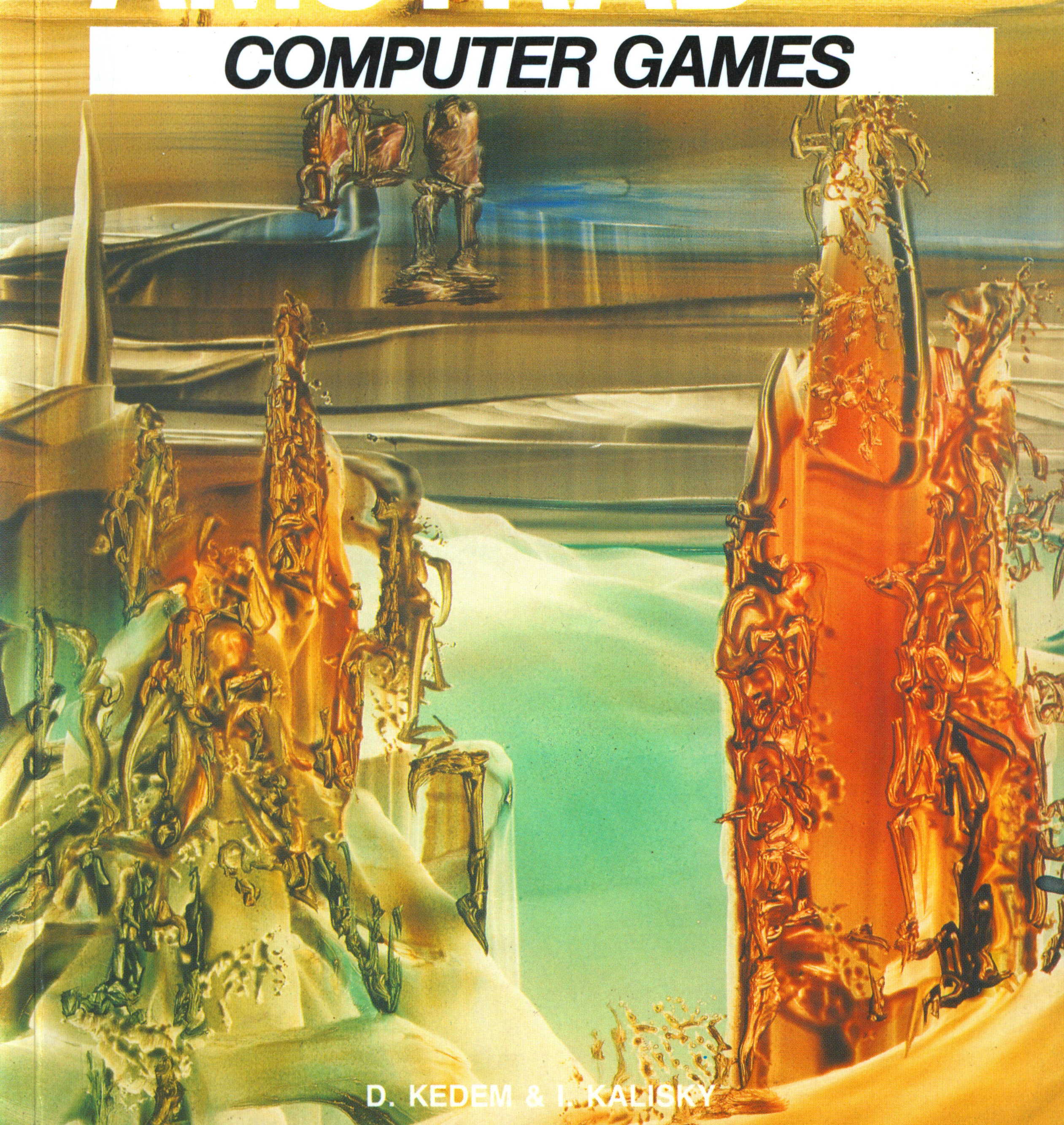
3 Watson's Notes

AMSTRAD 464,664 & 6128

COMPUTER GAMES



D. KEDEM & I. KALISKY

Watson's Notes

AMSTRAD 464, 664 & 6128

I. KALISKY & D. KEDEM

UNIT 3

COMPUTER GAMES

GLENTOP
PUBLISHERS ☐ LIMITED

DECEMBER 1985

ISBN 1 85181 005 6

Published by:

Glentop Publishers Ltd
Standfast House
Bath Place
High Street
Barnet
Herts EN5 1ED
Tel: Ø1-441-413Ø

Cover illustration from an original painting by Nick Woods.

Contents

Foreword

In this unit, we will concentrate on developing computer video games and in doing so, learn more about Amstrad BASIC.

The techniques taught during this unit are the basic background to writing all computer video games, however advanced they may be.

By the end of the unit, you will have improved you knowledge of the BASIC language considerably.

Itzhak Kalisky and Dani Kedem.

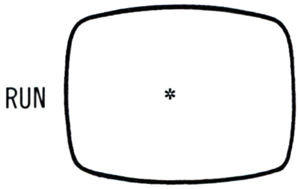
Chapter 1

VIDEO PENCIL

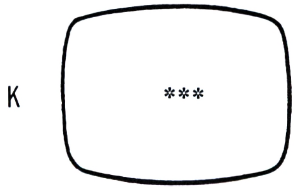
In the first chapter of this unit, we will develop a 'video pencil' program which is rather like a computer 'etcha sketch' and should work in the following way:

Video Pencil

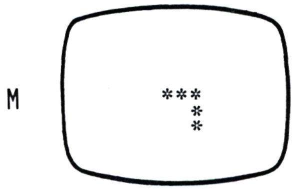
- When you run the program, an asterisk is displayed in the centre of the screen.



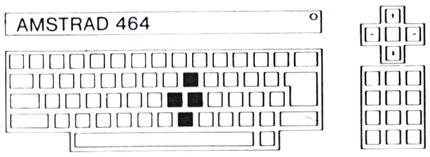
- If you press the letter 'K', the asterisk moves to the right, leaving the previous asterisks on the screen.



- If you press the letter 'M', the asterisk moves down the screen until you take your finger off the key.



- If you press 'J', the asterisk moves to the left and if you press 'I', the asterisk moves up the screen.



(The diagram above is of an Amstrad 464 computer which we have chosen to use to show the various key positions. This shouldn't cause any problems for those of you who have 664 or 6128 models.)

Note: we have chosen I, J, K and M as the direction keys, however, we could have chosen any four keys.

Before you can write the video pencil program, you must know the following command.

IF...THEN

? Do you know what the following program does?

```
10 INPUT A
20 IF A=5 THEN PRINT "BOOM"
30 GOTO 10
```

When you think you know what the program does, type it in and run it.

Input the numbers from 1 to 10, remembering to press ENTER after each one.

? What happens?

Explanation

When the computer executes line 20, it checks to see if A equals 5.

- If it doesn't, the computer ignores line 20 and carries on to line 30.
- If the condition exists, ie. you entered 5, the computer prints 'BOOM'.

Before we continue with the video pencil program, lets just look at the IF...THEN command a bit more.

Task

Write a program to perform the following:

- The computer waits for the input of a number from 1 to 1Ø and on entering a number, eg. 1, the computer prints 'ONE' on the screen.
- When you enter the number 2, the computer prints 'TWO'.

Answer 1

Task

Write a program to do the following:

- On pressing the first letter of your name, the computer displays your name in full.
- If you press the letter 'L', the computer displays the whole program.
- If you press the letter 'N', the program is erased from memory (NEW).

Answer 2

Games Corner

☐ Write a program to create the following game which you and a friend can play.

- Tell your friend to leave the room for a moment whilst you run the program. At this point, you enter a 'secret' number (from 1 to 99) into the computer which the program then erases from the screen so your friend can't see it.
- Your friend then returns to the room and tries to guess the secret number.

- If he/she guesses 'higher' than the secret number, the computer prints 'Too high'.
- If the guess is 'lower' than the secret number, the computer prints 'Too low'.
- If the guess is correct, the computer informs your friend appropriately.
- When your friend has guessed the number, it is your turn to go out of the room.

The aim of the game is to guess the secret number in as few guesses as possible.

Hint:

```
IF A>B
```

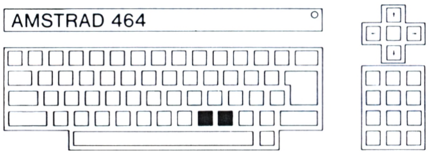
Greater Than

means if A is GREATER THAN B and

```
IF A<B
```

Less Than

means if A is LESS THAN B.



Back to the Video Pencil

Now you know the IF...THEN command, we can go back to the Video Pencil program .

If your not sure how the IF...THEN command can be utilized in the video pencil program, look at the following lines:

```
IF A$="K" THEN X=X+1

LOCATE X,Y:PRINT "*"
```

Answer 3

Although the program works, it is slow because you have to press ENTER after each direction key.

Improving the Video Pencil

To make the program work according to the description on page 7, i.e. without pressing ENTER, we have to learn a new command.

INKEY$

Instead of having INPUT A$ in the program, put:

```
    A$=INKEY$
```

Note: if you used a different variable name eg. INPUT C$, the line will become C$=INKEY$.

INKEY$

☐ Now run the program and move the asterisk around the screen. You can see that you don't have to press ENTER anymore.

Explanation

It's obvious that this change came about from the inclusion of the INKEY$ command, but, what does INKEY$ do?

When the computer executes an INKEY$ command, nothing happens until you type in a character. The character input is then stored in the variable specified.

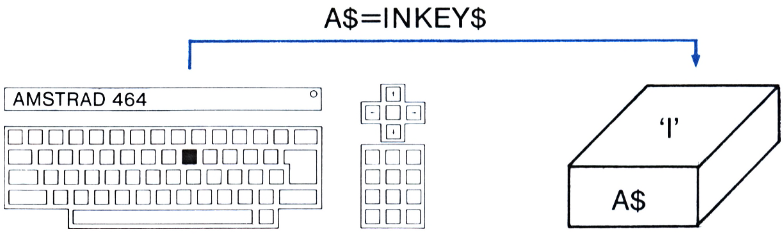
Note: INKEY$ only accepts one character whereas INPUT accepts upto 255 characters (String INPUT).

Therefore, when the computer executes:

```
    A$=INKEY$
```

it puts the character input into the variable A$, without having to press ENTER.

For example:



In this case, A$ is equal to ______(I, J, K, M)? The computer then displays an asterisk above the previous one.

When the computer executes A$=INKEY$ again, it puts the current character pressed into A$ and continues as before.

Restricting the Pencil

You've probably noticed that when the 'pencil' goes over the edge of the screen, the computer reacts in a strange way.

It would be a better program if you stopped the user doing this.

Therefore, add lines to the program so that the pencil cannot go over the edges of the screen. **Answer 4**

What do you think would happen if you changed line 80 to PRINT A$ instead of PRINT "*"?

Now press CAPS LOCK and run the program. Why doesn't the pencil move? **Answer 5**

More on INKEY$

What would happen if you didn't input a character when the computer executed an INKEY$ command?

The computer would simply put a Null String into the variable A$. (If you don't remember what a Null String is, look at Unit 2, Chapter 4.)

In this case, A$ would receive a value of ______ ("", "I", "J", "K", "M")?

The important thing to remember is that INKEY$ will always receive a value whether it be a character or a null string. Therefore, the program will not wait for input, like it does with the INPUT command.

How can we prove this? **Answer 6**

Another way of proving that the computer continues to run is by producing a short sound each time the computer executes the INKEY$ command. (If you don't remember how to produce sound, look at Unit 1, Chapter 7.) **Answer 7**

You will hear a continuous sound, which means that the INKEY$ is executed repeatedly. If it was only executed once, the sound would have stopped.

Before going on, delete line 35.

Comparing INPUT and INKEY$

The INPUT and INKEY$ commands are both ways of defining variables, however they do differ:

- With INKEY$, you do not have to press ENTER.

- INKEY$ only accepts one character whereas INPUT accepts upto 255 characters.

- INKEY$ does not wait for a character to be input. ie. it accepts null characters.

Changing the Pencil

☐ Add lines to the program so that when you press a certain key (eg. "↑"), the computer goes to a part of the program which allows you to change the pencil character, which at present is a '*'.

When you have changed the pencil, the computer continues drawing from the last point displayed with the previous pencil.

Hint: what do the following lines do:

```
150 B$=INKEY$
160 IF B$="" THEN GOTO 150
```

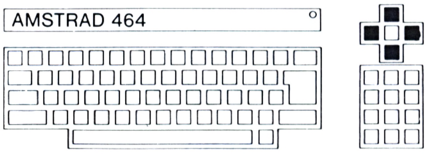
☐ Add these lines to the program and run them (RUN 150).

☐ Now try to complete the program on your own.

Answer 8

Task

Change the video pencil program so that instead of using the existing keys (I, J, K, and M), the four cursor keys are used.



Hint

In Unit 2, Chapter 5, we saw how each character had a character number (CHR$).

? What are the character numbers of the four cursor keys (←,→,↑,↓)? **Answer 9**

Now, all you have to do to solve this task is to substitute the previous keys with these new character values. **Answer 1Ø**

Coloured Pencil

☐ Modify the program so that you can change the colour of the pencil at any time. **Answer 11**

Task

Using the video pencil, write your name on the screen.

```
PPPP  AAAA  U  U  L
P  P  A  A  U  U  L
PPPP  AAAA  U  U  L
P     A  A  U  U  L
P     A  A  UUUU  LLLL
```

At this stage, the task is difficult because we need to move the pencil without printing anything. eg. from the 'P' to the 'A'. This can be done by either using a 'space pencil' (B$=" ") or by setting the colour of the pencil to the same colour as the screen (C=4), however, you cannot be sure of the pencil's position.

☐ Therefore, change the program so that when you press the '–' key, the pencil moves as before but it doesn't leave the characters on the screen.

```
PPPP □□□ .........AAAA
```

Here you press "–" (□ means space)

When you want to continue drawing, press the '=' key and the pencil should return to its original state. **Answer 12**

Joysticks

Joysticks

For those of you who have joysticks, we will show you how to use them in the video pencil program instead of having to use the cursor keys.

Firstly, you must connect your joystick to the Amstrad making sure that it is not on 'Autofire'. If it is, all sorts of things will happen to the computer.

JOY(Ø)

Add the following lines to the program in memory and run them:

```
500 CLS
510 LOCATE 20,10:PRINT JOY(0)
520 GO TO 510
```

JOY(Ø)

You can see that the number Ø appears in the middle of the screen.

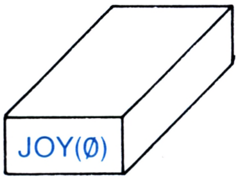
What happens if you move the joystick forward?

What happens when you centre the joystick again?

Move the joystick in every possible direction and see how the numbers change according to its position.

Explanation

In line 51Ø, you can see that we PRINT JOY(Ø). JOY(Ø) is a function that returns the status of the joystick. ie. it displays a different value for each joystick position.

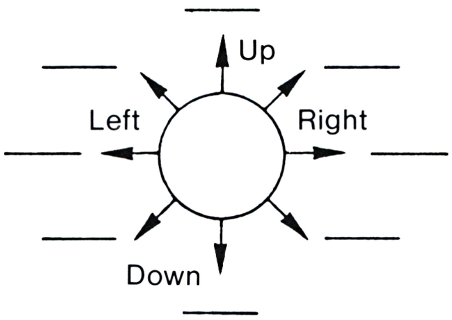


When the joystick is centred, JOY(Ø) returns the value of ______(1, Ø)?

When the joystick is forward, JOY(Ø) returns the value of ______(1, Ø)?

Fill in the values that JOY(Ø) returns when the joystick is in each of the following positions:

Answer 13



Note

When the computer prints JOY(Ø), the current position of the joystick is displayed and the program continues to run.

What value does JOY(Ø) return when you press the fire button on the joystick?

And what about when you move the joystick and press fire?

Answer 14

Now you know about joysticks, try to incorporate the joystick into the video pencil program.

Answer 15

Improving the Program

At present, the video pencil can move in four directions. Change the program so that it can move in every direction that the joystick can move in. ie. diagonally.

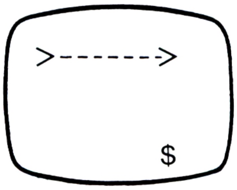
Chapter 2

SHOOTING THE ALIENS

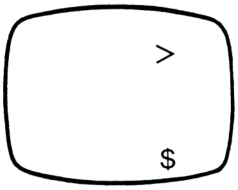
We think that at some point in time, most of you must have played a video game where you have had to shoot invaders or runaway from monsters.

In this chapter, we will construct a video game called 'Shooting the Aliens' which will work as follows:

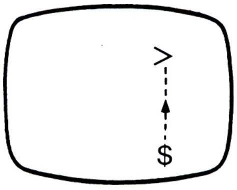
- The 'alien spaceship' (>) will continuously move across the screen (from left to right). Your spaceship ($) waits below.



- When the alien spaceship is directly above your spaceship, press 'fire'. (Fire can be any key you wish but it should be easy to reach. ie. space bar.)



- At this point, the alien spaceship stops and the missile (↑) is launched.



The aim of the game is to hit the alien spaceship.

Let's see if you know all you need to know to write this program:

- Do you know how to move the alien spaceship across the screen? (Yes or No).
- Do you know how to display your spaceship at the bottom of the screen? (Yes or No).
- Do you know how to detect when the fire button has been pressed? (Yes or No).
- Do you know how to move the missile? (Yes or No).

The answer to all these questions should be 'yes' meaning you can start writing the program.

Stage 1

The first stage is to move the alien spaceship across the screen continuously and place your spaceship at the bottom of the screen.

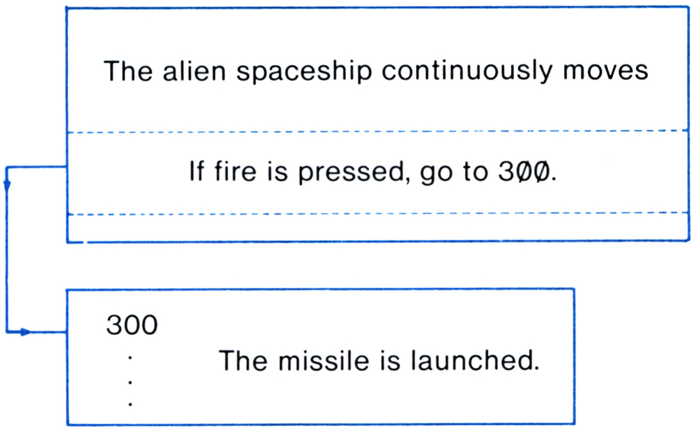
Answer 16

Stage 2

You must now choose a key to act as the fire button. We will assume that you chose the space bar as the fire button.

The alien spaceship continuously moves across the screen until you press the spacebar (fire). When you press fire, the computer launches its missile.

Below is a diagram which illustrates how the program should be constructed:


The alien spaceship continuously moves
If fire is pressed, go to 3ØØ.
300
The missile is launched.

Note: use the INKEY$ command to implement the fire button. ie. you don't have to press ENTER after pressing the spacebar. **Answer 17**

Stage 3

Once you have pressed fire, the missile needs to be launched. What we mean by 'launch' is, move the missile up the screen.

(Use a negative STEP value.) **Answer 18**

You have now finished the basic structure of the game, however, improvements can always be made.

Space Sounds

A video game is not a video game without sound. Add sound to the program to accompany the alien spaceship's movement and the firing of the missile.

(If you don't remember how to use the SOUND command, look at Unit 1, Chapter 7 and Unit 2, Chapter 1.)

Task

Change Stage 1 of the program so that as the alien spaceship moves across the screen, a sound is produced whose pitch gets lower and lower.

Hint

The pitch should start at about 5Ø and be incremented by the loop variable X.

Note: make sure you get rid of the delay first.

Answer 19

A Strange Reaction

? What happens to the alien spaceship and the sound produced when you first run the program?

? What happens to the sound immediately after pressing fire?

☐ Run the program and stop it when the alien spaceship is in motion.

☐ Now instruct the computer to:

```
SOUND 1,3ØØ,1ØØ
```

When you press ENTER, the computer continues to sound the notes within the program.

At this stage, we won't explain why the sound doesn't immediately stop when you press fire, but if you add the following line to the program, it will.

```
57 IF SQ(1)>127 THEN GOTO 57
```

☐ After you've added the line, run the program.

Task

Modify the program so that when you press fire, a sound whose pitch rises, is produced.

Answer 20

Sound Default

As you may remember, the third number in the SOUND command specifies the length of the note produced.

What happens if you leave out the third number? For example:

```
SOUND 1,X+5Ø
```

Take out the third number in line 55 and run the program.

Why does each note now take longer to play?

If you leave out the third number in the SOUND command, the note played will take the default value as the length. If something has a default value, it means that it has already been specified by the computer. In this case, the default value for the length of a sound is 2Ø.

Does the second number in the SOUND command have a default value?

Check this by excluding the second number from the command.

Summary

It is only the third number in the SOUND command that has a default value and this is equal to ______.

Improving the Program

- Add lines to the program so that the game continues even after you have fired the first missile.

Note: if you miss the alien spaceship, remember to erase it from the screen before continuing onto the next alien.

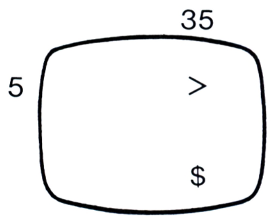
Answer 21

- Display the amount of missiles you have fired.

Hint: you need to include a section which counts. For example, the watch program in Unit 2, Chapter 3.

Answer 22

- Instruct the computer to display the amount of hits you have had.



If you don't know how to do this, answer the following questions:

? In which column (X) must the alien spaceship be in order for the missile to hit it?
______(5, 1Ø, 35)?

? Assuming you have chosen M as the count, which of the following lines will solve the problem:

```
LET M=M+1

IF X=35 THEN LET M=M+1

IF Y=35 THEN LET M=M+1
```

Answer 23

❑ Now insert the appropriate lines into the program.

Answer 24

- Stop the program running when the user has fired 1Ø missiles.

(Use the END instruction.)

Answer 25

Part LIST

There are times when you only want to see certain parts of a program rather than the whole program. The LIST command can be used to display certain specified lines. For example:

```
    LIST -1ØØ
```

will display all the lines upto line 1ØØ.

```
    LIST 1ØØ-2ØØ
```

will display all the lines between line 1ØØ and line 2ØØ.

```
    LIST 1ØØ
```

will display line 1ØØ.

- Before you start the game, the computer asks you at what speed you want the aliens to move at.

- If you have a joystick, use its fire button instead of the spacebar.

(If you don't know how to, look at page 16.)

To End the Game

Add lines to the program so that when you have mastered the game, ie. 10 out of 10 hits, the computer erases the program from memory (NEW).

Chapter 3

NUMBER GAMES

The Electronic Die

In this chapter, we will write a program that, on the press of a key, displays a number between 1 and 6 inclusively, like a die.

We will construct the die program stage by stage:

☐ Firstly, run the following program:

```
10 PRINT RND
20 GOTO 10
```

RND

? What does the RND command do?

It tells the computer to RaNDomly choose a number. ie. the number displayed is not related to any other number.

? The numbers displayed on the screen are greater than or equal to ______(Ø.1, Ø, Ø.5) and less than ______(1, Ø.9, 2).

Note: do you remember what the following number represents:

```
5.136E-03
```

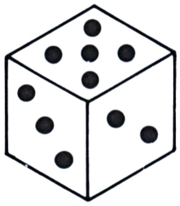
This is another way of writing the decimal fraction Ø.ØØ5136.

Run the program and check you answers.

Summary

From running the program, you can see that the RND command chooses numbers randomly between Ø and 1 (but never 1).

Change the program so that the computer displays numbers between Ø and 6, remembering that a die has 6 sides.



```
10 PRINT RND*______
20 GOTO 10
```

Answer 26

Does the program display numbers between Ø and 6?

It does, but a die never displays fractions.

Therefore, we must somehow get rid of the fractions. The following program will do this:

```
10 PRINT INT(RND*6)
20 GOTO 10
```

INT

INT

The INT command stands for INTeger and an integer is a whole number. ie. a number that has no fraction. It is used in the following way:

```
INT(4.6)=4
```

Basically, all the INT command does is 'round down' the number in brackets. For example, 4.6 must be made a whole number. Usually in maths, you take the whole number that it is closest to. In this case, it would be 5, however the INT command takes the 'lowest' whole number. Therefore, 4.6 becomes 4.

What would be the result of the following:

```
INT(7.99)
```

Think about it and then type in:

```
PRINT INT(7.99)
```

Now run the program again and answer the following questions:

The number Ø is displayed. Does this number appear on a real die? ______ (Yes or No)?

The number 6 is not displayed. Does this number appear on a real die? ______ (Yes or No)?

In other words, the computer is choosing numbers from Ø to 5 instead of 1 to 6.

Change the program so that numbers between 1 and 6 are displayed:

```
1Ø PRINT INT(RND*6)+ _____
2Ø GOTO 1Ø
```

Answer 27

Football Pools

Write a program that will fill in a football pool form, where:

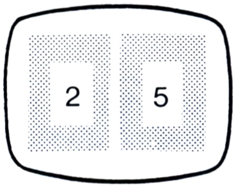
```
Ø=Draw
1=Team 1 wins
2=Team 2 wins
```

Answer 28

Using Two Dice

Write a program so that when you press a key, the computer displays the result of throwing two dice.

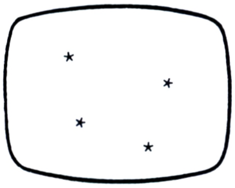
- Draw a coloured box around each number.



- Inform the user when there is a double. eg. play a tune or change the screens colour.

A Very Random Program

Write a program that displays stars (*) in random positions, in MODE Ø:



- The computer displays the stars in random colours and produces random sounds for each star displayed.

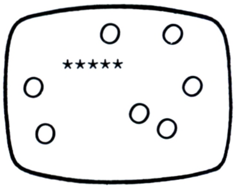
(The pitch of the sound produced should be between 5Ø and 25Ø.)

Answer 29

Games Corner

The computer displays a submarine on the screen which consists of 5 asterisks in a line (*****). You are then asked how many bombs you want to drop on it.

If, for example, you chose 4Ø bombs, the computer randomly displays 4Ø bombs (O) on the screen.



If the bombs miss the submarine, you get 4Ø points. If it is hit (*O***), you don't get any points.

You can play this game with a friend (each having alternate turns) or with the computer where the computer chooses for itself, the number of bombs it is prepared to endanger its submarine with. **Answer 3Ø**

Improving the Game

Let's assume that you want the computer to count both player's score. For the computer to do this, it must know when the submarine has been hit.

? How will the computer know this?

AND

For the moment, let's say that your submarine consists of only 1 asterisk which is located in row 8, column 1Ø. The computer displays the bomb at the random position specified by X and Y. Therefore, we can say, 'If Y=8 **AND** X=1Ø, the submarine has been hit'.

In computer language, it becomes:

```
    IF Y=8 AND X=1Ø THEN END
```

AND

Now put five IF...THEN commands into the original program to detect whether any part of the submarine has been hit. **Answer 31**

OR

It is possible to shorten this program by incorporating all the IF...THEN commands into one line.

Let's assume for a moment that your submarine consists of two asterisks, which are located in row 8, columns 1Ø and 11.

```
    IF (Y=8 AND X=1Ø) OR (Y=8 AND
    X=11) THEN END
```

In this case, if either one of the brackets is true, the program will end. ie, if Y=8 and X=1Ø **OR** 11, stop the program.

OR

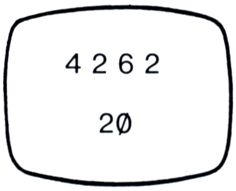
☐ Incorporate all five tests into one line of the submarine program, using the AND and OR commands.

☐ Now that the computer can detect when the submarine has been hit, tell it to keep both player's scores.

Number Games

☐ Write a program to do the following:

- The computer randomly chooses four numbers (eg, 4,2,6,2) and displays them on the screen. A fifth random number is then displayed below the four numbers (eg, 2Ø).



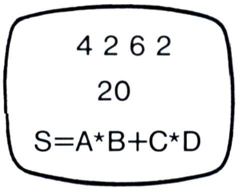
```
4 2 6 2

2Ø
```

You have to work out what mathematical operations need to be performed on the four numbers to produce the fifth number.

Note: you can use addition, subtraction, multiplication and division as many times as is required and you don't have to use all of the numbers.

- On pressing the spacebar, the computer displays a possible solution.



```
4 2 6 2
20
S=A*B+C*D
```

For example, in our case, the computer displays S=A*B+C*D where:

A=1st Random Number
B=2nd ....
C=3rd ....
D=4th ....
S=5th Random Number

When you substitute the variables with their values, the equation looks like this:

```
4*2+6*2=2Ø
```

Note: if you're not sure how we arrived at the answer 2Ø, look at the Appendix in Unit 1.

? How does the computer work out the equation? There are 2 possible methods:

The first method is by doing lots of AND and OR tests on the four random numbers but this is difficult and would take a long time to program.

The second method, which we have chosen to use, is easier, but it is cheating a bit. Instead of producing five random numbers, we produce four and work out the fifth from a random equation. The fifth number is then displayed on the screen.

The difference between the two is that in the first method, the equation is worked out by the computer (from the fifth number) whereas in the second method, the fifth number is worked out from the equation.

Now, before we write the program, lets see what you do or don't know. You know how to display four random numbers on the screen, however, you don't know how to randomly choose an equation. In order to do this, you need to learn a new command:

ON...GOTO

Enter the following program into the computer and run it:

```
10 CLS
20 INPUT R
30 ON R GOTO 40,50,60,70
40 PRINT "AAA":GOTO 20
50 PRINT "BBB":GOTO 20
60 PRINT "CCC":GOTO 20
70 PRINT "DDD":GOTO 20
```

ON...GOTO

Input the numbers from 1 to 4.

Explanation

```
30 ON R GOTO 40,50,60,70
```

This line tells the computer to go to a line in the program depending on the value of R. If R=1, the program will go to line 40, if R=2, the program will go to line 50 etc. In fact, this one line replaces the four following IF...THEN commands:

```
IF R=1 THEN GOTO 40
IF R=2 THEN GOTO 50
IF R=3 THEN GOTO 60
IF R=4 THEN GOTO 70
```

Task

Change the program in memory so that when you press the number '2', the computer displays 'CCC', and when you press the number '3', the computer displays 'BBB'.

Answer 32

? What happens if you press '5' or '9'?

The computer reacts in the same way as if you pressed the number '1'.

☐ Add the following line to the program:

```
35 PRINT "XXX":GOTO 2Ø
```

☐ Now run the program and see what happens when you press a number greater than 4.

The computer ignores line 3Ø and goes onto line 35.

Conclusion

If you input a value that does not have a corresponding line number, the computer will do one of two things:

- Firstly, it will look to see if there is a line after the ON...GOTO line and if there is, and providing this line is not specified in the ON...GOTO command, it will execute it.
- If the first check is not successful, the program will continue running from the first line number specified in the ON...GOTO command.

Task

Alter the program in memory so that the computer randomly chooses a number between 1 and 4 instead of the user inputing a number.

Answer 33

Back to the Number Game

You should now be able to write the Number Game that we described on page 32. If you have any difficulties writing this program, follow the stages below:

Note: remember that we changed the program so that the four random numbers produced are put into a random equation to produce the fifth number.

Stage 1

The first stage is to randomly choose a value (1-9) for the variables A, B, C and D and then to apply a random equation to the variables. Below are some equations that you can use, however, the more equations you have, the harder the game.

```
    110 S=A*B+C-D:PRINT S:GOTO 180
    120 S=A+B+C-D:PRINT S:GOTO 180
    130 S=A/B+C*D:PRINT S:GOTO 180
```

Answer 34

Note: the program will not run until all the stages have been added.

Stage 2

At this point, if the user presses the spacebar, the solution is displayed.

The program lines to display the solution should be similar to those that define the fifth number. ie. lines 110-130

```
    210 PRINT "S=A*B+C-D":GOTO 300
    220 PRINT "S=A+B+C-D":GOTO 300
    230 PRINT "S=A/B+C*D":GOTO 300
```

Note the order of the solutions. You can see that it is the same as the order of the equations. This means that you can use the variable R again, but this time to display the solution.

Answer 35

```
    ON R GOTO 210,220,230
```

You may well need to use the computer to work out the solution. If you do, stop the program running by pressing the ESC key twice and when you have finished working it out, type in the following command:

```
    CONT
```

When you press ENTER, the computer CONTinues to run the program from the point that you stopped it at.

CONT

Improving the Program

You may have noticed that the fifth number is not always a whole number.

☐ Change the program so that when this occurs, another set of random numbers is displayed.

Hint: use the INT command. **Answer 36**

You could also change the program so that if a negative number is displayed, it is treated in the same way as numbers with fractions. **Answer 37**

Summary

We will end this chapter by asking you a few questions about it.

? The RND command tells the computer to choose a number between ______ and ______.

? What would be displayed if you typed in the following line:

```
PRINT INT(Ø.9)
```

? The following command:

```
ON 3 GOTO 3,3ØØ,5Ø,1ØØØ
```

causes the computer to continue execution from line ______.

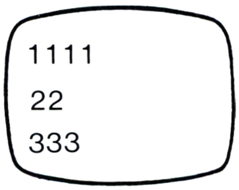
? The ______ command tells the computer to continue running the program.

Chapter 4

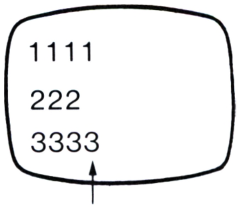
A HORSERACE GAME

In this chapter, we will develop a 'Horserace' game:

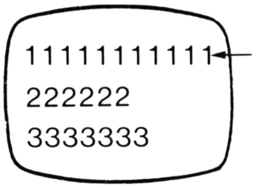
- Horses 1, 2 and 3 are in a race together. The first horse to reach the (specified) 'line' is the winner.



- The computer randomly chooses a number between 1 and 3 inclusively. These numbers represent the horses in the race. Therefore, if the number 3 is chosen, horse 3 must be moved one position to the right.



- It is the first horse to reach the 'line' that wins the race. In our example, horse 1 has won the race because the number 1 was chosen more times than 2 or 3.



Below is a listing of our solution to the problem, however, it has got a few 'bugs' in it.

```
10 MODE 0
15 X1=1:X2=1:X3=1
20 R=INT(RND*3)+1
25 IF X1 OR X2 OR X3=21 THEN END
30 ON R GOTO 50,60,70
50 LOCATE X1,5:PRINT "1":
X1=X1+1
60 LOCATE X2,10:PRINT "2":
X2=X2+1
70 LOCATE X3,15:PRINT "3":
X3=X3+1
100 GOTO 20
```

Type the program into the computer and run it.

Where is the first bug and how do we overcome it? **Answer 38**

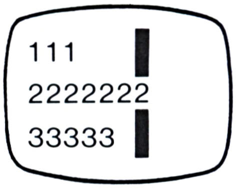
When you have corrected the first bug, run the program a few times.

You can see that the outcome is always the same. Horse 3 wins followed by horse 2 and finally, horse 1.

Change the program so that the outcome is always different. **Answer 39**

Improvements

- Draw a finishing line on the screen.



- Modify the program so that the computer asks you which horse you want to bet on.

DIM

At present, there are only three horses in the race, however, if you wanted to add more horses to make the race more exciting, you would end up copying a lot of the existing lines.

It is possible to add more horses to the race and at the same time, shorten the program.

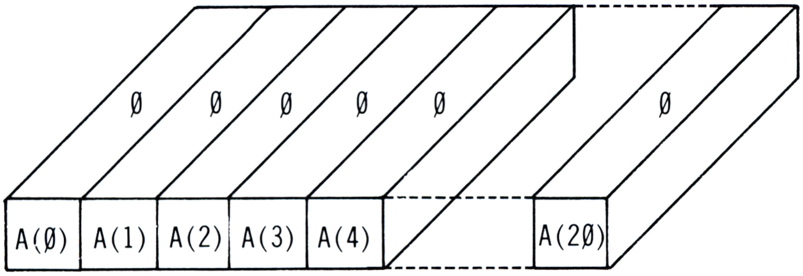
In order to do this, you must learn a new command: the DIM command. **DIM**

Save the program currently in memory and type in the following line:

```
    DIM A(2Ø)
```

It appears that nothing has happened, but, inside the computer, 2Ø memory locations have been DIMensioned (defined); A(Ø) to A(2Ø).

At this point, they all contain the value of Ø.

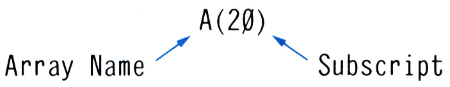


Note: although the first location defined is Ø, in our case A(Ø), we will not use it. The reason for this will become apparent as you get familiar with the use of the DIM command.

Arrays

A group of memory locations that have the same name (A) is called an ARRAY.

Obviously, each location (element) must have a unique name or it would be impossible to access each individual element. Therefore, each element in an array is given a number (Subscript) that makes it different from the other elements.



Tell the computer to display the value of A(12).

The computer displays the number ________(1, Ø).

What would happen if you displayed the value of A(25)?

The computer's response would be to display:

```
    Subscript out of range
```

All the computer is trying to tell you is that there isn't a element with a subscript of 25 and as you will remember, we only dimensioned 2Ø elements.

A Limit on DIM

What would happen if you dimensioned an array with 1ØØØØ elements? Try it.

```
    DIM Z(1ØØØØ)
```

The computer displays the following message:

```
    Memory full
```

In other words, the computer doesn't have enough room in memory for an array of this size.

What does the following program do:

```
    1Ø DIM C(4)
    2Ø FOR N=1 TO 4
    3Ø INPUT C(N)
    4Ø NEXT N
```

When you run the program, the computer asks you for the value of C(1). After inputing a number and pressing ENTER, the computer asks you for another value, C(2) and so on upto C(4).

Note the way that we used the variable N to access each element in the array.

☐ Now instruct the computer to display the contents of C(1). The number displayed should be the same as the first number that you input.

☐ Add lines to the program which will display the contents of each element in the array.

Answer 40

☐ Erase the computer's memory and dimension the following array:

```
DIM Z(5Ø)
```

Let's assume that you want to enlarge the array to have 100 elements. You would probably type in:

```
DIM Z(1ØØ)
```

However, on pressing ENTER, the computer displays an error message:

```
Array already dimensioned
```

Conclusion

The computer cannot dimension an array with the same name, twice.

If for some reason you need to change the size of an array, type in the following command:

```
CLEAR
```

CLEAR

This command clears the computer's memory of any variables or arrays that have previously been defined but will not erase the program in memory.

Note: there is no need to dimension an array that has less than 11 elements in it.

To prove this, type in the following commands:

```
PRINT G(9)
PRINT G(11)
```

On executing the second command, the computer displays:

```
Subscript out of range
```

In other words, you tried to access an element in the array which has not been defined and as we have said, only 1Ø elements are given to an undimensioned array.

Erase the computer's memory and execute the following commands:

```
DIM A(7)
DIM B(3Ø)
DIM C(5ØØ)
```

What does the computer display when you tell it to:

```
PRINT A(6)
PRINT B(4Ø)
PRINT D(8)
PRINT C(5ØØ)
DIM A(13)
```

Answer 41

Type in and run the following program:

```
1Ø DIM C(15)
2Ø R=INT(RND*15)+1
3Ø C(R)=5
4Ø PRINT R
```

Which elements of the array equal Ø and which equal 5?

Check your answer on the computer.

The program randomly chooses an element in the array and puts the value of 5 into it.

What will the program do if you change lines 3Ø and 4Ø to the following:

```
30 C(R)=C(R)+5
40 GOTO 20
```

Does anything appear on the screen when you run this program?

Does the program continuously run?

The computer continuously chooses an element in the array and adds 5 to it.

Stop the program running and display all the elements in the array using the following line:

```
FOR N=1 TO 15:PRINT "C(";N;
")=";C(N):NEXT
```

You can see that each element has a different value depending on the amount of times it was randomly chosen.

Back to the Horseracing

You should now be able to shorten the horserace program described at the start of this chapter and also include the following improvements:

- Increase the number of horses to 9.
- Only one horse number is moved across the screen rather than a long line of numbers.
- Stop the race when the first horse reaches the line.

- Display the number of the horse that wins. **Answer 42**

- When there are a number of people playing the game, the computer asks for each player's name and the horse he/she wants to back. At the end of the race, the computer displays the number of the horse which wins and its backer.

- The computer also displays the players names and the numbers of the horses which came 2nd and 3rd.

Chapter 5

STRING GAMES

A Word Recognition Game

We will start this chapter off by developing a 'Word Recognition Game' which should work as follows:

- The computer stores a few words in memory, such as the following words:

  ```
  RESOLUTION, INTERFACE, MICROPROCESSOR,
  SUBROUTINES, INPUT, DATA, READ, MEMORY
  PERIPHERAL, REGISTER, POINTER, HARDWARE,
  SOFTWARE
  ```

- The computer then randomly chooses one of the words and displays it on the screen. You must try to spell the word correctly.

  If you do so, the computer displays another word. If you don't, the computer informs you of the error and displays the word again for a longer period of time.

Try to write this program on your own, however, if you should have any difficulties as we expect you will, continue reading.

String Arrays

The easiest way to display strings randomly is by entering the words into a STRING ARRAY. This allows us to randomly choose an element in the array and thus, display the word.

String Arrays

String arrays work basically in the same way as numeric arrays. Below is an example of a string array:

```
DIM A$(3)

A$(1)="AMSTRAD"
A$(2)="COMPUTER"
```

- A string array does not need to be dimensioned if it has less than 11 elements in it.

- The length of a string cannot be longer than 255 characters.

? In the above example, what is the value of A$(3).

Answer 43

Putting the Words into the Array

As we said before, we must put the words into an array. There are a number of ways to do this and below is one way using INPUT.

Fill in the missing commands:

```
10 DIM W$(11)
20 FOR _____
30 INPUT _____
40 NEXT
```

Answer 44

The problem with using INPUT is that everytime you want to use the program, you have to fill up the array manually, which is a long process.

To overcome this, you have to learn a new command:

READ...DATA

Erase the computer's memory and type in the following program:

```
    10 READ X
    20 DATA 5,6,7
```

Note how the numbers are separated.

Run the program.

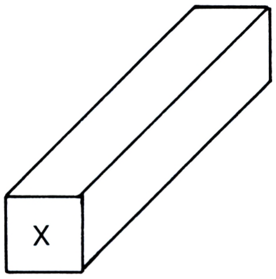
It appears that nothing has happened but if you print the value of X, you will see that something has.

What did the computer do when you executed this program.

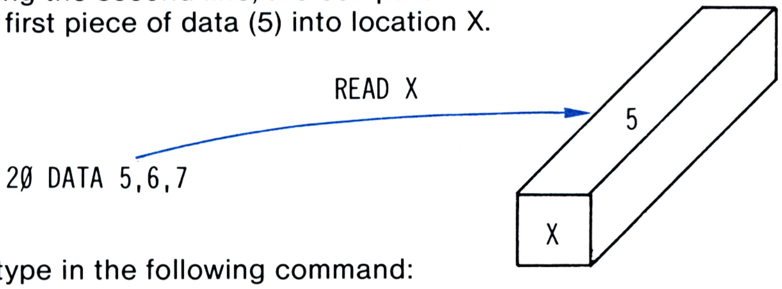
When it executed the first line:

```
    10 READ X
```

it set up a location in memory and called it X.



On executing the second line, the computer copies the first piece of data (5) into location X.



Now type in the following command:

```
    READ X
```

What is the value of X now?

Check your answer on the computer.

Explanation

Within the program is a DATA statement which holds the data that the program requires. Before executing the READ command, the Data Pointer points to the first piece of data, which, in this case, is 5.

Data Pointer

```
20 DATA 5,6,7
        ↑
     Data Pointer
```

When the computer executes the READ command, it puts the value that the data pointer is pointing to, into X. ie. X=5. The data pointer is then automatically moved to the next item of data.

```
20 DATA 5,6,7
          ↑
       Data Pointer
```

Therefore, when the computer executed the READ command for the second time, X received the value of __________.

What would the value of X be after instructing the computer to READ X for the third time?

What about the fourth time?

On the fourth time, the computer displays the following error message:

```
DATA exhausted
```

which means that there is no more data left, and as you can see from the program, there is only three items of data.

What will appear on the screen when you run the following program:

```
10 FOR N=1 TO 4
20 READ X
30 PRINT X
40 NEXT N
50 DATA 1,3,8,13
```

Change the program so that only the first two numbers (1 and 3) are displayed.

Answer 45

What would happen if you put the DATA statement before the READ command? Try it.

Conclusion

The DATA statement can go anywhere in the program. When the computer executes a READ command, it takes the data from the first DATA statement that it finds in the program, however, it is usual to write the DATA statement at the end of the program.

What will the computer display on the screen when you run this program?

```
10 READ X
20 READ Y
30 PRINT X/Y
40 DATA 10,2
```

Think about it first and then run the program.

Task

By adding the GOTO command, change the program in memory so that the computer displays the following equations with there respective answers:

```
11 / 17 =
35 / 18 =
19 / 37 =
34 / 83 =
96 / 33 =
```

Answer 46

Did the computer display the 'DATA exhausted' error message?

We have already seen one way of overcoming this (using FOR...NEXT), however, below is another way:

```
10 READ X
20 READ Y
25 IF X=-1 THEN END
30 PRINT X;"/";Y;"=";X/Y
35 GOTO 10
40 DATA 11,17,35,18,19,37,34
,83,96,33,-1,-1
```

Can you explain why this program does not display the DATA exhausted message?

Is it necessary to have both '-1' at the end of the DATA statement? Erase the last one and see what happens.

Answer 47

Note

In the program above, there are two READ commands on lines 10 and 20. It is possible to include both READ variables into one READ command as follows:

```
10 READ X,Y
```

The computer still takes two pieces of data from the DATA statement, where X equals the first piece (11) and Y equals the second piece (17).

On executing the READ command for a second time, the following two pieces of data (35 and 18) are put into X and Y respectively.

Task

There is a bug in the following program. Sort out what it is and correct the program accordingly.

```
10 FOR N=1 TO 5
20 DIM X(5)
30 READ X(N)
40 NEXT N
50 DATA 1,13,-5,7,0
```

Answer 48

? What happens when you run the following program?

```
10 READ A$
20 PRINT A$
30 GOTO 10
40 DATA PAUL,ELAINE,LISA,JOHN
```

Explanation

The READ and DATA commands can also handle strings. Line 10 sets up a string variable (A$) and puts the first piece of data that it finds in the DATA statement (PAUL), into A$.

Note: you do not need to put quotation marks around the four strings in the DATA statement. As with the string INPUT command, the computer knows what type of data to expect by the variable used in the program.

? What would happen if you took the '$' out of lines 10 and 20 and then executed the program?

RESTORE

☐ After putting the '$'s back into the program, enter the following line:

```
25 IF A$="JOHN" THEN RESTORE
```

You may remember that when we first executed the program, a 'DATA exhausted' message was displayed.

Now, after adding line 25, the program continuously displays the strings in line 4Ø.

The RESTORE command returns the Data Pointer to the first piece of data in the DATA statement. In our case, the following happens:

RESTORE

```
4Ø DATA PAUL,ELAINE,LISA,JOHN

                RESTORE
```

Now, when READ is executed again, the first piece of data is put into A$.

Task

☐ List the program currently in memory.

Swap the RESTORE command in line 25 for another command, without changing the overall effect of the program.

Answer 49

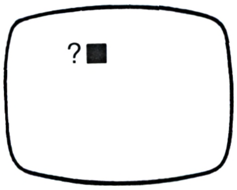
Conclusion

When you execute a RUN command, whether it be immediately or within a program, it moves the Data Pointer back to the first piece of data, as well as reseting variables etc.

Task

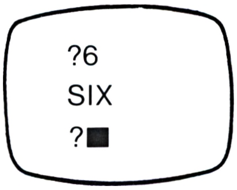
Write a program to do the following:

- The computer asks you to choose a number between 1 and 1Ø.



```
?■
```

- Let's assume that you chose the number 6. On pressing ENTER, the the word 'SIX' is displayed. The computer then waits for you to input another number.



```
?6
SIX
?■
```

Thus, the computer displays the number input, in word form.

Hint

What number is displayed when you run the following program:

```
1Ø FOR N=1 TO 3
2Ø READ A
3Ø NEXT N
4Ø PRINT A
5Ø DATA 7,6,5,4,3
```

Answer 50

Back to the Word Recognition Game

You should now be able to write the program that we described at the start of this chapter. Read over the description before attempting to write it.

Answer 51

Improvements

- If you misspell the word twice, the computer displays the word again and waits for you to press a key before continuing.

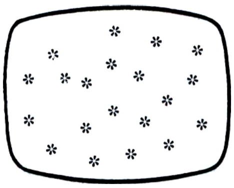
- If you spell the word correctly, the computer doesn't bother displaying it again.

Chapter 6

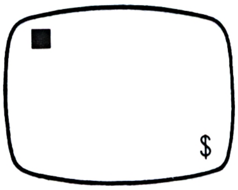
A TREASURE HUNT

In this last chapter, we will develop a game that will use most of the commands taught during this unit.

- On running the program, the computer randomly displays 20 'mines'. After about five seconds, the mines disappear from the screen.

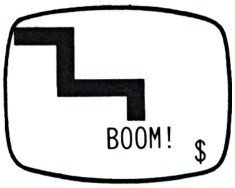


- When the mines have disappeared from the screen, but not from memory, the treasure hunter (■) and the treasure ($) appear.



The aim of the game is to get the treasure hunter to the treasure, without stepping on any of the mines.

- The treasure hunter is only allowed to move to the right and down. If he steps on a mine, the computer displays 'BOOM!'.



You should be able to write this program with the knowledge acquired during this unit. We advise you to write this program stage by stage but should you have any problems with it, you can always look at the following hints.

Hint 1

The main problem in programming this game is keeping track of the mines. We need to do this so that if the treasure hunter steps on a mine, we can blow him up.

In order to do this, we need to dimension two arrays:

```
    DIM X(2Ø)          DIM Y(2Ø)

      X(1)               Y(1)
      X(2)               Y(2)
      X(3)               Y(3)
       .                  .
       .                  .
      X(2Ø)              Y(2Ø)
```

```
1st mine is located at X(1),Y(1).
2nd mine is located at X(2),Y(2).
3rd mine is located at X(3),Y(3).
2Øth mine is located at X(2Ø),Y(2Ø).
```

The program to do this should look something like this:

```
     5 MODE Ø
     1Ø DIM X(2Ø),Y(2Ø)
     2Ø FOR N=1 TO 2Ø
     3Ø X(N)=INT(RND*2Ø)+1
     4Ø Y(N)=INT(RND*24)+1
     5Ø LOCATE X(N),Y(N):PRINT "*"
     6Ø NEXT N
```

Note the way we dimensioned two arrays in one line.

Hint 2

At some point in the program, the treasure hunter's position needs to be compared with the positions of the mines.

(Use the FOR...NEXT command.)

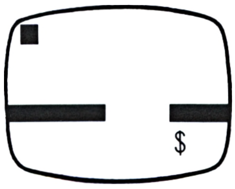
Note: these hints are directed towards one solution. Obviously, there are other ways to construct this program and we will leave it up to you to choose.

The full program is listed in Answer 52.

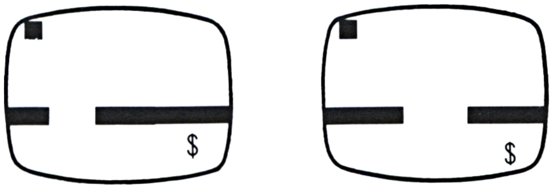
Answer 52

Improving the Program

- When the treasure hunter steps on a mine, the computer produces an explosion.
- As you get nearer to the treasure, a rising pitch sound is produced.
- When you reach the treasure, the computer plays a tune.
- If the game is too easy, put an extra obstacle in the way such as a hidden wall, which has only one opening.



- If the game is still too easy, vary the position of the opening in the wall.



- If the player gets lost, he/she is allowed to display the obstacles (mines and wall) once only.

SUMMARY INDEX

Below is a list of the commands and techniques taught during this unit.

Go over each one of them and see if you can remember them.
(Explanations are given on the pages stated in brackets.)

Video Pencil	(7)	OR	(31)
Greater Than	(10)	ON...GOTO	(34)
Less Than	(10)	CONT	(36)
INKEY$	(11)	DIM	(40)
Joysticks	(16)	Arrays	(41)
JOY(Ø)	(16)	CLEAR	(43)
Part LIST	(25)	String Arrays	(47)
RND	(27)	READ...DATA	(49)
INT	(28)	Data Pointer	(50)
AND	(31)	RESTORE	(54)

To Conclude

Now that you have finished the third unit in this series and acquired a great deal more knowledge about the Amstrad, you may be interested in 'Amstrad Unit 4' which goes into high resolution graphics.

ANSWERS

Answer 1

```
10 CLS
20 INPUT A
30 IF A=1 THEN PRINT "ONE"
40 IF A=2 THEN PRINT "TWO"
50 IF A=3 THEN PRINT "THREE"
60 IF A=4 THEN PRINT "FOUR"
 .
 .
 .
110 IF A=10 THEN PRINT "TEN"
150 GOTO 20
```

Answer 2

```
10 CLS
20 INPUT A$
30 IF A$="P" THEN PRINT "PAUL"
40 IF A$="L" THEN LIST
50 IF A$="N" THEN NEW
100 GOTO 20
```

Answer 3

```
10 CLS
20 X=20:Y=20:LOCATE X,Y:PRINT "*"
30 INPUT A$
40 IF A$="K" THEN X=X+1
50 IF A$="J" THEN X=X-1
60 IF A$="I" THEN Y=Y-1
70 IF A$="M" THEN Y=Y+1
80 LOCATE X,Y:PRINT "*"
100 GOTO 30
```

Answer 4

```
45 IF X=41 THEN X=40
55 IF X=0 THEN X=1
65 IF Y=0 THEN Y=1
75 IF Y=26 THEN Y=25
```

Note: we have not restricted the movement of the pencil into the bottom right hand corner of the screen, even though this does cause the program to go wrong. This is because the tests to do this are rather more difficult than the tests we are currently doing.

Answer 5

The video pencil doesn't move because the letters tested for are all uppercase, and when you press CAPS LOCK, you input a lowercase letter.

Answer 6

Stop the program running and display the contents of A$. As you can see, A$ equals nothing.

Answer 7

Add the following line:

```
35 SOUND 1,50
```

Answer 8

Add the following lines:

```
15 B$="*"
20 X=20:Y=10:LOCATE X,Y:PRINT B$
77 IF A$="↑" THEN GOTO 150
80 LOCATE X,Y:PRINT B$
170 GOTO 30
```

Answer 9

```
↑=24Ø
↓=241
←=242
→=243
```

Answer 1Ø

```
4Ø IF A$=CHR$(243) THEN X=X+1
5Ø IF A$=CHR$(242) THEN X=X+1
6Ø IF A$=CHR$(24Ø) THEN Y=Y-1
7Ø IF A$=CHR$(241) THEN Y=Y+1
```

Answer 11

Add these lines:

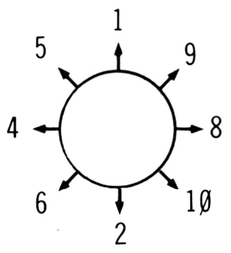
```
79  IF A$="C" THEN GOTO 2ØØ
2ØØ LOCATE 1,25:INPUT "COLOUR";C
21Ø PEN C
22Ø GOTO 3Ø
```

Answer 12

Add the following lines:

```
35 IF A$="-" THEN K=1
37 IF A$="=" THEN K=Ø
9Ø IF K=1 THEN LOCATE X,Y:PRINT " "
```

Answer 13



Answer 14

If you press fire, the number 16 is displayed. If you move the joystick and at the same time, press fire, the direction value is added to the fire value (16).

For example, if you move the joystick to the right and press fire:

```
    JOY(Ø)=8+16
```

which equals 24.

Answer 15

Add these lines:

```
     4Ø IF JOY(Ø)=8 THEN X=X+1
     5Ø IF JOY(Ø)=4 THEN X=X-1
     6Ø IF JOY(Ø)=1 THEN Y=Y-1
     7Ø IF JOY(Ø)=2 THEN Y=Y+1
```

Answer 16

```
     1Ø CLS
     2Ø LOCATE 35,24:PRINT "$"
     3Ø FOR X=1 TO 4Ø
     4Ø LOCATE X,5:PRINT ">"
     5Ø FOR T=1 TO 5Ø:NEXT T
     6Ø LOCATE X,5:PRINT "□"
     7Ø NEXT X
     2ØØ GOTO 3Ø
```

Answer 17

Add these lines:

```
    43 A$=INKEY$
    45 IF A$="Ø" THEN GOTO 3ØØ
```

Answer 18

Add these lines:

```
300 FOR Y=23 TO 1 STEP -1
310 LOCATE 35,Y:PRINT "↑"
320 FOR T=1 TO 50:NEXT T
330 LOCATE 35,Y:PRINT "□"
340 NEXT Y
```

Answer 19

Delete line 50 and add the following line:

```
55 SOUND 1,X+50,10
```

Answer 20

Add these lines:

```
315 SOUND 1,Y+50,10
317 IF SQ(1)>127 THEN GOTO
317
```

Answer 21

Add this line:

```
350 GOTO 10
```

Answer 22

Add these lines:

```
15  LOCATE 1,1:PRINT N
345 N=N+1
```

Answer 23

```
IF X=35 THEN LET M=M+1
```

Answer 24

```
17 LOCATE 1,3:PRINT M
347 IF X=35 THEN M=M+1
```

Answer 25

Add the following line:

```
18 IF N=1Ø THEN END
```

Answer 26

```
1Ø PRINT RND*6
```

Answer 27

```
1Ø PRINT INT(RND*6)+1
```

Answer 28

```
1Ø CLS
2Ø FOR N=1 TO 13
3Ø PRINT INT(RND*3)
4Ø NEXT N
```

Answer 29

```
1Ø MODE Ø
2Ø LOCATE INT(RND*2Ø)+1,INT(
RND*25)+1
3Ø PEN INT(RND*16)
4Ø PRINT "*"
5Ø SOUND 1,INT(RND*2ØØ)+5Ø
1ØØ GOTO 2Ø
```

Answer 3Ø

```
10 CLS
20 LOCATE 20,10:PRINT "*****"
30 INPUT A
40 FOR N=1 TO A
50 X=INT(RND*40)+1
60 Y=INT(RND*25)+1
70 LOCATE X,Y:PRINT "O";
100 NEXT N
```

Answer 31

Add the following lines:

```
75 IF X=20 AND Y=10 THEN END
77 IF X=21 AND Y=10 THEN END
80 IF X=22 AND Y=10 THEN END
82 IF X=23 AND Y=10 THEN END
84 IF X=24 AND Y=10 THEN END
```

Answer 32

Change the following line:

```
30 ON R GOTO 40,60,50,70
```

Answer 33

```
20 R=INT(RND*4)+1
```

Answer 34

```
10 CLS
20 A=INT(RND*9)+1
30 B=INT(RND*9)+1
40 C=INT(RND*9)+1
50 D=INT(RND*9)+1
60 R=INT(RND*3)+1
70 LOCATE 15,10:PRINT A;"□";B;
"□";C;"□";D
80 LOCATE 21,12
90 ON R GOTO 110,120,130
110 S=A*B+C-D:PRINT S:GOTO 180
120 S=A+B+C-D:PRINT S:GOTO 180
130 S=A/B+C*D:PRINT S:GOTO 180
```

Answer 35

Add the following lines:

```
180 A$=INKEY$
190 IF A$<>" " THEN GOTO 180
200 ON R GOTO 210,220,230
210 PRINT "S=A*B+C-D":GOTO 300
220 PRINT "S=A+B+C-D":GOTO 300
230 PRINT "S=A/B+C*D":GOTO 300
300 A$=INKEY$
310 IF A$="" THEN GOTO 300
320 GOTO 10
```

Answer 36

Add the following lines:

```
180 IF INT(S)<>S THEN GOTO 10
185 A$=INKEY$
```

Answer 37

Add this line:

```
182 IF S<0 THEN GOTO 10
```

Answer 38

The first bug is in line 25. It should read as follows:

```
25 IF X1=21 OR X2=21 OR X3=21
THEN END
```

Answer 39

The reason why horse 3 always wins is because line 7Ø is always executed. For example, the random horse chosen is 2. The computer executes line 6Ø as expected but when it finishes, it executes line 7Ø as well. ie. both horse 2 and 3 are moved.

```
5Ø LOCATE X1,5:PRINT "1":
X1=X1+1:GOTO 2Ø
6Ø LOCATE X2,1Ø:PRINT "2":
X2=X2+1:GOTO 2Ø
7Ø LOCATE X3,15:PRINT "3":
X3=X3+1:GOTO 2Ø
```

And you can of course, delete line 1ØØ.

Answer 4Ø

Add the following lines:

```
14Ø FOR N=1 TO 4
15Ø PRINT "C(";N;")=";C(N)
16Ø NEXT N
```

Answer 41

Displaying the contents of A(6) will produce Ø.

Displaying the contents of B(4Ø) will produce a 'Subscript out of range' error.

Displaying the contents of D(8) will produce Ø.

Displaying the contents of C(5ØØ) will produce Ø.

Executing DIM A(13) will produce an 'Array already dimensioned' error.

Answer 42

```
10 MODE 2
20 FOR I=1 TO 9
30 X(I)=1
40 NEXT I
50 R=INT(RND*9)+1
60 LOCATE X(R),2*R:PRINT R:
X(R)=X(R)+1
70 IF X(R)=79 THEN GOTO 200
80 GOTO 50
200 LOCATE 1,20:PRINT "THE
WINNER IS HORSE.";R
```

Answer 43

A$(3) has not been defined. It therefore has a value of "" (nothing).

Answer 44

```
10 DIM W$(11)
20 FOR N=1 TO 11
30 INPUT W$
40 NEXT N
```

Answer 45

Change this line:

```
10 FOR N=1 TO 2
```

Answer 46

Add the following lines:

```
30 PRINT X;"/";Y;"=";X/Y
35 GOTO 10
40 DATA 11,17,35,18,19,37,
34,83,96,33
```

Answer 47

It is necessary to have both '-1's in the DATA statement because the first -1 is not tested for until both READ command have been executed.

Answer 48

The bug is caused by putting DIM X(5) into the loop. As we have already said, an array with the same name cannot be dimensioned twice.

To correct the program, move line 20 to line 5.

Answer 49

```
25 IF A$="JOHN" THEN RUN
```

Answer 50

```
10 CLS
20 INPUT A
30 FOR N=1 TO A
40 READ A$
50 NEXT N
60 PRINT A$
70 RESTORE:GOTO 20
100 DATA ONE,TWO,THREE,FOUR,
FIVE,SIX,SEVEN,EIGHT,NINE,TEN
```

Answer 51

```
10 Z=50
20 CLS
30 R=INT(RND*5)+1
40 FOR N=1 TO R
50 READ A$
60 NEXT N
70 LOCATE 15,10:PRINT A$
80 FOR T=1 TO Z:NEXT T
90 CLS
100 INPUT B$
110 IF A$=B$ THEN RUN
120 SOUND 1,200
130 Z=Z+200
140 RESTORE:GOTO 40
150 DATA INPUT,DATA,READ,
MEMORY,POINTER
```

We will leave it to you to add the rest of the words. Remember that line 30 will also have to change depending on the amount of words you have.

Answer 52

```
10 MODE 0:DIM X(20),Y(20)
30 FOR N=1 TO 20
40 X(N)=INT(RND*20)+1:Y(N)=
INT(RND*24)+1
60 LOCATE X(N),Y(N):PRINT "*"
80 NEXT N
90 FOR T=1 TO 3000:NEXT T:CLS
100 PRINT CHR$(143)
110 LOCATE 19,23:PRINT "$"
120 X=1:Y=1
150 A$=INKEY$:IF A$="" THEN
GOTO 150
160 IF A$=CHR$(243) THEN
X=X+1:IF X=21 THEN X=20
170 IF A$=CHR$(241) THEN
Y=Y+1:IF Y=24 THEN Y=23
180 LOCATE X,Y:PRINT CHR$(143)
```

```
200 FOR N=1 TO 20
210 IF X(N)=X AND Y(N)=Y THEN
 GOTO 400
220 NEXT N:GOTO 150
400 LOCATE X,Y:PRINT "BOOM!"
410 FOR A=1 TO 1000:NEXT A
420 RUN
```

NOTES

This series of self-instruction books will teach you the secrets of writing programs in BASIC on your AMSTRAD computer.

Unit 1: **FIRST STEPS IN BASIC**
Starting with the first things every programmer need to know, you will learn to issue commands to the computer, as well as writing and running programs. By the end of the unit you'll be able to make your computer perform useful and interesting tasks.

Unit 2: **EXPLORING BASIC**
This unit teaches you the most important concepts of BASIC: numeric variables, string variables, FOR . . . NEXT and IF . . . THEN statements, and much more. You'll create a digital computer clock, and interesting graphics programs including animation.

Unit 3: **COMPUTER GAMES**
In this unit you will learn to develop various computer video games. As you progress through the unit, new programming concepts such as random numbers will be introduced. By the end you'll have considerable programming skills.

And after that . . . we're planning more units to deal with special subjects, such as 3D graphics, machine code and more.

£3.95

ISBN 1-85181-005-6

9 781851 810055

GLENTOP
PUBLISHERS LIMITED

Glentop Publishers Ltd.,
Standfast House, Bath Place,
High Street, Barnet, Herts. EN5 1ED
Tel: 01-441 4130

AMSTRAD 464, 664 & 6128

COMPUTER GAMES

WATSON'S NOTES

VOL 3