Y, C)
70 SET (31 − X, 15 + Y, C)
80 SET (31 − X, 15 − Y, C)
90 GOTO 20
```

Abstract

```
10 PMODE 3, 1: PCLS: SCREEN 1, 1
20 FOR I = 1 TO 50
30 LINE −(RND (255), RND (191)),
   PSET
40 COLOR RND(4)
50 NEXT I
60 GOTO 60
```

Swiss flag

```
 10 CLS 4
 20 FOR Y = 13 TO 17
 30 FOR X = 10 TO 52
 40 SET (X, Y, 5)
 50 NEXT X
 60 NEXT Y
 70 FOR X = 28 TO 34
 80 FOR Y = 5 TO 26
 90 SET (X, Y, 5)
100 NEXT Y
110 NEXT X
120 GOTO 120
```
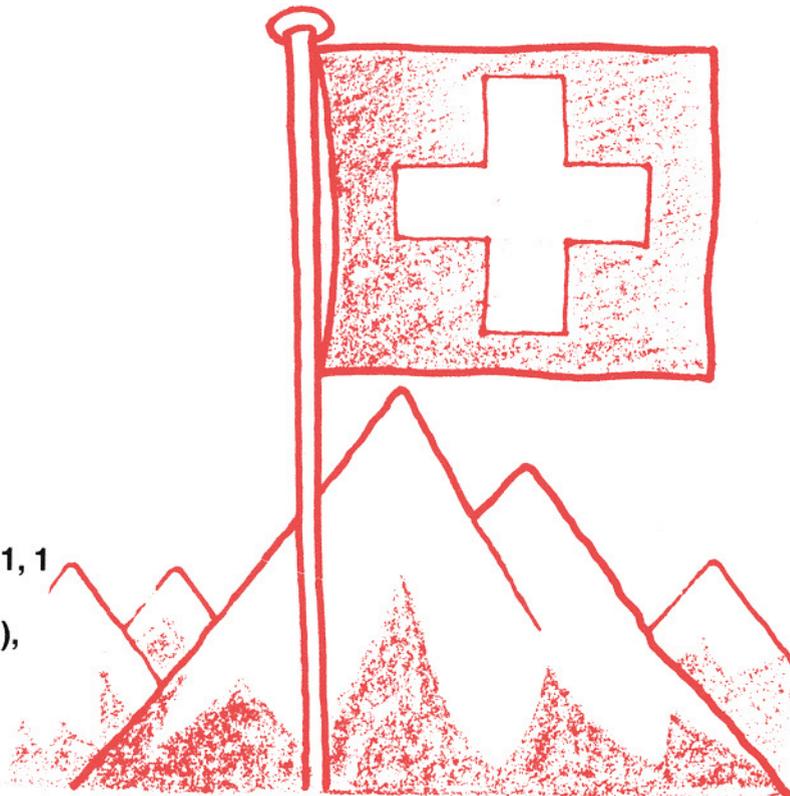
Number game

```
10 CLS
20 PRINT "FIND THE NUMBER GAME"
30 PRINT "FIND A 3 FIGURE NUMBER"
40 PRINT "ONLY USE FIGURES FROM
   1 TO 5"
50 PRINT "AFTER EACH TRY YOUR
   SCORE WILL BE SHOWN AS
   FOLLOWS"
60 PRINT "* IF FIGURE IN CORRECT
   POSITION"
70 PRINT "$ IF FIGURE IN WRONG
   POSITION"
80 PRINT "! IF FIGURE NOT IN
   CORRECT ANSWER"
90 PRINT "EXAMPLE IF ANSWER IS
   531"
100 PRINT "NUMBER ENTERED WAS
    134"
110 PRINT "SCORE  $*!"
120 X = RND (5) : Y = RND (5) : Z = RND
    (5) : Q = 1
130 PRINT "PLEASE ENTER 3 FIGURES
    SEPARATED BY COMMAS"
140 INPUT A, B, C
150 IF A < 1 OR B < 1 OR C < 1 THEN
    130
160 IF A > 5 OR B > 5 OR C > 5 THEN
    130 ELSE A$ = "!"
170 IF A = Y OR A = Z THEN A$ = "$"
180 IF A= X THEN A$ = "*"
190 IF B = Y THEN A$ = A$ + "*" :
    GOTO 220
200 IF B = X OR B = Z THEN A$ = A$ +
    "$" : GOTO 220
210 A$ = A$ + "!"
220 IF C = Z THEN A$ = A$ + "*" :
    GOTO 250
230 IF C = X OR C = Y THEN A$ = A$ +
    "$" : GOTO 250
240 A$ = A$ + "!"
250 PRINT "SCORE = " ; A$ : Q = Q + 1
260 IF A$ <>"***"THEN 130
270 PRINT "WELL DONE YOU TOOK"
    ; Q ; " TRYS "
280 PRINT "DO YOU WISH TO TRY
    AGAIN Y/N?"
290 INPUT M$
300 IF M$ = "Y" THEN 130 ELSE STOP
```

Rocket take-off

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 FOR Z = 1 TO 3
30 DRAW "BM125, 96; U26 R13 D26
   L13"
40 PAINT (128, 94), 2, 4
50 DRAW "L6 U6 E6 BR13 F6 D6 L6
   BU26 H6 G6"
60 FOR A = 110 TO 191 STEP 5
70 CIRCLE (119 + Z, A), 3*Z + 2
80 NEXT A
90 FOR B = 110 TO 191 STEP 5
100 CIRCLE (12*Z + 131, B), 3*Z + 2
110 PCLS
120 IF Z > 2 THEN 150
130 IF Z = 1 THEN M$ = "S8" ELSE M$
    = "S12"
140 DRAW M$
150 NEXT Z
160 DRAW "S4"
170 GOTO 20
```

I am your dragon

```
10 PMODE 3,1: SCREEN 1,0: PCLS
20 DRAW "BM10, 10; R30 F10 E10 R30"
30 DRAW "F40 G40 R120 F40 G40 L40"
40 DRAW "E40 L160 U40 E40 L80 U40"
50 DRAW "BM70, 135; R40 G40 L40
   E40"
60 DRAW "BM120, 80; R40 E30 R40
   E40 L60 G70"
70 DRAW "BM40, 5; E8 F8"
80 FOR A = 1 TO 4
90 PAINT (12, 12), A, 4
100 PAINT (80, 140), A, 4
110 PAINT (140, 70), A, 4
120 NEXT A
130 PCLS 1
140 FOR C = 1 TO 300
150 NEXT C
160 GOTO 20
```

School trip

```
10 CLS
20 PRINT "HOW FAR IS IT TO YOUR
   SCHOOL? (IN MILES)"
30 INPUT A:A = A*5*33
40 PRINT "HOW MANY MILES DO YOU
   TRAVEL TO SCHOOL IN A YEAR?
   HAVE A GUESS."
50 INPUT B
60 PRINT "YOU TRAVEL AROUND" ;A;
   "MILES A YEAR GOING TO,
   SCHOOL AND BACK."
70 C = A*.08
80 PRINT "IF YOU GO BY CAR IT
   COSTS YOUR DAD ABOUT" ;C;
   "POUNDS A YEAR. START
   WALKING!"
90 END
```

Backward land

```
10 CLS
20 PRINT "PLEASE TELL ME YOUR
   NAME"
30 INPUT N$
40 FOR I = LEN (N$) TO 1 STEP −1
50 A$ = A$ + MID$ (N$, I, 1)
60 NEXT I
70 PRINT "HELLO  "; A$
80 PRINT "WELCOME TO BACKWARD
   LAND"
```

Patchwork

```
10 CLS0
20 FOR I = 0 TO 63
30 FOR J = 0 TO 31
40 SET (I, J, RND(8))
50 NEXT J
60 NEXT I
70 GOTO 70
```

Secret messages

```
10 CLS
20 PRINT "THIS PROGRAM WILL
   CODE AND"
30 PRINT "DECODE YOUR
   MESSAGES"
40 PRINT "TO CODE, THE SECRET
   NUMBER"
50 PRINT "MUST BE BETWEEN 1 AND
   5"
60 PRINT "TO DECODE, USE THE
   SAME NUMBER"
70 PRINT "BUT ENTER A MINUS SIGN
   FIRST (I.E. −4)"
80 PRINT "ENTER CODE NUMBER"
90 INPUT C
95 IF C < −5 OR C > 5 THEN 10
100 A$ = "ABCDEFGHIJKLMNOPQRSTU
    VWXYZ  ABCDE"
110 PRINT "TYPE IN LINE TO BE
    CODED OR DECODED"
120 LINE INPUT L$
130 FOR I = 1 TO LEN (L$)
140 B$ = MID$ (L$, I, 1)
150 N = INSTR (A$,B$) + C
160 IF C < 0 AND N < = 0 THEN N = N +
    27
170 D$ = MID$ (A$, N, 1)
180 C$ = C$ + D$
190 NEXT I
200 PRINT "THE CODED LINE
IS : − "
210 PRINT C$
```

Perambulating

```
10 PMODE 3, 1: SCREEN 1, 0: PCLS
20 CIRCLE (180, 156), 28, 3: PAINT
   (189, 156), 3, 3
30 CIRCLE (110, 156), 28, 3: PAINT
   (110, 156), 3, 3
40 CIRCLE (144, 80), 68, 4, 1, 0, .5
50 LINE (212, 80) − (76, 80), PSET: LINE
   − (48, 32), PSET
60 PAINT (144, 82): CIRCLE (144, 80),
   70, 4, .8, .79, 1
70 LINE (160, 80) − (160, 28), PSET:
   PAINT (210, 75)
80 GOTO 80
```

Now add a loop to the program and see what happens. Insert the new and revised lines below.

```
15 FOR C = 0 TO 4
20 CIRCLE (180, 156), 28, 3: PAINT
   (189, 156), C, 3
30 CIRCLE (110, 156), 28, 3: PAINT
   (110, 156), C, 3
75 FOR A = 1 TO 600
76 NEXT A
80 PCLS 1
90 NEXT C
100 FOR D = 1 TO 600
110 NEXT D
120 GOTO 10
```

Drawing board

```
10 CLS: PRINT "DRAWING BOARD"
20 PRINT: PRINT "USE THE ARROW
   KEYS TO MOVE THE PEN"
30 PRINT "PRESS THE LETTER C TO
   CHANGE THE COLOUR"
40 PRINT: PRINT "PRESS THE SPACE
   BAR TO START"
50 A$ = INKEY$: IF A$ = ""THEN 50
60 CLS0: C = 1: A = 31: B = 15
70 A$ = INKEY$: X = 0: Y = 0
80 IF A$ = " ↑ " THEN Y = −1: GOTO
   140
90 IF A$ = CHR$ (10) THEN Y = 1:
   GOTO 140
100 IF A$ = CHR$ (9) THEN X = 1: GOTO
    140
110 IF A$ = CHR$ (8) THEN X = −1:
    GOTO 140
120 IF A$ = "C" THEN C = C + 1: GOTO
    140
130 GOTO 70
140 A = A + X: B = B + Y: IF C > 9 THEN
    C = 1
150 IF A < 0 THEN A = 0
160 IF A > 63 THEN A = 63
170 IF B < 0 THEN B = 0
180 IF B > 31 THEN B = 31
190 SET (A, B, C): GOTO 70
```

Fine drawing board

```
10 CLS: PRINT "FINE DRAWING
   BOARD"
20 PRINT: PRINT "USE THE ARROW
   KEYS TO MOVE THE PEN"
30 PRINT "PRESS THE LETTER C TO
   CHANGE THE COLOUR"
40 PRINT: PRINT "PRESS THE SPACE
   BAR TO START"
50 A$ = INKEY$: IF A$ = ""THEN 50
60 C = 1: A = 128: B = 96
65 PMODE 3, 1: PCLS: SCREEN 1, 1
70 A$ = INKEY$: X = 0: Y = 0
80 IF A$ = " ↑ " THEN Y = −1: GOTO
   140
90 IF A$ = CHR$ (10) THEN Y = 1:
   GOTO 140
100 IF A$ = CHR$ (9) THEN X = 1: GOTO
    140
110 IF A$ = CHR$ (8) THEN X = −1:
    GOTO 140
120 IF A$ = "C" THEN C = C + 1: GOTO
    140
130 GOTO 70
140 A = A + X: B = B + Y: IF C > 8 THEN
    C = 1
150 IF A < 0 THEN A = 0
160 IF A > 255 THEN A = 255
170 IF B < 0 THEN B = 0
180 IF B > 191 THEN B = 191
190 PSET (A,B,C): GOTO 70
```
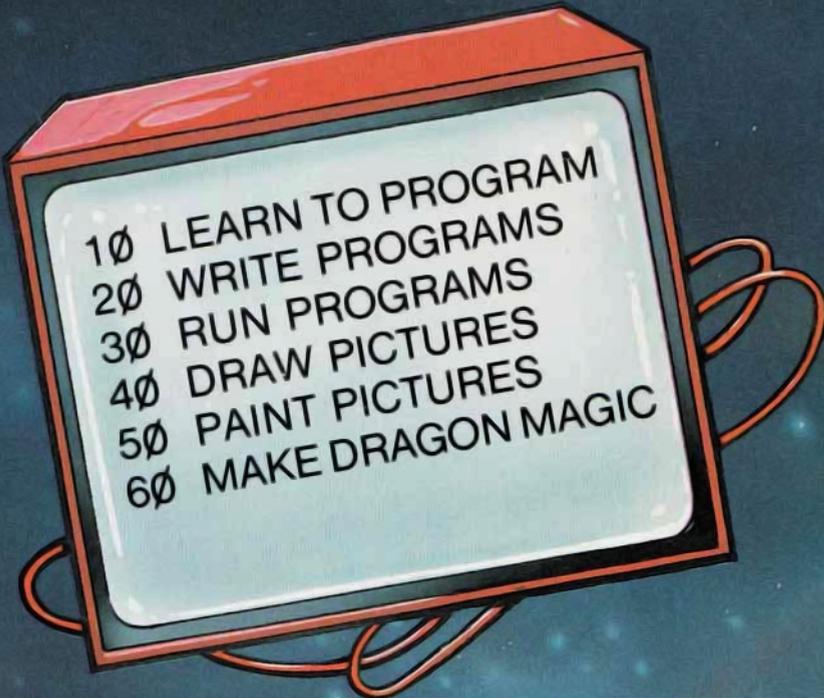
1Ø LEARN TO PROGRAM
2Ø WRITE PROGRAMS
3Ø RUN PROGRAMS
4Ø DRAW PICTURES
5Ø PAINT PICTURES
6Ø MAKE DRAGON MAGIC

DRAGON MAGIC Your First Programming Book

FOULSHAM

DRAGON
magic

Your First Programming Book
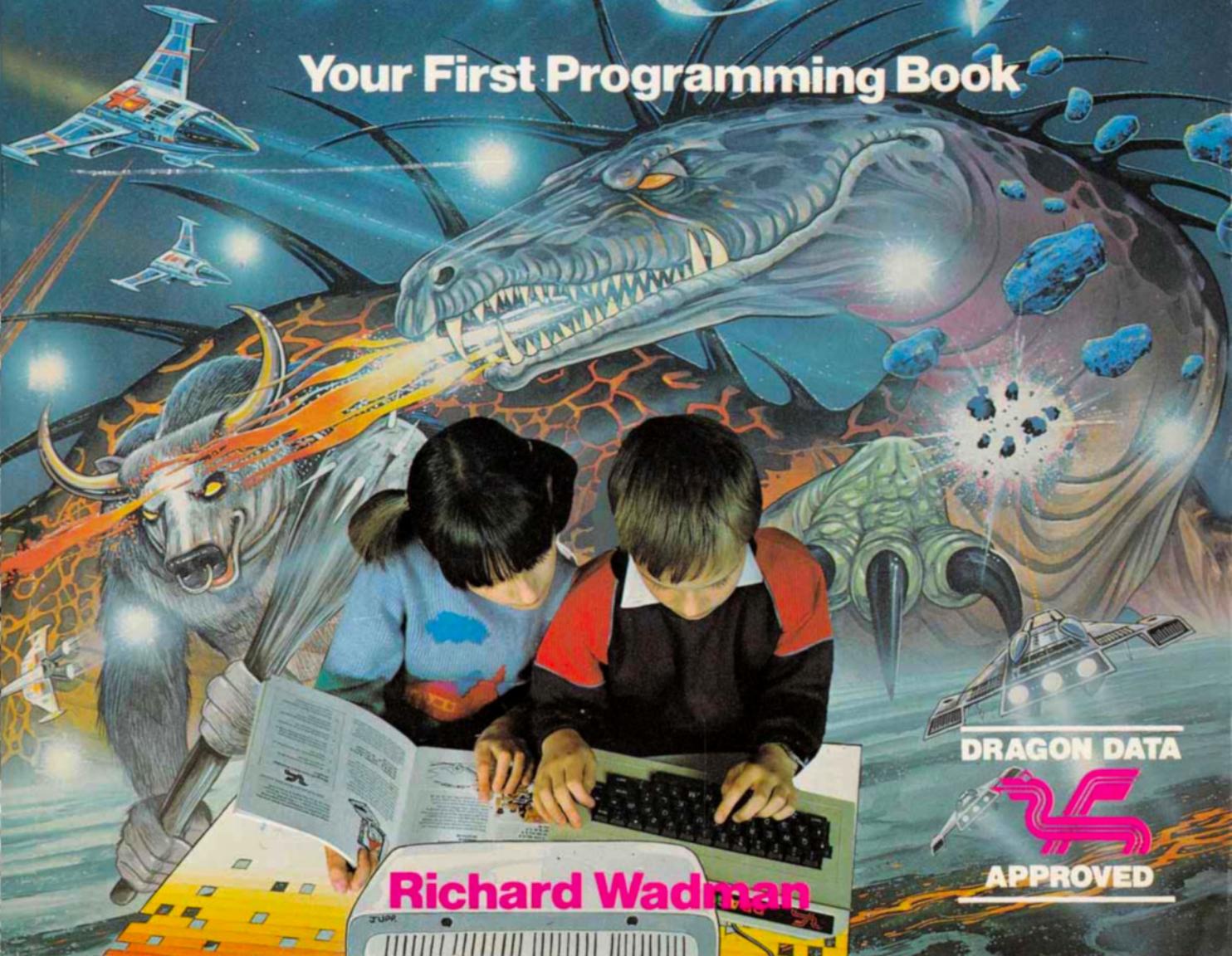
Richard Wadman

DRAGON DATA

APPROVED