AMSTRAD AMSTRAD AMSTRAD AMSTRAD

Computing with the AMSTRAD

INC. AMTIX!

$4.50

The independent magazine for Amstrad computer users
Registered by Australia Post – Publication No. VBP7811

JULY, 1987
A Database Publication

- **NEW 12 PAGE AMTIX BONUS**
- MACHINE CODE • BUSINESS SECTION • SOUND •
- NEWS • GRAPHICS • GAME OF THE MONTH •
- SAVINGS ON SOFTWARE • AND MUCH MORE •

$AVINGS ON $OFTWARE

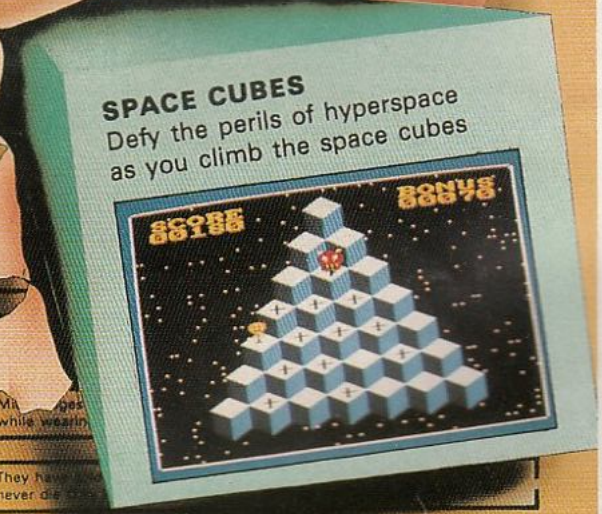
AMSTRAD – AMSTRAD –

Chart Busters

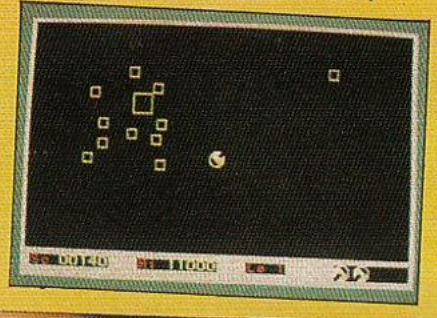
Four action-packed classics to challenge the most expert gamester

PLANET RESCUE
New Earth has been invaded.
Can you rescue the survivors?

LEAPER
Escape the rush-hour chaos . . .
then brave the raging torrent



SPACE CUBES
Defy the perils of hyperspace
as you climb the space cubes



NEBULAE
Surrounded by deadly anti-matter,
destruction is a millisec away . . .

		Last month	Months in chart	
1			1	
2			2	
3			1	Faith add
4				
5				
6				he famous action. High
7				Gauntlet style view game. Very addictive with a huge map.
8	Knight	ne	1	Cute combination of adventure and arcade. Pop down menus and clever puzzles make this a very different game.
9	**Speed King** M	ne	1	Fast, high-speed action. Budget priced and certainly worth looking at.
10	**Thrust** *Firebird*	ne	1	Simple and lun
11	**Kung Fu Master** *US Gold*	11	2	
12	**Radzone** *Mas*	6		
13	**Star Firebi**			
14	**Jack**			
15			3	
16				
17				Mi ges while wearin
18			4	They have never die
19				Helicopter flight simulation which is both accurate and fun.
20			10	

		60
		59
		58
		45
		44
		24

Non-mover Up

All 4 games in ONE package

CPC 464, 664, 6128

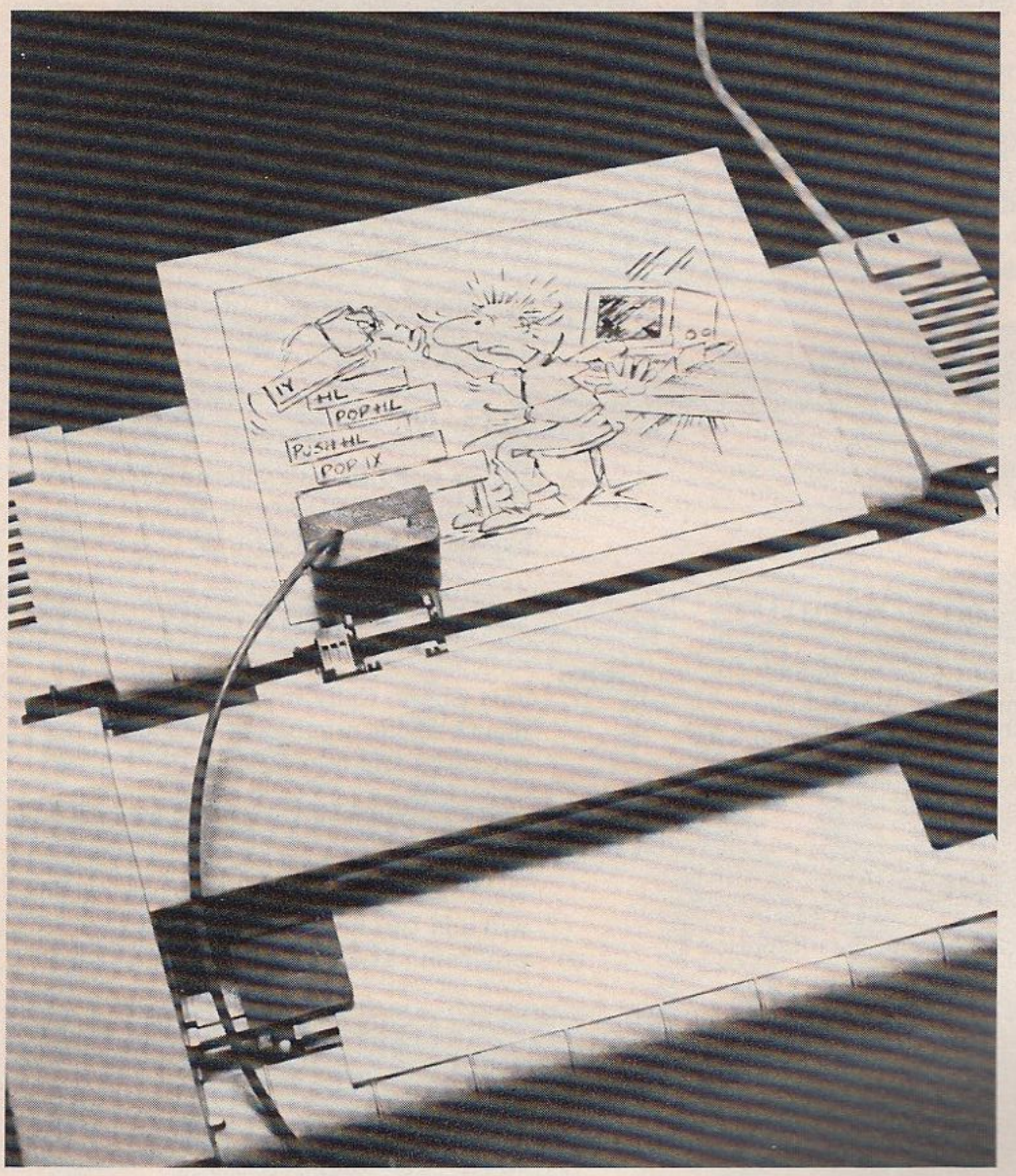
On tape	–	$15.95
On disc	–	$27.95

Use the order form on centre page

Contents

For information and telephone numbers on subscriptions, advertising and contributions, please turn to the business contents on page 65.

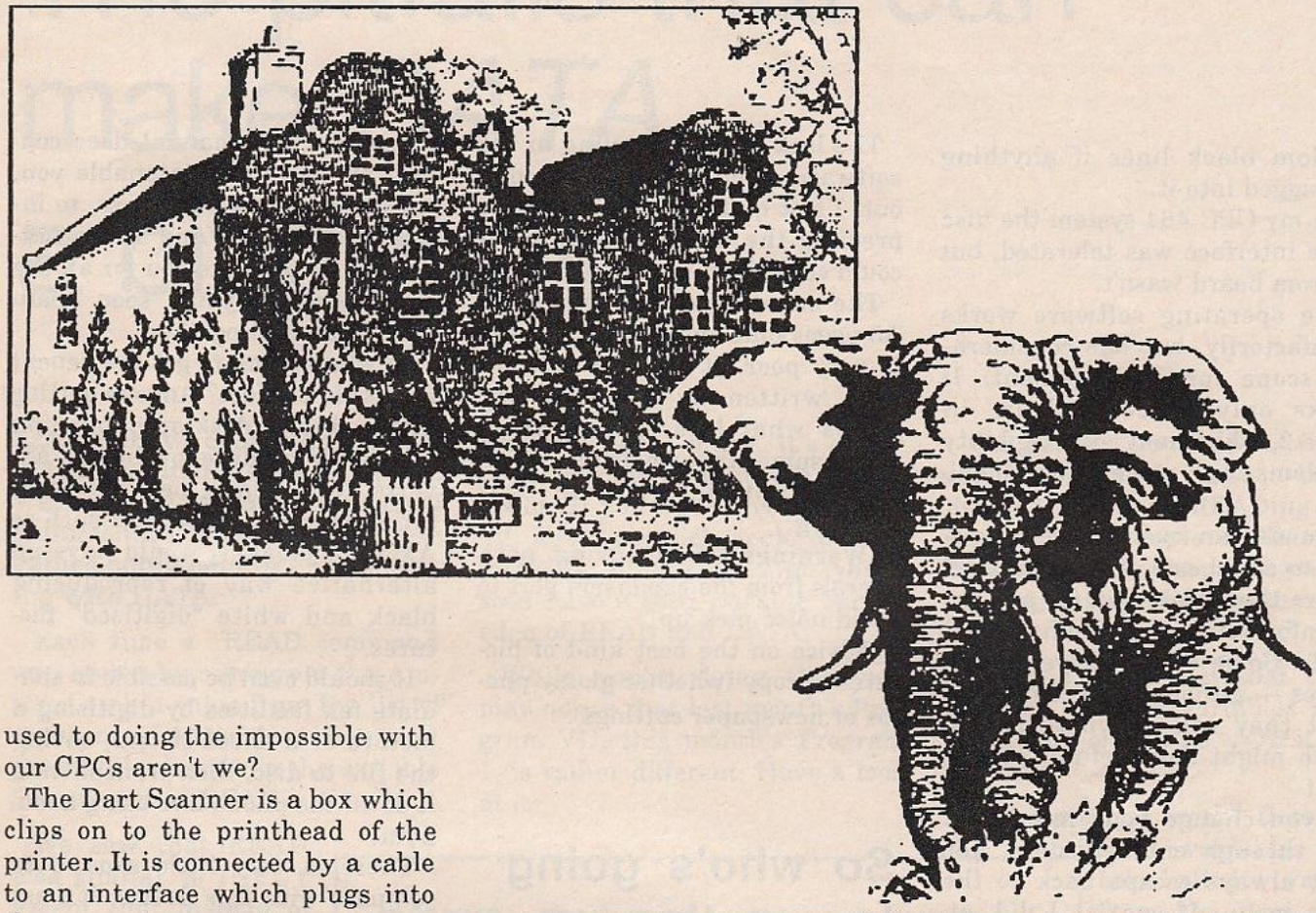
When a printer becomes an input device



TONY HUNTINGDON examines an alternative to the video digitiser

A printer is an output device which prints words and pictures. Wrong! Take one Amstrad DMP printer, add a Dart Scanner and your printer is now an input device which will read words and pictures into your computer.

Incredible? Impossible? Of course it is; but we are



used to doing the impossible with our CPCs aren't we?

The Dart Scanner is a box which clips on to the printhead of the printer. It is connected by a cable to an interface which plugs into the expansion port of your micro.

Using the software provided (on both disc and tape) you get the printer mechanism to scan a picture and "digitise" it.

The hardware is in three distinct pieces. The scanner head is a sealed unit which appears to contain a bulk and a photo-diode.

Its companion on the printer is a small plastic moulding which has to be mounted on the extreme left hand end of the scanner's travel by means of a piece of double-sided adhesive tape. Although this appears to be a flimsy arrangement, it gave no trouble at all.

The scanner head is linked to the interface box by about a metre of cable. This is rather short, allowing little choice as to where to position the printer, but as the cable is wired in, there is no scope for extending it.

The only hardware control is a "brightness" knob, used to set the threshold at which the scanner decides that the grey level it sees is either black or white.

On the back of the interface box is an edge connector. Don't be misled into thinking that you can add your usual peripherals to it. The scanner is very sensitive to "noise" pickup, and may produce

Using the Dart Scanner, these two cuttings from a magazine (above) and a photograph (right) were reproduced on the DMP printer

random black lines if anything is plugged into it.

On my CPC 464 system the disc drive interface was tolerated, but the rom board wasn't.

The operating software works satisfactorily, but has considerable scope for improvement. It works only in two colours in Mode 2, which may give legibility problems with some colour monitors.

You use an overlay menu system to select one of 12 functions. You're then prompted for any extra information that the hardware needs. Some entries have default values — although you're not told what they are — while others, which might have default values, don't.

If you change your mind part way through entering data, you can't always escape back to the main menu. If you, as I did, attempt to save a picture to a write-protected disc, you will see the standard Amsdos error message. Unfortunately it overwrites, and becomes part of your picture.

Editing facilities are limited. You can copy one area to another, flip left to right or top to bottom, scroll an image in any direction, and add text.

You can create a permanent box frame for any part of the image so that anything outside that area can be cleared, and there's zoom facility for magnifying a selected area of the screen for pixel editing.

Also the final image can be produced on paper from half to six times the size of the original.

If these facilities aren't enough, as pictures are saved as binary screen images you could use this software to acquire and save the pictures and then use another graphics package to edit them.

The lack of idiot-proofing in the software often caught this idiot out. I lost quite a few pictures by pressing the wrong keys before I could save them properly.

The manual, consisting of eight A5 pages plus an addendum sheet, is very poor. It appears to have been written by someone who knows what he's talking about and assumes that we do too.

Important information was missing:

- Warnings on removing peripherals from the expansion port to avoid noise pick up.
- Advice on the best kind of pictures to copy (whether glossy photos or newspaper cuttings).

So who's going to get the benefit from this product? Anyone getting involved with desktop publishing in order to produce low-budget adverts, letterheads

- How to incorporate these into your own programs. I had quite a few unsuccessful attempts until Dart put me on the right lines.

However the manual does contain enough hints to enable you, with a little common sense, to install the scanner. And as the software then prompts you for all the data required, you'll soon abandon the book altogether.

So who's going to get the benefit from this product? Anyone getting involved with desktop publishing in order to produce low-budget adverts, letterheads, new sheets or brochures, using a product such as AMX Pagemaker, would have an alternative way of reproducing black and white "digitised" pictures.

It should even be possible to simulate fax facilities by digitising a picture or a sheet of text, saving the file to disc, then transmitting it down the phone line using a modem.

There is very little around to compare with this device. It's not a picture digitiser in the true sense, as it only produces black and white pictures with no attempt at interpreting grey scales.

The concept of using the printer mechanism to scan pictures is very clever, but it could have been implemented better. The hardware suffers from noise problems, and the cable to the scanner head is a little too short to be convenient.

Even though I had problems with the software, at least it's partially written in (unprotected) Basic so you could attempt to improve it.

I find it difficult to see how Dart justified the high retail price as there are few components, the software needs improving and the manual is minimal.

If you don't have video equipment and are looking for an alternative to a video digitiser, and you're happy with reproduction in pure black and white, then look no further.

The pitfalls that can make *DATA* a bit dodgy

Last month we dealt with the Basic statements READ and DATA. We saw how we could use READ to put values into a variable, these values being held in lines beginning with DATA.

Each time a READ command was issued by a program the Amstrad took a value from the list of constants in the data line and stored this number in the variable following that READ.

We saw that READ was similar to INPUT except that instead of looking to the keyboard for a value, the micro looks to a value held in the program itself.

One major point to grasp is the way that a READ takes information from the list of numbers following a DATA: It does it sequentially.

The first time a READ is obeyed the initial number after the first DATA is taken. The next time a value is required it is taken from the second number following the DATA.

Each time a value is taken from the data list the Amstrad makes a note of where it is up to and sets what is known as a data pointer. This ensures that the next time a READ is issued the Amstrad looks at the next unREAD item in the data list.

If all that sounds a bit formal, don't worry too much. It's easier to use in practice that it is to read about. If you have any doubts at all have a quick look at last month's programs and you'll soon have a good working knowledge of READ and DATA.

While you're doing that you may notice that last month's Program VII, this month's Program 1, is rather different. Have a look at it.

PETE BIBBY digs deeper into DATA and READ in Part Eleven of his series for programming

```
10 REM Program I
20 REM Formerly Program VII
30 sumone=0:sumtwo=0
40 FOR loop=1 TO 5
50 READ first,second
60 sumone=sumone+first
70 sumtwo=sumtwo+second
80 NEXT loop
90 PRINT sumone,sumtwo
100 DATA 1,2,3,4,5
110 DATA 6,7,8,9,10
```

Program 1

The general structure is familiar, it's just the usual FOR...NEXT loop READing values from a couple of lines of DATA. As the loop progresses running totals of the values are kept.

What is unusual is that there are two numeric variables after the READ of line 50 — *first* and *second*.

How does this work? Does the *first* variable first take all its values from the first data line, line 100? And does *second* read its values from line 110?

The answer is no. That would be a bit too complicated. What happens is that when a READ is followed by two or more variables the data pointer keeps on moving down the data line one item at a time.

The first variable takes the value of the first item in the line, the second variable the next and so on. If you think about it, this is much the more sensible method.

Let's see how it works in the case of Program 1.

Line 50, tucked away inside a FOR...NEXT loop that cycles five times, is the one we want to understand. Here a READ is followed by the two variables *first* and *second*.

The first time round the loop, when the READ command is obeyed, the Amstrad has to put a value in *first*. It looks at the first item in the first line beginning with DATA and takes that value. In this case it takes the value 1.

It doesn't stop there, though. There is another variable, *second*, following the READ and this needs a value as well, so it looks to the next free item in the data list — pointed to by the aptly-named data pointer — and takes that value.

So the first time round the loop *first* has the value 1 and *second* the value 2. Lines 60 and 70 just keep the running totals of the numbers stored in *first* and *second* as the loop progresses.

The second time round the loop line 50 again has to look to the data list for a value for *first*. This time the data pointer has reached 3, so *first* takes the value 3.

The *second* is used to store 4, the next value along. As you can see, the two variables take it in turn to get values from the data list. Add a line like:

55 PRINT first,second

if you're still in any doubt.

While you've got Program 1 in your micro, let's just take another look at two common errors that occur when using READ and DATA. Try changing line 110 to:

110 DATA 67,8,9,10

and see what happens. Unless your micro is very different from mine you'll get a:

DATA exhausted in 50

message for your pains. What has happened is that by having 67 in line 110 instead of the previous 6 and 7 separated by a common, we have reduced the number of data items in the list.

Before we had 10, now we have nine. The trouble is that as the loop cycles five times, each time taking two items from the list, the program needs 10 items in the data list.

What happens is that it runs out of data and tells you so. Notice, however, that the message refers to the READ line and not the line where the error actually occurred.

Next, try changing line 100 to:

110 DATA 6,7,8,9,1,0

which could happen if a lazy typist got the line wrong. Now there are 11 items in the list, not 10. Can you remember from last time what will happen when you run the program? Try it and see.

Even though you get the "wrong" answer, there's no error message. The Amstrad just takes the first 10 items. It doesn't know that the 1 and 0 at the end should have been 10. The moral is, be careful typing in data lists.

So far, all the items in data lists have been numbers which, unsurprisingly, we've read into numeric variables. We can, however, use READ and DATA to read in strings, as Program II shows.

```
10 REM Program II
20 FOR family=1 TO 4
30 READ name$
40 PRINT name$
50 NEXT family
60 DATA Peter, Eileen, Bodger, Spot
```

Program II

Here the FOR...NEXT loop cycles four times, each time READing in an item from the data list of line 60. These strings are stored in the string variable *name$* in just the same way as we've seen with numbers.

Try changing the program to read in your own string input and you'll see that reading strings and numeric values from data lists are similar operations. Even the errors that you can make are similar.

Try changing the data list to:

60 DATA Eileen, Bodger, Spot

and:

60 DATA Peter, Eileen, Bodger, Sp,ot

and you'll see what I mean.

Notice that line 60 consists of a list of strings which are assigned to the string variable *name$*. Normally when we assign a string of letters to a variable we enclose them in inverted commas to tell the Amstrad that it is a string and not a variable name.

However, when we're reading from a data list, we needn't use the inverted commas, as the micro can tell from the variable name *name$* that we want a string variable. So, while we can have a line like:

60 DATA"Peter","Eileen", "Bodger","Spot"

it's not usually necessary.

There is, however, one time when it is necessary. Can you think of it? We'll come across it later.

There are a couple of more points to be taken into consideration when reading in strings. The first is that you have to read a string value into a string variable. Try changing line 30 to:

30 READ name

and you get the message:

Syntax error in 60

This is because you've tried to store a string in a numeric variable. Notice that the error message points to the data list and not to line 30 where the mistake actually happened.

From this you can see that the following is good advice: When you come across an error message pointing to either a READ or a DATA line you should check all the program's READs and DATAs carefully, no matter where the error message points.

Before we finally leave Program II, try changing line 60 to:

60 DATA 1,2,3,4

and run the program. As you see, it works perfectly. This is because the numbers in the data list are stored as string variables in *name$*.

However they are strings, not numerics. You can't do sums with them as you'll see if you now add:

35 name$=name$+1

and try to run the program.

You get :

Type mismatch in 35

message because you've tried to add a numeric, 1, to a string, *name$*.

This may seem a bit pedantic in the case of Program II, after all it's so simple no one could mistake a string for a number, and vice versa.

However, when you learn that data lists can contain both strings and numbers you'll see how easily the above mismatch problems can arise. Program III shows what I mean.

```
10 REM Program III
20 FOR family=1 TO 4
30 READ name$,age
40 PRINT name$,age
50 NEXT family
60 DATA Peter,34,Eileen,21
70 DATA Bodger,3,Spot,2
```

Program III

Here line 30 is reading values into two variables, the string variable *name$* and the numeric *age*. So as the loop cycles the program will look to the data line and expect a string, then a number, a string, then a number and so on.

The data lines have to be constructed so that this is what actually happens. If items are out of order in either the read line or the data list, havoc occurs. Try changing line 30 to:

30 READ age,name$

and you'll see the point. Notice that the error message points to the data list and not to where the error occurred.

It's not just the read line where you can get your variable types in a twist. Put back the original line 30 and type in:

60 DATA 34,Peter,21,Eileen

Now when you run the program you'll get the message:

Syntax error in 60

Be grateful — at least this one's pointinjg to the right place!

Try your own variants on Program III, mixing up string and numeric variables in the data lists. You'll soon get the hang of them.

Remember not to put too much trust in the error messages. Just because they say that there's a mistake in line 60 doesn't necessarily mean that line 60 is where you made your mistake.

```
10 REM Program IV
20 FOR boast=1 TO 3
30 READ rubbish$
40 PRINT rubbish$;
50 NEXT boast
60 DATA I,am,wonderful
```

Program IV

And talking of mistakes, have a look at Program IV, which is trying to use a data list to print the modest message!

I am wonderul

It's a nice program that you should have no problems understanding. The loop cycles three times, each time reading a string from the data list into *rubbish$*. It then prints out that string and the loop goes round and gets another.

It does, however, have one fault beside the inaccurate message. the strings are all jammed together to produce the message:

Iamwonderful

It would be nice to have some spaces there. Can we do it by putting spaces in front of the strings in the data list? Try it and see. You'll find that altering line 60 to:

60 DATA I, am, wonderful

still results in

Iamwonderful

This is because the Amstrad ignores the leading spaces.

What we have to do is to turn to the inverted commas we came across earlier. By using these around the strings we can get our spaces to be accepted. So line 60 become something like:

60 DATA I," am"," wonderful"

Now, suppose I'm flushed with success at getting my spaces into the message and want to repeat it five times. Could I put the whole thing in a FOR...NEXT loop that cycles five times and so get the required number of messages? Program V attempts this, but doesn't get too far.

All you get when you run the program is one:

I am wonderful

and a:

DATA exhausted in 40

```
10 REM Program V
20 FOR repeat=1 TO 5
30 FOR boast=1 TO 3
40 READ rubbish$
50 PRINT rubbish$;
60 NEXT boast
70 PRINT
80 NEXT repeat
90 DATA I," am"," wonderful
```

Program V

What's happened is that the program has run out of data items in the list. The first time round the outer loop there's no problem. The inner loop cycles three times, reading in values for *rubbish$* and printing them. Hence the first message.

Now the program sets off around the outer loop for the second time. It gets to the inner loop, which cycles three times for every cycle of the outer loop, and line 40 attempt to READ *rubbish$*. And this is where the program grinds to a halt.

What has happened is that it has run out of data items. The first three READs have taken what is in the list and moved the pointer along each time. Now it's pointing to nothing. There are only three strings in the list and you've had them. There are no more left for the program to read.

Of course what you want the micro to do is to go back to the beginning of the list again. The trouble is that you haven't told it that that's what you want.

What you have to do is to use a RESTORE command. This sets the data pointer back to the beginning of the data list. In other words, the READ commands starts taking items from the front of the list again.

In the case of Program V the line needed is:

65 RESTORE

which has the program producing the required five messages. Every time the inner loop finishes its three cycles the RESTORE sets the data pointer back to the first item in the data list, 1.

Now when the inner loop starts again it has the necessary data for the second message. When that loop finishes, the RESTORE restores the data list and the program carried on.

RESTORE doesn't have to be used by itself. It can be followed by a line number specifying which data line the data pointer is to go to. So in this case we could have had a line such as:

65 RESTORE 90

which sets the data pointer to the first item after the DATA of line 90. However, as there's only one data line in the program it's a bit pointless. Program VI shows a more practical use of a line number with RESTORE.

While it's hardly a wonderful maths test, the program does show how RESTORE can be used to choose between two different data lists. Line 100 ensures that if the user has made a mistake, the next set of questions are the easier set. It does this by using the RESTORE 130 to get the data pointer to select the easier numbers.

On the other hand, if the answers to the easy questions are correct, the RESTORE 140 of line 110 has the Amstrad READing from the harder set of numbers.

```
10 REM Program VI
20 WHILE eternity=0
30 easier=0
40 FOR loop=1 TO 3
50 READ number
60 PRINT "What is ";number;" times ";
number;
70 INPUT answer
80 IF answer<>number*number THEN PRIN
T "WRONG!": easier=-1
90 NEXT loop
100 IF easier=-1 THEN RESTORE 130
110 IF easier=0 THEN RESTORE 140
120 WEND
130 DATA 2,3,4
140 DATA 6,7,8
```

Program VI

Can you alter the program so that there are three or even more alternatives? I leave that up to you.

And that's where we come to an end for this month. We've seen how READ and DATA can be used to read in both string and numeric variables from data lists. We've also come across some of the pitfalls that await the unwary.

Finally we explored RESTORE and saw how it could be used with line numbers to select different data lists.

```
10 REM Program VII
20 DIM number(5)
30 FOR loop=1 TO 5
40 READ number(loop)
50 NEXT loop
60 FOR loop=1 TO 5
70 PRINT number(loop)
80 NEXT loop
90 DATA 100,200,300,400,500
```

Program VII

After all that I leave you with Program VII.

What's happening here? The FOR...NEXT loops and the READ and DATA are fairly familiar, but what's that DIM? The answer comes next month when we'll be dealing with arrays.

Big fun software

The entertainment software boom forecast in *Computing with the Amstrad* three months ago is now well underway.

Latest figures from the trade show that CPC programs are steadily gaining a larger share of the market while sales of games software for other machines like Spectrum and Commodore have been declining.

One survey says CPC games now account for more than 15 per cent of total sales, up from about 10 per cent last year.

Industry observers say the boom owes much to the 40 per cent cut in the price of 3in discs three months ago.

Cost of Amstrad's CF-2 blank discs was slashed from $12.50 to $7.50, and software houses told *Computing with the Amstrad* at the time that high disc price was one of the reasons CPC program sales were being held back.

Previously, publishers disappointed that software sales had not matched the success of the CPC itself were releasing tape-only games, and in some cases even ignoring the machine. "Yet our research shows a high proportion of CPC users prefer discs to tapes", said Patricia Mitchell, product manager of Virgin Games.

Amsoft marketing director John Ainsworth predicted: "It will stimulate an enormous demand for disc-based games, and those firms who've got their wits about them and can move fast will do very well indeed".

This has now been borne out by the sales figures released since January which show CPC products as the fastest-growing software sector.

It is seen as a reward for loyal CPC games producers such as Activision — which launched seven CPC games at once in March — Martech, Bubble Bus, Hewson, Ariolasoft, Alligata, Bug Byte and Melbourne House.

Good news, too, for software houses who have ensured that their blockbusters — like The Growing Pains of Adrian Mole from Virgin, Repton 3 from Superior, Monty Mole from Gremlin, Ballbreaker from CRL, and the PSS wargamers series — are released for the CPC.

IBM FADING IN PC BATTLE

Amstrad is pulling away from its biggest rival IBM in the fight for supremacy in the British business PC market.

Latest figures from market researchers Romtec put Amstrad's share up to a record 35.8 per cent which represents a 10 per cent lead over the former leader.

Before the turn of the year Amstrad was only 1.4 per cent ahead — the first time they had taken the lion's share of the market. But Romtec's survey reveals that only a small percentage of the Amstrad sales were taken up by PCWs.

But the survey was made among independent distributors rather than High Street chains, where most PCWs are sold.

Amstrad is keeping cool about the figures, a spokesman describing them as "very pleasing". It is thought they are waiting to see whether this is a trend.

Superior database for PC1512

One of the world's leading database houses has entered the Amstrad market for the first time with a revolutionary database for the PC1512.

Precision Software has unveiled Superbase Personal, an easy-to-use package for $250 designed specifically to make the most of the Gem environment.

This powerful system places no limit on the number of fields or on the size of the record, and as many files as needed can be opened to build up a relational report.

Superbase Personal has been developed as a "very visual" data management system. It offers a unique facility for coordinating pictures and text. The graphics management capability allows pictures to be retrieved and displayed, and there is a built-in automatic slide show picture sequence.

This multi-file relational database incorporates all the latest user-friendly techniques such as windows and pull-down menus. Data manipulation is achieved by the click of a mouse on video recorder style symbols on screen.

File structures can be changed at any time without disturbing existing records, and there is a specially designed straightforward printing feature.

Precision is well known on the international database scene, with sales of more than 100,000 packages world-wide. However this is its first product for the Amstrad range, having made its name primarily in the Commodore market.

"We are well aware of the importance of Amstrad in the micro scene today", said Nigel Lovett-Turner, Precision's sales director. "That is why we have devoted considerable time and money to develop this package".

Four in one for the CPC

A spreadsheet for the Amstrad CPC has been released by Audiogenic. The company claims Matrix is probably the most powerful spreadsheet for any home computer.

It is billed as four programs in one. Apart from the usual spreadsheet for displaying tabulated accounts, it has a database capability, a text editing notebook and a graph plotting facility.

Features include cut and paste sheet editing, pull-down menus, large spreadsheet area, adjustable column widths and many mathematical functions.

The notebook features a text editing area for preparing letters, reports and such and the facility to draw on data from the spreadsheet.

Financial information can also be presented in graphic forms such as line graph, bar graph or pie chart.

Price $75 on cassette and $87.50 on disc.

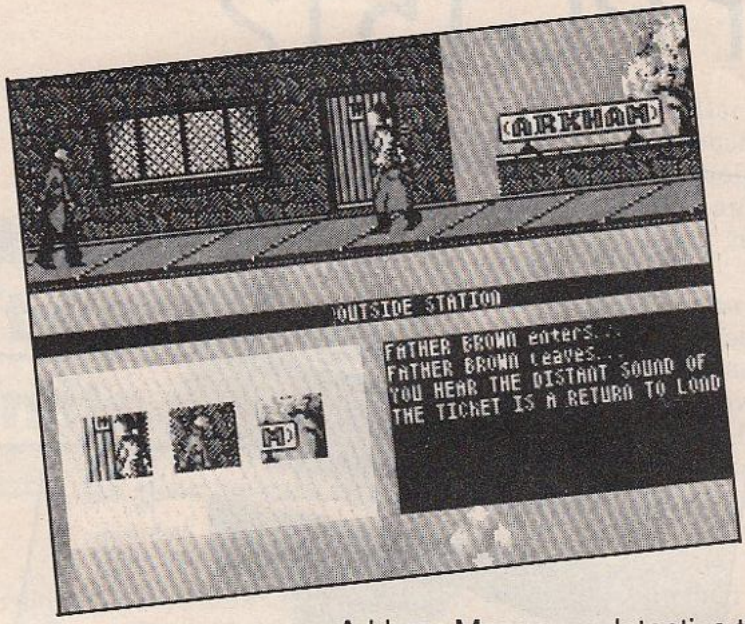
New games for CPC

Two new games for the CPC have been released by Melbourne House.

Doc the Destroyer is set in the future and players must try to save the Doomed City from flooding by penetrating the city's defences and confronting the temple priests who rule it.

The Mystery of Arkham Manor is a two-part detective thriller for the CPC. The player takes on the role of an investigative journalist and must try to discover the whereabouts of the missing Colonel Fortescue.

The village of Arkham must be explored and its inhabitants questioned. Regular telegrams and articles must be sent to the editor and these may be filed using a save and print option.



Arkham Manor . . . detective thriller

Both new games cost $22.50.

Assembler update

Arnor has finished work on a new version of its Maxam Assembler plus a C compiler for the CPC 6128 and the PCW.

Maxam 11 has a totally different editor, based on Arnor's Protext word processor.

A new monitor has also been written complete with symbiotic debugger. It enables switching through the various parts of the CP/M+ memory map.

The C compiler, which also runs on CP/M+ only, now allows external sources to be linked. Users can link with Maxam or vice-versa. Both cost $200.

Arnor is at present working on a version of Protext for the Amstrad PC.

DIXONS DRIVE INTO THE USA

High Street chain store Dixons is poised to expand into the US — and that could mean really good news for Amstrad.

The electrical stores group is a major retailer of Amstrad products, and its American venture could open up fresh markets for British micros.

Dixons is negotiating to buy Cyclops Corporation, owner of the coast-to-coast string of 120 Silo electrical appliance outlets.

Amstrad is hoping this means its micros will eventually appear on the shelves of every Silo store.

Amstrad already has an American presence through Sears chain stores and distributor Vidco, but a spokesman said: "We're after as much exposure as we can get.

"We hope our claims will be considered when Dixons completes the deal for Cyclops".

ROYAL FROLIC ON THE CPC

A light-hearted look at the Royal Family is the subject of the latest CPC release from 8th Day. In the hope of receiving a knighthood, HRH players must return the Queen's giro after an embarrassing mix-up at the DHSS.

The price is $7.50 on tape, $17.50

Discover some real characters!

In Part VI of our series on Amstrad graphics Geoff Turner and Micheal Noels show you how to design your own character set.

So far we've explored two ways of displaying information on the screen. We've used the text cursor for printing alphanumeric characters and the graphics cursor for drawing and plotting shapes.

Whichever method is used to write to the screen, the end result as far as the computer hardware is concerned is to change the colour of a number of points or pixels within the screen area.

Depending upon the particular combination of points selected, we hope to see a recognisable shape, whether a printed character such as the letter A or a shape drawn with the graphics cursor, like a triangle or circle.

In this article we are returning to the use of the text cursor, and we'll examine some of the many characters which can be printed on the screen. We'll also find out how these characters are produced.

Let's start off by resetting the computer. Now it's in command mode, and if we press any key then we expect that particular character to appear on the screen, positioned at the text cursor.

Press the letter A, and of course A will be displayed. Now if you look closely at the character it may be apparent that it is made up of a combination of dots. We could actually reproduce the letter A by using the PLOT command, as long as we can work out the correct combination of points to plot.

Program I demonstrates how the character may be plotted. Notice that we have printed A first at line 40, and then plotted a number of points to also produce the letter A.

Line 90 reads the X and Y coordinates from the DATA statements of lines 120 to 180. Notice that relative plotting is achieved by using the PLOTR command. The two characters produced are identical, but it was much easier to print the character rather than plot it.

Imagine having to plot every letter, number and punctuation mark! Even producing a short sentence would involve a considerable amount of work, and programming would be a tedious business.

Fortunately we don't need to plot any of these characters, as they can all be produced with the PRINT command. Buried somewhere deep in the Amstrad's firmware is all the information needed to generate the pattern of dots to represent all of our commonly-used characters, plus quite a few more. The work has been done for us.

The set of points required for each character is known as its bit pattern. The computer stores the bit pattern for each character and

```
10 REM PROGRAM I
20 MODE 1
30 LOCATE 1,5
40 PRINT"This is printed ";"A"
50 LOCATE 1,12
60 PRINT"This is plotted "
70 FOR pixel=1 TO 28
80 MOVE 256,208
90 READ x,y
100 PLOTR x,y
110 NEXT
120 DATA 2,2,4,2,10,2,12,2
130 DATA 2,4,4,4,10,4,12,4
140 DATA 2,6,4,6,6,6,8,6,10,6,12,6
150 DATA 2,8,4,8,10,8,12,8
160 DATA 2,10,4,10,10,10,12,10
170 DATA 4,12,6,12,8,12,10,12
180 DATA 6,14,8,14
```

Program I

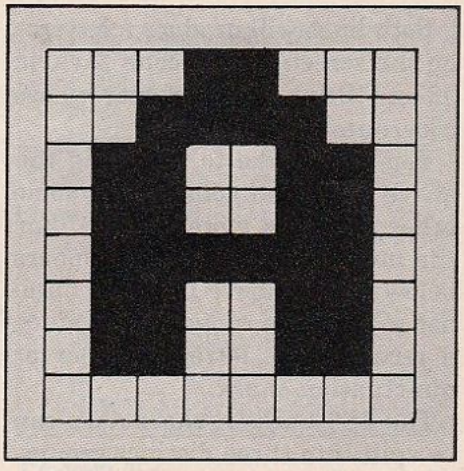


Figure I: Bit pattern for A

when it is asked to print a particular character it looks up the appropriate pattern and sends the data to the screen.

All this is performed by fast-working machine code routines, resulting in the character appearing even before we have had chance to release the key.

Each character is laid out or drawn on an 8 by 8 grid. Figure I illustrates how the letter A is produced. We used this particular bit pattern to plot the character in Program I.

The Amstrad has a capacity to store the information relating to 256 characters. If you wish to examine the bit pattern for any of them, they are all to be found in Appendix III of the CPC464 User Manual, and Chapter 7 of the CPC664 manual.

Each is given a number in the range 0 to 255 so that the computer may identify and print the correct character. For example, the letter A is referred to by the number 65 and we can use this code number to print the letter A.

So instead of entering:

PRINT "A"

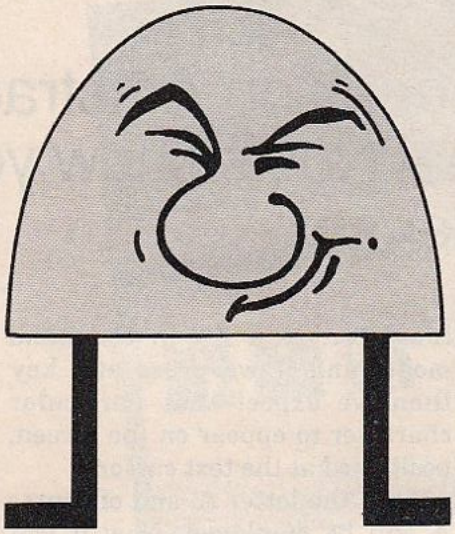
we can use this alternative:

PRINT CHR$(65)

Both lines will produce a letter A on the screen. The second says, in effect: "Print the character whose code number is 65".

You might like to try using the above statement substituting some other numbers instead of 65. For the moment avoid using numbers below 32 and above 126. Any number within this range will produce a recognisable letter, number or punctuation mark.

The 256 built-in characters can be split into three distinct groups. Those in the number range 0 to 31



are a special set of control characters. They are often referred to as non-printable characters, as they don't usually cause any character to appear on the screen.

A useful example of one of these control characters is the one with code number 7. Try entering:

PRINT CHR$(7)

You won't see the character, but you will hear it! This character causes a short beep to be generated via the computer's internal loudspeaker. Hence the term control characters – they control what the micro does.

```
10 REM PROGRAM II
20 MODE 1
30 x=1:y=1
40 FOR character =32 TO 126
50 LOCATE x,y
60 PEN 1
70 PRINT character;
80 PEN 2
90 PRINT CHR$(character)
100 y=y+1
110 IF y=25 THEN y=1 : x=x+6
120 NEXT
130 WHILE INKEY$="":WEND
```

Program II

There are quite a few other useful control characters. Numbers 8, 9, 10 and 11 control movement of the text cursor so that:

PRINT CHR$(10)

will move the text cursor down one line, while:

PRINT CHR$(11)

moves the cursor upwards one line.

Many of the control characters have equivalent Basic statements. For instance, character number 29 will set the border colour, although it will need to be followed by two other numbers to select the particular colour. The statement:

PRINT CHR$(29);CHR$(3);CHR$(3)

is exactly the same as the command:

BORDER 3

and will set the border to red. Notice that we need to enter two colour numbers even though they are the same. Different numbers would result in a flashing border.

Similarly character number 4 is equivalent to the MODE command, and the following statement will select Mode 2:

PRINT CHR$(4);CHR$(2)

We don't often need to use the control characters in Basic programming, but they can be useful when they are used as part of longer character strings, as we will see next month.

Although I said that characters 0

to 31 were non-printing ones, it is possible to print a symbol representing the character. To do this we use the Ctrl key in conjunction with one of the other keys. For example pressing Ctrl and G (the seventh letter of the alphabet) at the same time will print a symbol on the screen representing the bleeper.

You can examine all of these characters by holding down the Ctrl key and pressing each key A to Z in turn. Some of the symbols are quite clear in what they represent. The cursor movement characters 8 to 11 (Ctrl plus H to K), are represented by arrows showing the direction of cursor movement. However many of the symbols are a little bit obscure in relation to their meanings.

Notice that if we enter a command:

PRINT CHR$(7)

the control function is actually executed. In this case we hear the beep. If, however, we press Ctrl and G simultaneously the symbol is displayed but the function is not executed and no sound is heard.

We can use the Ctrl key to enter the control functions in program listings and get it to work, provided we surround it with quotes. Therefore instead of using a line like this:

10 PRINT CHR$(7)

we could achieve the same result by using this command:

10 PRINT "Q"

where Q is the symbol displayed when Ctrl and G are pressed together. Now when we run the program the control function will actually be performed when line 10 is executed, but the symbol will not appear on the screen.

The second group of characters are those numbered from 32 to 126. This group represents all the letters, numbers and punctuation marks which are found on the keyboard. Program II displays all these characters together with their respective code numbers.

You can also find the characters in the User Manual.

The interesting thing about these characters is that they are a standard set which can be found on most computers.

As we have already seen, if you instruct the computer to print character number 65 it will display the letter A. If you have access to any other type of computer you may like to try printing character number 65 on that one.

You should find it will also display the letter A, but the way it is built up may not be exactly identical. However it will be recognisable as the letter A. From this you might guess that there is a standard list of numbers that correspond to letters.

This set of characters is known as the Ascii set. Ascii is short for American Standard Code for Information Interchange. As its title implies, it is a code that originated in America. but it has been generally accepted as the standard code for all alphanumeric characters on most types of computer.

You might find slight variations with some of the lesser-used symbols and punctuation marks. Being an American code, there is no representation of the British pound sign within the standard set.

To overcome this problem some British manufacturers have inserted the pound sign within the standard character set in place of one of the lesser-used characters. However Amstrad have tackled this problem differently and allocated a number outside the standard range to the pound sign.

There is a Basic keyword to represent the Ascii code, so that if we wish to find the code number given to a particular character we can use the command:

PRINT ASC("A")

This will return the value 65, which we already know to be the code number of the letter A.

Program III is a variation on Program II, but this time it works the other way round. It prints out the Ascii value of each of the letters A to J. The letters are contained in the DATA statement in line 120.

If you wish the program to print out all of the Ascii codes, you would have to extend the DATA statement to include all the alphanumeric characters.

Alternatively you can use Program IV, which asks you to press a key and then displays the Asciii code number for that particular key. When you use this program, remember to take into account whether you have the Caps Lock key on or off. You can also make

```
10 REM PROGRAM III
20 MODE 1
30 FOR letter=1 TO 10
40 READ letter$
50 PEN 1
60 PRINT "The ASCII code for ";
70 PRINT letter$;
80 PRINT " is ";
90 PEN 2
100 PRINT ASC(letter$)
110 NEXT
120 DATA A,B,C,D,E,F,G,H,I,J
```

Program III

use of the Shift key to obtain the upper range of characters.

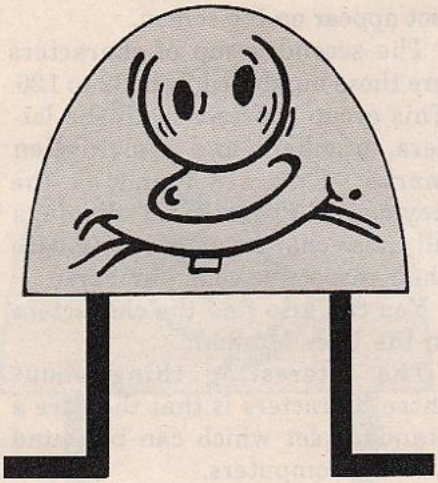
Finally we come to our third group of characters, those numbered from 127 to 255. These are a special set of symbols which are definitely non-standard. It's very unlikely that you would find an identical set of characters on any other computer.

You see, the designers of the Amstrad have taken full advantage of the fact that the micro can sotre a total of 256 characters, and have designed and interesting range of non-standard symbols for use in our programs.

Once again you may like to examine the User Manual to see all of these characters, or alternatively you can change line 40 of Program II to read:

40 FOR character=127 TO 255

You'll see all sorts of characters ranging from mathematical symbols to music symbols, playing-card symbols to little men. If you look carefully you may even spot the occasional Space Invader. (What micro would be complete



without one?)

Remember earlier we talked about the pound sign? This, too, is to be found within the range of special symbols as character number 163.

Many of these characters are not directly accessible from the keyboard, so to make them appear on screen it's usually necessary to use:

PRINT CHR$(n)

where n is the number of the character. Notice however that the pound sign can be accessed from the keyboard just to the left of the Clr key.

With all these characters to choose from, you would think that there would be every symbol we even need to use in our programs. However, sooner or later you'll find a use for one or more symbols that don't already exist in the built-in character set.

To overcome this problem, the programmers at Locomotive have given us the ability to design our own set of characters in addition to the internal set.

In fact, any of the built-in characters may be re-designed to something completely different.

Before we start to design our own characters we need to examine in detail how how any character is built up.

Remember that we said that each character is laid out on an 8 by 8 grid? Let's examine closely how the letter A is designed. We saw the layout in Figure I, and this is repeated in Figure II, but this time we have added some numbers to the drawing.

Across the top of the figure, each column is allocated a number. If you know anything about binary

```
10 REM PROGRAM IV
20 MODE 1
30 PRINT "Press a key - followed by E
nter"
40 PRINT
50 PEN 1
60 INPUT "key ";K$
70 PEN 3
80 PRINT "The ASCII code is ";ASC(K$)
90 GOTO 50
```

Program IV

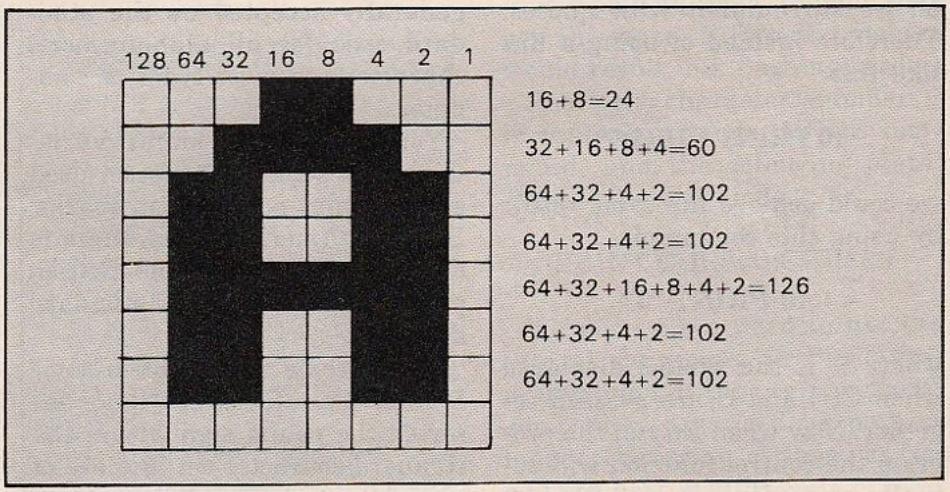


Figure II: Row values for A

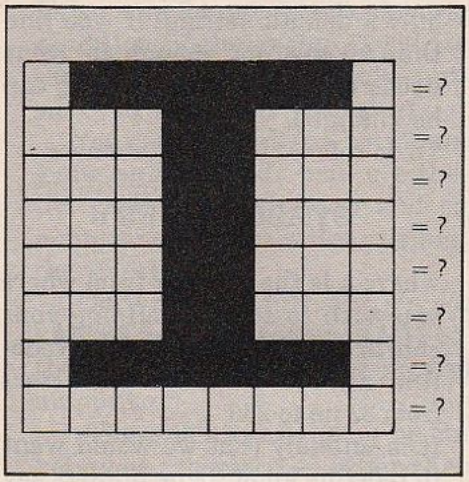


Figure III: A simple character

64+32+16+8+4+2	=126	01111110
16+8	= 24	00011000
16+8	= 24	00011000
16+8	= 24	00011000
16+8	= 24	00011000
16+8	= 24	00011000
64+32+16+8+4+2	=126	01111110
0+0+0+0+0+0+0+0	= 0	00000000

Table I: Bit pattern for fig.III

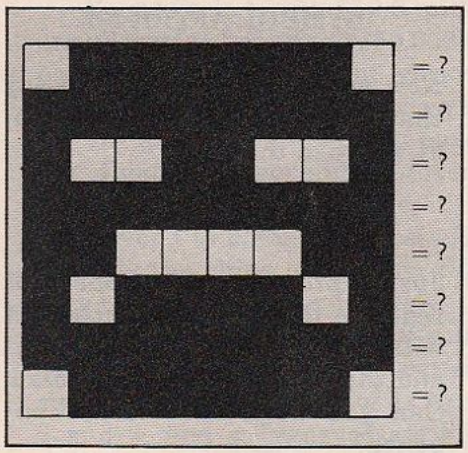


Figure IV: Alien Invaders

0+64+32+16+8+4+2+0=126	01111110
128+64+32+16+8+4+2+1=255	11111111
128+ 0+ 0+16+8+0+0+1=153	10011001
128+64+32+16+8+4+2+1=255	11111111
128+64+ 0+ 0+0+0+2+1=195	11000011
128+ 0+32+16+8+4+0+1=189	10111101
128+64+32+16+8+4+2+1=255	11111111
0+64+32+16+8+4+2+0=126	01111110

Table II: Bit pattern for Figure IV

numbers then you should recognise this sequence of numbers. Starting from the right-hand column with number 1, each column to the left has a number double the previous one. These numbers represent the eight bits of a binary number, and any combination of these numbers can represent a number from 0 to 255.

In the top row of Figure II, the squares in columns 16 and 18 are plotted. Now if we add these two column numbers together, we get 24. The number 24 represented in binary is:

00011000

You can see that the numbers 0 and 1 are in exactly the same positions as the white and black squares in the top row of Figure II. In other words the 0s and 1s of the binary number mirror the structure of the rows. A 1 means that that column is plotted, a 0 that is left untouched.

The same rule applies to each of the eight rows of Figure II. We can add up the column values of all the black squares in each row, and this gives us the number shown at the right hand side of the figure.

To see if you have understood this, you might like to try calculating the values of some other characters. Figure III (the letterI) is an easy one to start off with, but Figure IV (our Space Invader friend) is a little more complex.

You'll need to get the hang of calculating these values if you want to design your own characters. Tables I and II give the answers.

The Amstrad has already got the maximum 256 characters built in, so how can we accomadate any new characters? Well, the computer initially allows up to 16 characters to be redefined and these are numbered from 240 to 255. If we redesign character number 240 then the new symbol simply overwrights the existing symbol stored at number 240.

Let's try designing a new character now. First of all enter the command:

PRINT CHR$(240)

The micro will now display the original character numbered 240, which is in fact an upward-pointing arrow. To define a new character, we use the **SYMBOL** statement which is followed by the character number, which in turn is followed by eight numbers each represnting the value of each row of the character.

Try entering:

```
10 REM PROGRAM V
20 MODE 1
30 SYMBOL 240,0,0,0,0,0,0,0,0
40 FOR number=1 TO 8
50 LOCATE 1,number
60 INPUT"Number ";n(number)
70 SYMBOL 240,n(1),n(2),n(3),n(4),n(5
),n(6),n(7),n(8)
80 NEXT
90 LOCATE 20,5
100 PRINT CHR$(240)
110 LOCATE 1,20
120 INPUT "Another one  ";y$
130 IF UPPER$(y$)="Y" THEN GOTO 20
```

Program V

SYMBOL 240,255,255,255,
255,255,255,255,255

followed by:

PRINT CHR$(240)

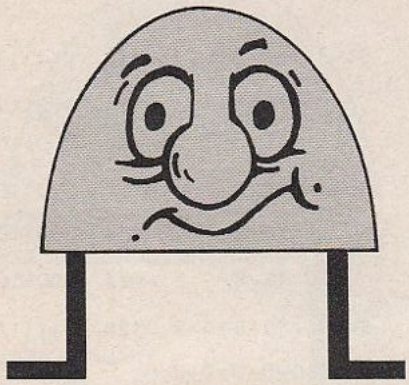
Now instead of the upward-pointing arrow we should have a solid square displayed. You see we have used the maximum value of 255 in each row of the character, which results in all the 64 of the 64 squares in the 8 by 8 grid being plotted.

You might like to try designing a character of your own now. Remember you may use any character numbering from 240 to 255.

Try following the SYMBOL command with eight completely random numbers after the character number, between 0 and 255. Alternatively try to work out a sequenceof numbers that will produce a recognisable symbol.

You may find a sheet of squared graph paper useful in helping you to design your own characters.

Program V allows you to enter eight veriable numbers which are then used to design a new character. In this program you will actually see each line of the symbol displayed as each number is entered.



The **SYMBOL** command worksequally well whichever screen mode you are using. Although Mode 0 actually displays the characters over a wider area than µode 1 and therefore uses more pixels, the symbol is still designed using the same numbering scheme.

Try changing line 20 in program V to Mode 0 and you will see that the program works just as well , but ofcoursethe characters appear wider than before.

As we mentioned earlier, the Amstrad will allow up to 16 new characters to be defined, but this is only a default setting. If you find that 16 are not enough for your needs then the default setting can be changed.

To do this we use the statement :

SYMBOL AFTER n

where *n* is the number from where you wish to start redefining your characters. We can, in fact, redefine all the prinable characters from 32 up to 255.

It is unlikely that we would want to redefine all of them, but lets say we needed 64 new characters. We would need to redefine character numbers 192 to 255. To do this we would use the command:

SYMBOL AFTER 192

Now we can start redefining any character from the 192 in the usual way like this:

SYMBOL 192,1,2,3,4,5,6,7,8

Just to prove it can be done we'll redefine the whole character set with program VI. Try running it and then try to use the keyboard. Can you work out what's happened?

Thats it for this month. As you seethen, redefining some of your characters can be very useful even if you only want to produce a more complex set of friendly aliens from space.

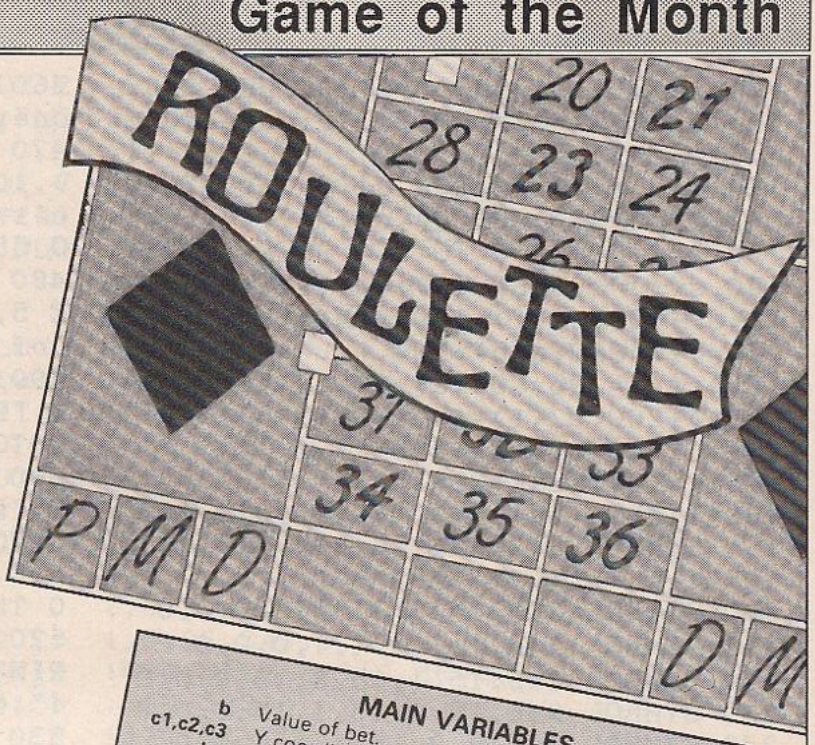
Designing a lot of new characters can be a tedious business which can be made easier by using a special utlity program called a character generator.

We'll make use of this in the next article, and we'll also see how we can build up larger characters by joining them together.

```
10 REM PROGRAM VI
20 MODE 1
25 PRINT "Wait for the beep !"
30 SYMBOL AFTER 32
40 FOR character =32 TO 255
50 SYMBOL character,255,255,255,255,2
55,255,255,255
60 NEXT
70 PRINT CHR$(7)
```

Program VI

By Richard Pearce



This game for one or two players is a faithful rendition of the casino favourite where fortunes may be won or lost — but in this version you won't lose your shirt.

You can place all the bets found in the real game and the odds paid are the same as in the casino. Anticipation and suspense are maintained as you watch the ball spin round the wheel.

Once past the introduction you choose the kind of bets you require (A-K on the menu), then follow the remaining prompts to set the amount.

There is no restriction on the number of bets you can place — other than your bank balance.

You can look at the wheel by pressing L and spin it using the M key. In the two player mode the N and O keys switch the bet-placing option between players.

Your aim is to win $12.5 million and you are allowed to bet up to $250,000 if your bank balance will stand it. The various bets and the odds paid on each are shown in Figure 1.

The game ends when you have either run out of money or, hopefully, broken the bank.

When in two player mode, it's a good idea for player 1 to place all his bets first, as occasionally the chips overlap on the table and may obscure the other player's.

Player 1's chips are white and those of player 2 are red, but in two particular sets of circumstances — the bet of a Carre using 0, 1, 2 and 3, and a Transversale Plein using 0 with 1 and 3 — a chequered chip is used for clarity.

If you are restricted to a green screen monitor your display will benefit from a change of INK 2 in line 320, from 9 to 18. Although this is not strictly the correct green for a roulette table you will be able to see the numbers more clearly.

Once Roulette's running you'll have hours of fun trying to break the bank. So, ladies and gentlemen, faites vos jeux ...

MAIN VARIABLES

b	Value of bet.
c1,c2,c3	Y coordinates of wheel.
d,e	Used when drawing table.
f	Subscript of c1,s1 and so on.
h	Set to 1 if bet wins.
k	Bank balance.
l	Number you have bet on.
ld,lm	Calculate chip coordinates.
m	Choice from menu.
n1,n2	X,Y coordinates of wheel numbers.
num	Number thrown.
pc	Chip colour.
pl	Player number.
plyr	Number of players.
px,py	Chip coordinates.
q	Amount of money left.
r	Odds of individual bets.
s1,s2,s3	X coordinates of wheel.
spin	If set to 1 ball will spin.
t	Number thrown on re-spin.
w	Reads data for letters on table.
x	Remembers bank balance at start of each round.
y	Set to 1 if you have an even bet and zero turns up.
z	Used to ensure correct position for ball to stop

Figure I:
The different bets and their odds

Name	Description	Odds
Plein	Any single number	35 – 1
Cheval	Any two adjacent numbers	17 – 1
Transversale plein	Horizontal row of three numbers	11 – 1
Carre	Square of four adjacent numbers	8 – 1
Sixaine	Two adjacent horizontal rows	5 – 1
Colonne	Column of 12 numbers	5 – 1
Douxaine	12 numbers 1–12, 13–24 or 25–36	2 – 1
Deux douxaines	24 numbers 1–24 or 13–36	1 – 2
Pass/Manque	18 numbers 1–18 or 19–36	Evens
Pair/Impair	Any odd/even number	Evens
Noire/Rouge	Any red/black number	Evens

```
10 REM              Roulette
20 REM         By Richard Pearce
30 REM (c) Computing with the Amstr
ad
40 REM ---------- CPC Only --------
--
50 MODE 1:DEG
60 INK 0,0:INK 1,24:INK 2,6:INK 3,1
1
70 DEFINT c-j,l-n,p,s-u,y-z
80 LOCATE 8,12:PRINT CHR$(18);:INPU
T;"How many players (1 or 2)";play$
90 IF play$="" THEN 80
100 plyr=ASC(play$)-48
```

```
110 IF plyr<1 OR plyr >2 THEN 80
120 LOCATE 8,15:PRINT "Please wait
a few seconds"
130 DIM s(36),c(36),s1(36),c1(36),s
2(36),c2(36),s3(36),c3(36)
140 f=0:FOR a=0 TO 360 STEP 360/37
150 s(f)=195*SIN(a)+200:s1(f)=160*S
IN(a)+200:s2(f)=140*SIN(a+5)+193:s3
(f)=175*SIN(a+5)+200
160 c(f)=195*COS(a)+200:c1(f)=160*C
OS(a)+200:c2(f)=140*COS(a+5)+207:c3
(f)=175*COS(a+5)+200
170 f=f+1:NEXT:CLS
180 SYMBOL 241,0,0,0,0,16,8,0,124
190 SYMBOL 242,12,124,204,118,0,0,0
,0
200 SYMBOL 243,15,9,9,9,15,0,0,0
210 SYMBOL 244,1,1,1,1,1,0,0,0
220 SYMBOL 245,15,1,15,8,15,0,0,0
230 SYMBOL 246,15,1,15,1,15,0,0,0
240 SYMBOL 247,9,9,15,1,1,0,0,0
250 SYMBOL 248,15,8,15,1,15,0,0,0
260 SYMBOL 249,15,8,15,9,15,0,0,0
270 SYMBOL 250,15,1,1,1,1,0,0,0
280 SYMBOL 251,15,9,15,9,15,0,0,0
290 SYMBOL 252,15,9,15,1,1,0,0,0
300 SYMBOL 253,0,56,124,124,124,56,
0,0
310 WINDOW#1,25,40,1,25:WINDOW#2,25
,40,11,25:WINDOW#3,25,40,19,25:WIND
OW#4,25,40,6,25:WINDOW#5,25,40,15,2
5:WINDOW#6,25,40,10,25:WIND
OW#7,25,40,25,25
320 INK 0,0:INK 1,6:INK 2,9:INK 3,2
6:INK 4,4
330 ENV 1,1,120,3,24,-5,3
340 ENV 2,1,120,2,1,-120,2
350 k(1)=500:k(2)=500:WHILE k(1)<50
00001 AND k(1)>0 AND k(2)<5000001 A
ND k(2)>0
360 q(1)=k(1):q(2)=k(2)
370 x(1)=k(1):x(2)=k(2):y=0
380 spin=0:num=RND*37
390 GOSUB 590:GOSUB 1470
400 GOSUB 870
410 SOUND 1,260,10,0,2:SOUND 1,195,
10,0,2:SOUND 1,159,10,0,2:SOUND 1,1
19,10,0,2
420 GOSUB 1150
430 CLS:WEND
440 IF plyr=2 THEN 480
450 IF k(1)=0 THEN 460 ELSE LOCATE
9,12:PRINT "You have broken the ban
k!!":LOCATE 14,14:PRINT "Feeling fl
ush?":LOCATE 9,15:PRINT "Ho
w about lending me ten":LOCATE 11,1
6:PRINT" dollars till Tuesday?":GOT
O 550
460 LOCATE 9,12:PRINT"You have no m
oney left!"
470 FOR i=1 TO 13:SOUND 1,676+(i*13
),10,0,2:NEXT:LOCATE 5,14:PRINT "Do
n't plan any spending sprees !":GOT
O 550
480 IF k(1)=0 AND k(2)=0 THEN LOCAT
E 5,12:PRINT "You have both run out
 of money!":GOTO 550
490 IF k(1)=0 THEN LOCATE 6,12:PRIN
T "Player 1 has run out of money!":
GOTO 550
500 IF k(2)=0 THEN LOCATE 6,12:PRIN
T "Player 2 has run out of money!":
GOTO 550
510 IF k(1)>5000000 AND k(2)>500000
0 THEN 540
520 IF k(1)>k(2) THEN LOCATE 6,12:P
RINT "Player 1 has broken the bank!
!":GOTO 550
530 LOCATE 6,12:PRINT "Player 2 has
 broken the bank!!":GOTO 550
540 LOCATE 5,12:PRINT "You have bot
h broken the bank !!!":GOTO 550
550 PEN 1:LOCATE 8,24:PRINT "Do you
 want another game?":v$=INKEY$
560 IF v$="" THEN 550 ELSE v$=UPPER
$(v$)
570 IF v$="Y" THEN 350
580 IF v$<>"N" THEN 550 ELSE END
590 REM ****** draw table *********
*
600 CLG (2)
610 FOR i=0 TO 3:MOVE 16,366-(i*108
):DRAW 352,366-(i*108),3:NEXT
620 FOR i=0 TO 1:MOVE 16+(i*366),10
:DRAW 16+(i*336),366:NEXT
630 FOR i=0 TO 12:MOVE 112,366-(i*2
7):DRAW 256,366-(i*27):NEXT
640 MOVE 16,10:DRAW 352,10
650 FOR i=0 TO 1:FOR j=0 TO 1:MOVE
48+(i*32)+(j*240),42:DRAW 48+(i*32)
+(j*240),10:NEXT:NEXT
660 FOR i=0 TO 1:MOVE 112+(i*144),1
0:DRAW 112+(i*144),390,3:NEXT
670 FOR i=0 TO 1:MOVE 160+(i*48),10
:DRAW 160+(i*48),366:NEXT
680 TAG
690 MOVE 112,390:DRAW 256,390:PLOT
176,384,0:PRINT "0";
700 d=1:RESTORE 790
710 FOR i=0 TO 2:FOR j=0 TO 2:d$=ST
R$(d):READ e:PLOT 128+(j*48),358-(i
*27),e:d$=MID$(d$,2):PRINT d$;:d=d+
1:NEXT:NEXT
720 FOR i=3 TO 11:FOR j=0 TO 2:d$=S
TR$(d):READ e:PLOT 120+(j*48),358-(
i*27),e:d$=MID$(d$,2):PRINT d$;:d=d
+1:NEXT:NEXT
```

```
730 RESTORE 800:FOR i=0 TO 2:PLOT 2
6+(i*32),32,0:READ b$:PRINT b$;:PLO
T 330-(i*32),32:PRINT b$;:NEXT
740 RESTORE 810:FOR i=0 TO 5:PLOT 7
2,358-(i*16),0:READ w:PRINT CHR$(w)
;:NEXT:RESTORE 820:FOR i=0 TO 4:PLO
T 280,350-(i*16):READ w:PRI
NT CHR$(w);:NEXT
750 RESTORE 830:FOR i=7 TO 12:PLOT
40,360-(i*16):READ w:PRINT CHR$(w);
:PLOT 312,360-(i*16):READ w:PRINT C
HR$(w);:NEXT
760 RESTORE 840:FOR i=0 TO 4:PLOT 4
0,352-(i*16):READ w:PRINT CHR$(w);:
NEXT:RESTORE 850:FOR i=0 TO 5:PLOT
312,356-(i*16):READ w:PRINT
 CHR$(w);:NEXT
770 FOR j=0 TO 1:RESTORE 860:FOR i=
0 TO 3:READ h1,h2,h3,h4:PLOT 32+(j*
240),126-(i*16),j:PRINT CHR$(h1);CH
R$(h2);CHR$(h3);CHR$(h4);:N
EXT:NEXT
780 TAGOFF:PEN 3:RETURN
790 DATA 1,0,1,0,1,0,1,0,1,0,0,1,0,
1,0,1,0,1,1,0,1,0,1,0,1,0,1,0,0,1,0
,1,0,1,0,1
800 DATA p,m,d
810 DATA 49,57,241,242,51,54
820 DATA 49,241,242,49,56
830 DATA 73,32,77,80,80,65,65,73,73
,82,82,32
840 DATA 80,65,83,83,69
850 DATA 77,65,78,81,85,69
860 DATA 32,214,215,32,214,143,143,
215,213,143,143,212,32,213,212,32
870 REM ********** draw wheel *****
****
880 CLG 2:TAG
890 RESTORE 1030:FOR i=1 TO 64:READ
 n,n1,n2,n3:PLOT n1,n2,n3:PRINT CHR
$(n+243);:NEXT
900 TAGOFF
910 MOVE 200,200
920 FOR f=0 TO 36
930 DRAW s(f),c(f),3
940 IF f=36 THEN f=-1
950 DRAW s(f+1),c(f+1)
960 MOVE 200,200
970 IF f=-1 THEN f=36
980 NEXT
990 MOVE s1(36),c1(36)
1000 FOR f=0 TO 36
1010 DRAW s1(f),c1(f)
1020 NEXT
1030 DATA 0,174,382,3,2,212,382,1,3
,198,382,1,5,240,378,0,1,228,378,0,
9,270,370,1,1,256,364,1,4,290,352,0
,1,320,332,1,2,304,332,1,2,
332,310,0,5,356,284,1,2,340,284,1,7
,368,256,0,1,352,256,0,4,372,226,1,
3,356,226,1,6,364,196,0,7,372,166,1
,2,356,166,1
1040 DATA 3,362,136,0,1,346,140,0,6
,346,106,1,3,332,114,1,1,330,86,0,1
,314,92,0,0,308,64,1,3,294,68,1,8,2
78,50,0,3,258,36,1,2,244,40
,1,0,226,24,0,1,214,36,0,5,190,28,1
,4,164,36,0,2,152,26,0,6,138,44,1,1
,122,34,1,3,110,54,0,3,96,48,0,1,76
,66,1
1050 DATA 0,62,86,0,2,50,86,0,4,46,
112,1,1,30,112,1,1,34,138,0,3,18,13
8,0,9,18,164,1,2,22,196,0,2,6,196,0
,8,22,226,1,1,6,226,1,9,30,
254,0,2,14,254,0,7,32,282,1,8,56,31
2,0,2,42,312,0,2,74,332,1,1,60,336,
1,5,96,346,0,3,82,352,0
1060 DATA 3,116,364,1,6,152,372,0,2
,138,380,0
1070 RETURN
1080 REM**********look at wheel
1090 GOSUB 870
1100 TAG
1110 MOVE 370,50:PRINT"Press space
to";:MOVE 370,25:PRINT"return to me
nu";
1120 IF INKEY(47)<>0 THEN 1120
1130 GOSUB 590:GOSUB 1680
1140 TAGOFF:RETURN
1150 REM***** spin sequence********
****
1160 GOSUB 1370
1170 IF num>0 OR num=0 AND y=0 THEN
 1240
1180 num=t:TAG
1190 MOVE 370,50:PRINT"Press space
to";:MOVE 420,25:PRINT"respin";
1200 IF INKEY(47)<>0 THEN 1200
1210 PLOT s2(f),c2(f),2:PRINT CHR$(
253);
1220 MOVE 370,50:PRINT SPACE$(14);:
MOVE 420,25:PRINT SPACE$(6);
1230 GOSUB 1370
1240 GOSUB 1260
1250 TAGOFF:RETURN
1260 REM***** win or lose *********
***
1270 TAG:IF plyr=1 THEN 1310
1280 IF x(1)>k(1) THEN MOVE 450,375
:PRINT "Player 1";:MOVE 450,350:PRI
NT "has lost";:MOVE 450,325:PRINT "
$";x(1)-k(1);:ELSE MOVE 450
,375:PRINT "Player 1";:MOVE 450,350
:PRINT "has won";:MOVE 450,325:PRIN
T "$";k(1)-x(1);
1290 IF x(2)>k(2) THEN MOVE 450,250
:PRINT "Player 2";:MOVE 450,225:PRI
NT "has lost";:MOVE 450,200:PRINT "
```

```
$";x(2)-k(2);:ELSE MOVE 450
,250:PRINT "Player 2";:MOVE 450,225
:PRINT "has won";:MOVE 450,200:PRIN
T "$";k(2)-x(2);
1300 GOTO 1320
1310 IF x(1)>k(1) THEN MOVE 400,375
:PRINT"You have lost";:MOVE 425,340
:PRINT "$";x(1)-k(1);:ELSE MOVE 400
,375:PRINT "You have won";:
MOVE 425,340:PRINT "$";k(1)-x(1);
1320 MOVE 378,100:PRINT "Winning no
.:-";num;
1330 PLOT 400,50,0:PRINT "Press spa
ce to";:MOVE 450,25:PRINT"continue"
;
1340 TAGOFF
1350 IF INKEY(47)<>0 THEN 1350
1360 RETURN
1370 REM****** spin ball **********
*
1380 TAG
1390 sp=RND*8:IF sp<3 THEN sp=3
1400 FOR i=1 TO sp:FOR f=0 TO 36:PL
OT s2(f),c2(f),3:PRINT CHR$(253);:P
LOT s2(f),c2(f),2:PRINT CHR$(253);:
NEXT:NEXT
1410 RESTORE 1450
1420 p=37:z=-2:WHILE p<>num:READ p:
z=z+1:WEND
1430 FOR f=0 TO z:PLOT s2(f),c2(f),
3:PRINT CHR$(253);:PLOT s2(f),c2(f)
,2:PRINT CHR$(253);:NEXT
1440 PLOT s2(f),c2(f),3:PRINT CHR$(
253);
1450 DATA 32,15,19,4,21,2,25,17,34,
6,27,13,36,11,30,8,23,10,5,24,16,33
,1,20,14,31,9,22,18,29,7,28,12,35,3
,26,0
1460 TAGOFF:RETURN
1470 REM ***** menu ************
1480 ORIGIN 0,0,1000,2000,3000,4000
:CLG 0:ORIGIN 0,0,1,640,1,400
1490 GOSUB 1680:pl=1
1500 IF q(1)=0 AND plyr=1 THEN RETU
RN
1510 CLS#3:IF play=13 THEN 1560
1520 LOCATE#3,2,2:PRINT#3,"Player";
pl;"to bet"
1530 LOCATE#3,2,4:PRINT#3,"pl 1 $";
q(1)
1540 LOCATE#3,2,5:PRINT#3,"pl 2 $";
q(2)
1550 LOCATE#7,2,1:INPUT#7,"choice :
",m$:IF m$="" THEN 1550 ELSE 1580
1560 LOCATE#3,2,2:PRINT#3,"Money re
maining    $";q(1)
1570 LOCATE#3,2,5:INPUT#3,"choice :
",m$:IF m$="" THEN 1570
1580 m=ASC(UPPER$(m$))-64
1590 IF m<1 OR m>play THEN 1510
1600 TAG
1610 ON m GOSUB 1920,2010,2190,2370
,2510,2580,2660,2730,2810,2960,3110
,1080,1640,1660,1660
1620 TAGOFF
1630 IF spin=1 THEN RETURN ELSE 150
0
1640 REM ***** allow spin *********
1650 spin=1:RETURN
1660 REM******* change player *****
***
1670 pl=m-13:RETURN
1680 REM ********** list **********
***
1690 CLS#1:PEN#1,2:LOCATE#1,2,1:PRI
NT#1,"Choose your bet":PEN#1,1:LOCA
TE#1,1,3:PRINT#1,"key   bet"
1700 IF plyr=1 THEN play=13 ELSE pl
ay=15
1710 PEN#1,3:FOR i=1 TO play:LOCATE
#1,2,(i+3):PRINT#1,CHR$(i+64):NEXT
1720 RESTORE 1740:FOR i=1 TO play:R
EAD g$:LOCATE#1,4,(i+3):PRINT#1,g$:
NEXT
1730 RETURN
1740 DATA single number,two numbers
,horiz. row,square,two rows,column,
dozen,two dozen,high/low,even/odd,b
lack/red,see wheel,spin bal
l,player 1 bet,player 2 bet
1750 REM ******* bet **************
*
1760 CLS#4:PEN#4,3
1770 LOCATE#4,2,13:PRINT#4,"Money r
emaining     $";q(pl)
1780 LOCATE#4,1,1:INPUT#4,"How much
 do you want to bet ?       $",b
1790 IF b>100000 OR q(pl)-b<0 GOTO
1750
1800 RETURN
1810 REM ******* choose number ****
****
1820 CLS#2:l=36
1830 LOCATE#2,1,1:INPUT#2,"What is
the     lowest number   your bet is
 to   cover ";l
1840 IF l<0 OR l>35 GOTO 1820
1850 GOSUB 1870
1860 h=0:RETURN
1870 REM ******* ld/lm ************
1880 lm=l MOD 3:IF lm=0 THEN lm=3
1890 ld=FIX(l/3):IF lm=3 THEN ld=ld
-1
1900 RETURN
1910 REM ******* bets *************
1920 l=37:h=0
1930 CLS#1:PEN#1,1:LOCATE#1,5,1:PRI
NT#1,"en plein":LOCATE#1,2,2:PRINT#
```

```
1,"single number"
1940 GOSUB 1750:PEN#2,3
1950 WHILE l>36:CLS#2:LOCATE#2,1,1:
INPUT#2,"what number";l:WEND
1960 IF l=0 THEN px=128:py=385:GOTO
 1980
1970 GOSUB 1870:px=80+(lm*48):py=35
8-(ld*27)
1980 GOSUB 3270
1990 q(pl)=q(pl)-b:IF l=num THEN h=
1
2000 r=35:GOSUB 3310:RETURN
2010 CLS#1:PEN#1,1:LOCATE#1,5,1:PRI
NT#1,"a cheval":LOCATE#1,4,2:PRINT#
1,"two numbers"
2020 hi=40
2030 GOSUB 1750
2040 GOSUB 1820
2050 GOSUB 1870
2060 IF l=0 THEN 2090
2070 WHILE hi<l+1 OR hi=l+2 OR hi>l
+3:CLS#3:LOCATE#3,1,1:INPUT#3,"What
 is the second number ";hi
2080 WEND:GOTO 2120

2090 WHILE hi<1 OR hi>3:CLS#3:LOCAT
E#3,1,1:INPUT#3,"What is the    sec
ond number";hi:WEND
2100 px=70+(hi*48):py=375
2110 GOTO 2160
2120 IF l MOD 3=0 AND hi=l+1 THEN 2
040
2130 IF hi=l+3 THEN 2150
2140 px=104+(lm*48):py=360-(ld*27):
GOTO 2160
2150 px=82+(lm*48):py=347-(ld*27)
2160 GOSUB 3270
2170 q(pl)=q(pl)-b:IF l=num OR hi=n
um THEN h=1
2180 r=17:GOSUB 3310:RETURN
2190 CLS#1:PEN#1,1:LOCATE#1,1,1:PRI
NT#1," transversale       plein":LO
CATE#1,1,3:PRINT#1,"horizontal row"
2200 GOSUB 1750
2210 GOSUB 1820
2220 IF l=0 THEN 2290
2230 IF l MOD 3<>1 OR l>34 THEN 221
0
2240 GOSUB 1870
2250 px=104+((pl-1)*144):py=360-(ld
*27)
2260 GOSUB 3270
2270 q(pl)=q(pl)-b:IF num>l-1 AND n
um<l+3 THEN h=1
2280 r=11:GOSUB 3310:RETURN
2290 l1=0:WHILE l1=0 OR l1>2:CLS#5:
LOCATE#5,1,2:INPUT#5,"What is the
  second  number";l1:WEND
2300 l2=0:WHILE l2>3 OR l2<l1+1:CLS
#3:LOCATE#3,1,2:INPUT#3,"What is th
e    third number";l2:WEND
2310 IF pl=1 THEN pc=3 ELSE pc=1
2320 IF l2=l1+2 THEN 2340
2330 px=104+(l1*48):py=373:GOSUB 32
70:GOTO 2350
2340 PLOT 142,373,pc:PRINT CHR$(206
);:PLOT 238,373:PRINT CHR$(206);
2350 q(pl)=q(pl)-b:IF num=l OR num=
l1 OR num=l2 THEN h=1
2360 r=11:GOSUB 3310:RETURN
2370 CLS#1:PEN#1,1:LOCATE#1,6,1:PRI
NT#1,"carre":LOCATE#1,2,2:PRINT#1,"
block of four"
2380 GOSUB 1750
2390 GOSUB 1820
2400 IF l=0 THEN 2470
2410 IF l MOD 3=0 OR l>32 THEN 2390
2420 GOSUB 1870
2430 px=104+(lm*48):py=345-(ld*27)
2440 GOSUB 3270
2450 q(pl)=q(pl)-b:IF num=l OR num=
l+1 OR num=l+3 OR num=l+4 THEN h=1
2460 r=8:GOSUB 3310:RETURN
2470 IF pl=1 THEN pc=3 ELSE pc=1
2480 PLOT 238,387,pc,:PRINT CHR$(20
6);
2490 q(pl)=q(pl)-b:IF num<4 THEN h=
1
2500 r=8:GOSUB 3310:RETURN
2510 CLS#1:PEN#1,1:LOCATE#1,5,1:PRI
NT#1,"sixaine":LOCATE#1,1,2:PRINT#1
,"two horiz. rows"
2520 GOSUB 1750
2530 GOSUB 1820:IF l MOD 3<>1 THEN
2530
2540 GOSUB 1870:px=104+((pl-1)*144)
:py=345-(ld*27)
2550 GOSUB 3270
2560 q(pl)=q(pl)-b:IF num>l-1 AND n
um<l+6 THEN h=1
2570 r=5:GOSUB 3310:RETURN
2580 CLS#1:PEN#1,1:LOCATE#1,5,1:PRI
NT#1,"colonne":LOCATE#1,1,2:PRINT#1
,"vertical column"
2590 GOSUB 1750
2600 GOSUB 1820:IF l<1 OR l>3 THEN
2600
2610 GOSUB 1870:px=116+((lm-1)*48)+
((pl-1)*24):py=34
2620 GOSUB 3270
2630 q(pl)=q(pl)-b:numb=num MOD 3:I
F numb=0 THEN numb=3
2640 IF l=numb THEN h=1
2650 r=2:GOSUB 3310:RETURN
2660 CLS#1:PEN#1,1:LOCATE#1,5,1:PRI
NT#1,"douxaine":LOCATE#1,6,2:PRINT#
1,"dozen"
2670 GOSUB 1750
```

```
2680 GOSUB 1820:IF l MOD 12<>1 OR l
>25 THEN 2680
2690 px=26+(FIX(l/12)*32)+((pl-1)*2
40):py=34
2700 GOSUB 3270
2710 q(pl)=q(pl)-b:IF num>l-1 AND n
um<l+12 THEN h=1
2720 r=2:GOSUB 3310:RETURN
2730 CLS#1:PEN#1,1:LOCATE#1,2,1:PRI
NT#1,"deux douxaines":LOCATE#1,3,2:
PRINT#1,"two dozen"
2740 GOSUB 1750
2750 IF b MOD 2=1 THEN b=b-1
2760 GOSUB 1820:IF l MOD 12<>1 OR l
>13 THEN 2760
2770 px=42+(FIX(l/12)*32)+((pl-1)*2
40):py=34
2780 GOSUB 3270
2790 q(pl)=q(pl)-b:IF num>l-1 AND n
um<l+24 THEN h=1
2800 r=0.5:GOSUB 3310:RETURN
2810 CLS#1:PEN#1,1:LOCATE#1,3,1:PRI
NT#1,"passe/manque":LOCATE#1,4,2:PR
INT#1,"high/low"
2820 GOSUB 1750
2830 CLS#6:PEN#6,3
2840 LOCATE#6,1,1:INPUT#6,"Press ke
y for    choice (p/m)",c$
2850 IF c$="" THEN 2840
2860 c=ASC(UPPER$(c$))
2870 IF c=77 OR c=80 THEN 2880 ELSE
 2830
2880 py=350-((pl-1)*65):IF c=80 THE
N px=20 ELSE px=334
2890 GOSUB 3270
2900 IF num=0 THEN 2930
2910 h=0:q(pl)=q(pl)-b:IF num<19 AN
D c=77 OR num>18 AND c=80 THEN h=1
2920 r=1:GOSUB 3310:RETURN
2930 GOSUB 3350:IF t=0 THEN RETURN
2940 h=0:q(pl)=q(pl)-b:IF t<19 AND
c=77 OR t>18 AND c=80 THEN h=1
2950 r=0:GOSUB 3310:RETURN
2960 CLS#1:PEN#1,1:LOCATE#1,2,1:PRI
NT#1,"pair/impair":LOCATE#1,3,2:PRI
NT#1,"even/odd"
2970 GOSUB 1750
2980 CLS#6:PEN#6,3
2990 LOCATE#6,1,6:INPUT#6,"press ke
y for choice (p/i)",c$
3000 IF c$="" THEN 2990
3010 c=ASC(UPPER$(c$))
3020 IF c=73 OR c=80 THEN 3030 ELSE
 2980
3030 py=246-((pl-1)*65):IF c=73 THE
N px=20 ELSE px=334
3040 GOSUB 3270
3050 IF num=0 GOTO 3080
3060 h=0:q(pl)=q(pl)-b:IF c MOD 2=n
um MOD 2 THEN h=1
3070 r=1:GOSUB 3310:RETURN
3080 GOSUB 3350:IF t=0 THEN RETURN
3090 h-0:q(pl)=q(pl)-b:IF c MOD 2=t
 MOD 2 THEN h=1
3100 r=0:GOSUB 3310:RETURN
3110 CLS#1:PEN#1,1:LOCATE#1,4,1:PRI
NT#1,"noir/rouge":LOCATE#1,4,2:PRIN
T#1,"black/red"
3120 GOSUB 1750
3130 CLS#6:PEN#6,3
3140 LOCATE#6,1,6:INPUT#6,"press ke
y for   choice (n/r)",c$
3150 IF c$="" THEN 3140
3160 c=ASC(UPPER$(c$))
3170 IF c=78 OR c=82 THEN 3180 ELSE
 3130
3180 py=138-((pl-1)*65):IF c=78 THE
N px=20 ELSE px=334
3190 GOSUB 3270
3200 IF num=0 GOTO 3230
3210 h=0:q(pl)=q(pl)-b:RESTORE 3260
:FOR i=1 TO num:READ k:NEXT:IF k=0
AND c=78 OR k=1 AND c=82 THEN h=1
3220 r=1:GOSUB 3310:RETURN
3230 GOSUB 3350:IF t=0 THEN RETURN
3240 h=0:q(pl)=q(pl)-b:RESTORE 3260
:FOR i=1 TO t:READ k:NEXT:IF k=0 AN
D c=78 OR k=1 AND c=82 THEN h=1
3250 r=0:GOSUB 3310:RETURN
3260 DATA 1,0,1,0,1,0,1,0,1,0,0,1,0
,1,0,1,0,1,1,0,1,0,1,0,1,0,1,0,0,1,
0,1,0,1,0,1
3270 REM ****** draw chip *********
3280 IF pl=1 THEN pc=3 ELSE pc=1
3290 PLOT px,py,pc:PRINT CHR$(233);
3300 RETURN
3310 REM********* win ************
*
3320 IF h=1 THEN k(pl)=k(pl)+(r*b)
ELSE k(pl)=k(pl)-b
3330 GOSUB 1680:RETURN
3340 REM ********* respin *********
**
3350 IF y=1 THEN RETURN
3360 t=RND*37:IF t=0 THEN k(pl)=k(p
l)-b:q(pl)=q(pl)-b:GOSUB 1680
3370 y=1:RETURN
3380 REM******** press space ******
****
3390 PEN 2:LOCATE 9,23:PRINT"Press
space to continue"
3400 IF INKEY(47)<>0 THEN 3390
3410 CLS:PEN 3:LOCATE 17,2:PRINT"Ro
ulette"
3420 RETURN
```

BALLBREAKER

CRL $19.95 cass, $42.50 disc, keys and joystick

In the dim and distant past, when home computers were a figment of Clive Sinclair's imagination, there existed a state-of-the-art video game by the name of *Breakout*. A variation on the ping pong theme, you used a ball to chip away at a brick wall until it completely disappeared.

Ballbreaker, the latest hi-tech offering from CRL, is an up-to-the-minute version of *Breakout* which employs all the wonderful facilities that are at the disposal of today's programmers.

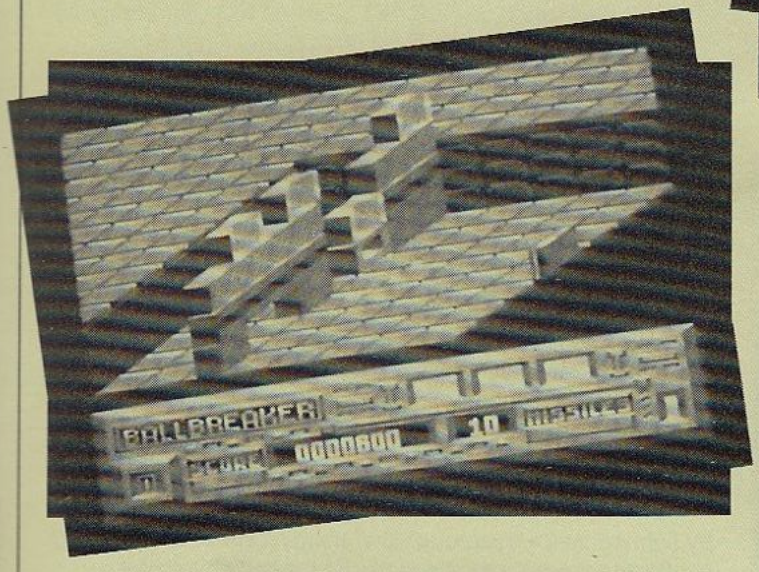
The first thing that you notice about *Ballbreaker* is that everything is drawn in 3D in glorious technicolour.

The game is played in the Ballbreaker arena, which has three solid walls and a fourth side which you must defend with a bat.

Unlike the original game the wall is vertical: As the ball hits a brick at the base of the wall it disintegrates, which causes a landslide from above as the bricks fall down to fill the gap.

The bricks aren't all identical in shape so you may be left with a rectangle resting on a square, which provides a gap through which you can hit the ball.

Once on the far side of the wall the ball bounces repeatedly against the bricks, wreaking havoc with the multi-coloured masonry. You can take a breather at this point and await the ball's return.



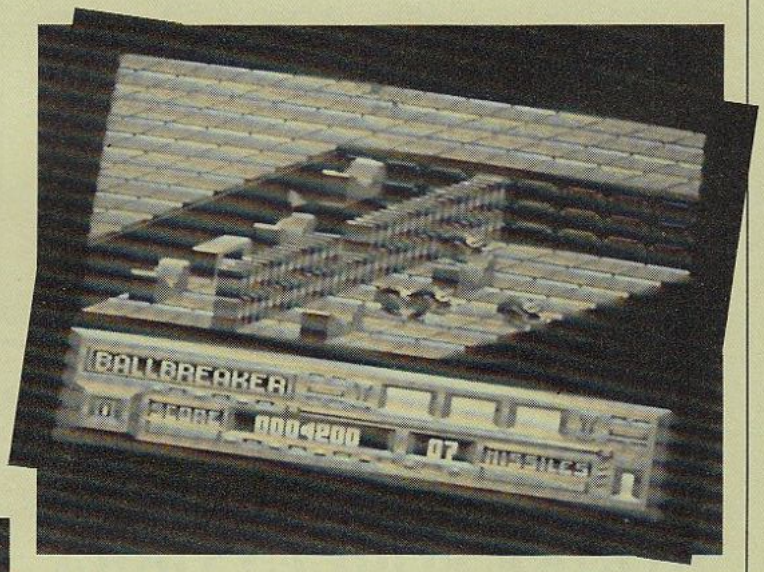
Some of the blocks are concave, which results in the occasional awkward bounce. On the plus side you now have a fire button and a stock of 10 missiles — most useful for taking out that final elusive block.

Screen one is a straight-forward brick wall (no problems there), but screen two is a different can of worms. Apart from the ordinary stretch of wall running across the middle of the arena there are two sets of columns, in front of and behind it.

The very fact that there are now three layers of wall means that the front columns are much closer to your bat. A ball rebounding from one of these requires a lightning fast response if you are to return it.

Being a nasty bunch of chaps the CRL programmers didn't think this was tricky enough so they placed purple frogs on top of the front two columns. The instant that a column is destroyed, the frog leaps off and runs at you.

Avoiding the frog and letting it fall to its death will not help — the dratted things are reincarnated immediately in their original starting positions, allowing them to attack again and again.



Your only course of action is to hit the frog with the ball or a missile, but due to the close proximity of the wall this is easier said than done.

If you set yourself up to blast the frog with a missile there is a very good chance that you won't have time to return the ball. Concentrating on returning the ball will invariably result in you being rammed by a charging bullfrog ... nasty!

There are 35 different screens, all containing devious problems and puzzles. Once you have destroyed the wall on the first screen you progress to the next by letting the ball pass you and fall off the edge.

On the higher screens there are diamond-shaped bricks which when struck by the ball will destroy every brick in the arena, allowing you to move on to the next level. If you shoot the escape brick with a missile it will not function correctly and you cannot exit that screen.

Certain shapes of brick will replenish your missile stocks while others will earn you extra balls. But by accidentally striking the escape brick before hitting these bonus blocks you miss the opportunity to improve your chances of survival.

The perching purple frogs are not the only inhabitants of the

arena — you will also be attacked by nasties which dwell within the blocks. The first thing you know about these is when you destroy a certain shape of block and get mugged by a monster. The beasts may even be penned up behind a wall, to be released as the ball smashes its way through.

My only gripe is with the game cpntrols, as you are unable to move the bat before the ball has bounced into the arena. If you happen to lose a ball while your bat is at the side of the arena, it is frozen in this position until the new ball is served.

The new ball always appears in the centre of the arena so you have to make a mad dash for the middle of the arena and hit the ball on the run.

The speed of travel of both the ball and your bat are sensitive to the amount of on-screen action. If you've just blasted a massive hole in the wall and the debris is falling to take its place, the action slows down, not dramatically, but enough to serve as an excuse if you lose the ball.

Minor grumbles aside I found *Ballbreaker* to be a stunning game. The level of variety is tremendous and the graphics are a feast for the eyes.

John Revis

Presentation 88%

Opening menu with a good selection of options.

Graphics 92%

Wonderfully solid shapes that look good enough to eat.

Sound 85%

Choice of tune or sound FX.

Playability 85%

Occasionally unpredictable response of bat to controls.

Addictive qualities 92%

So simple but totally absorbing.

Value for money 91%

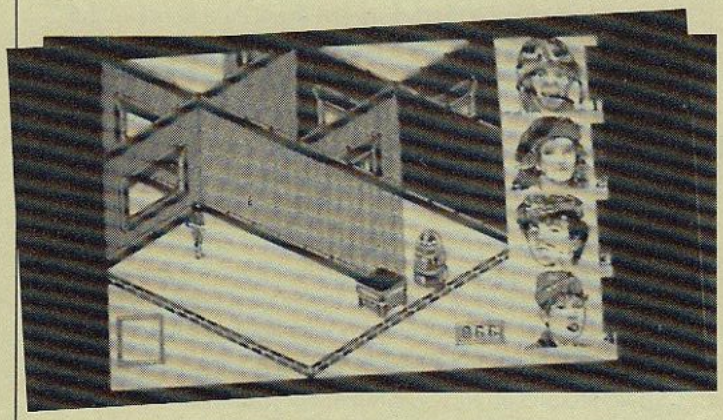
Worth every penny.

Overall 92%

State of the art Breakout.

STRIKE FORCE COBRA

Piranha, $22.50 cass, joystick and keys



Sylvester Stallone (Sly to his friends) has a lot to answer for. Not only does he produce sickeningly violent celluloid epics which show his inability to string together more than three consecutive words ... but he also seems to have inspired the numerous gun-toting characters currently populating the world of computer games.

Strike Force Cobra, a crack elite fighting force of do-gooders, has been formed by the world powers to stop an evil criminal genius from triggering an instant nuclear holocaust.

The mission — to destroy the evil genius's computer hacking system before it is too late. A familiar scenario, huh? Maybe James Bond has a lot to answer for as well!

By fighting their way through the enemy fortress and seeking out the kidnapped scientists, the four members of Strike Force Cobra must obtain the secret codes to the lock of the fortress's main computer room.

Hazards on the way include automatic weapon systems, electronic traps, killer robots and human guards.

The graphic scenario is of the 3D forced-perspective type, so many small rooms are visible at any one time due to the cut-away nature of the graphics.

A combination of joystick and keyboard control allows you to jump, dive, crouch, stand up and kick, but it would have been

nice for everything to have been controlled by just the joystick.

What makes the game slightly different from other games of a similar graphic perspective is the realism of the storyline, and the realistic animation of the characters.

And there's an option to choose four out of eight possible team members, each with a different character profile. Presumably a carefully chosen mix of team members will accomplish the task more quickly.

Your weapons include a lightweight submachine gun, and electromagnetic flux grenades (EFGs) which will either destroy or confuse the enemy, who will attempt to damage your own lightweight body armour.

But first aid facilities can be reached — I managed it several times during the game.

Gameplay in this new title from Piranha is extremely absorbing and will appeal to arcade fans who like to use their brains as well as their fingers.

Victor Laszlo

Presentation 85%

Colourful title page and detailed option screen.

Graphics 85%

Some of the cleanest hi-res graphics around.

Sound 60%

A bit disappoointing, more footstep and gun noises would have worked wonders.

Playability 80%

Action is slightly sluggish, but very good on the whole.

Addictive qualities 86%

You'll keep coming back to solve the mission.

Value for money 85%

THE SYDNEY AFFAIR

Infogrammes, $24.95 cass, $42.50 disc, keys

The crack of rifle fire shatters the peace of a little French town — it makes quite a mess of Mr Sydney's head too! Why should anybody want to kill this quiet family man? As a Detective Sergeant in St Etienne's crime squad it is your job to unravel the mystery of *The Sydney Affair.*

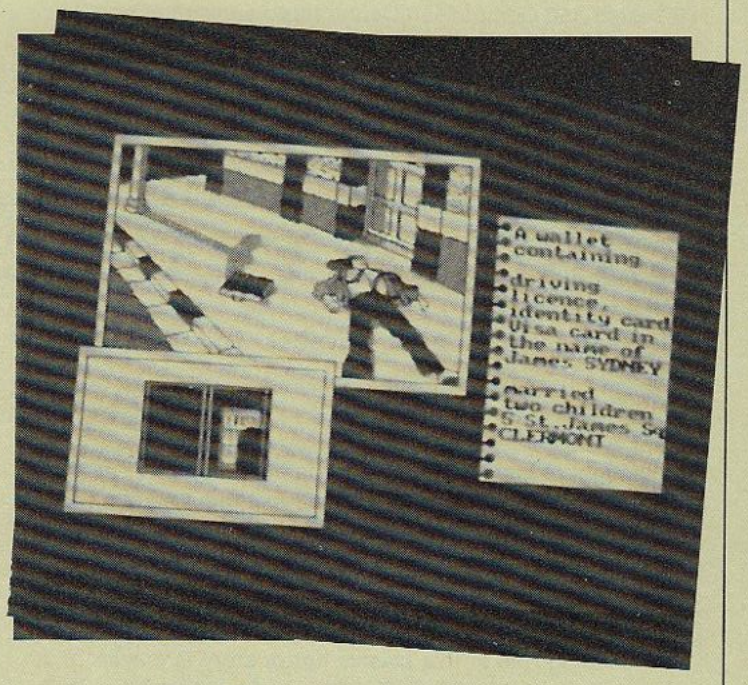
You begin the investigation as any good detective would, by visiting the scene of the crime. The screen is divided into three sections, the largest of which shows a detailed picture of the body and its immediate surroundings.

An icon in the shape of a magnifying glass is used to search the scene for clues. All you have to do is place the icon over the area you wish to examine more closely and press the Copy key.

If the specified area is hiding a clue then the other two screen windows are brought into operation. The lower one displays a picture of the item you have found while the window resembling a note pad identifies it and provides additional information. For example, rifling Mr Sydney's jacket pocket reveals a wallet, the contents of which are detailed on the note pad.

Certain items in the scene are related: Placing the magnifying glass on the briefcase informs you that it is locked. Once you have located the keys you will have more success.

Always be on your guard for sneaky tricks. Having opened the briefcase you find a diary containing names and address-



es. Don't go off and search elsewhere as the briefcase contains yet more goodies, accessed by pressing the Copy key a second time — dirty, eh!

All information should be committed to paper as it is discovered — once stage two of the program has loaded you cannot return to the opening scenes.

One of your team of officers has discovered traces of the assassin in the flat across the road, which forms the basis of the second opening scene. With trusty magnifying glass in hand you scour the room for clues.

In both instances the windows containing the scenes have dimensions of 22 x 12 units. It is therefore a worthwhile exercise to place the magnifying glass on every part of the scene, pressing the Copy key as you go.

With your notebook brimming withsnippets of information you return to the police station to continue your investigation. It is at this point that the second part of the game is loaded. In this stage you are seated at your desk, staring at a blank computer terminal — I know the feeling well!

The French police force has a very advanced computer network at its disposal. when you log-on you can communicate with a variety of official bodies.

Messages about particular investigations can be sent to other police squads in the following format: "Information on 'SYDNEY' affair". If the receiving squad has had any involvement in the case then they will supply you with all relevant information.

The District Squad for Judicial Information can be accessed and details requested on individuals. If the person in question is wanted then you will be told why and by whom. The Judicial Research and Comparison Centre has a similar function, but you can also request details on objects as well as people.

If all this hi-tech police work becomes too much for you there are always the old-fashioned techniques to fall back on. If you know a person's name and address then you can go round and beat a statement out of him! The statements are always brief, but come complete with a pretty photograph of the person being interviewed.

Pressing the Examination key allows you to perform autopsies and ballistic investigations. These explain the bullet hole in the wall and tell you from which building the shot was fired.

The Sydney Affair is an excellent whodunnit program, a real challenge for the amateur detective.

Nev Astly

Presentation 83%

Superb inlay card showing difficulty level, expected completion time and age range.

Graphics 85%

Clever use of graphics for scene-of-the-crime investigations.

Sound 50%

Very limited.

Playability 85%

Heavy going until you get a lead.

Addictive qualities 80%

The computer's uncooperative replies tended to spoil things.

Value for money 81%

Should provide many hours of frustration.

Overall 85%

A real brain strainer.

BRIDE OF FRANKENSTEIN

Ariolasoft, $22.50 cass, joystick and keys

The thought of marital bliss has all been too much for Frankie who has cracked up under the strain. As the future Mrs Frankenstein you take a crash course in body building — all you need now are the parts! The Bride of Frankenstein follows your grave robbing exploits as you strive to give life to your loved one.

The leading role in this classic horror story is played by a rather dumpy sprite who goes by the name of Mrs Stein. With her blonde hair flowing in the wind she explores a very colourful and detailed castle.

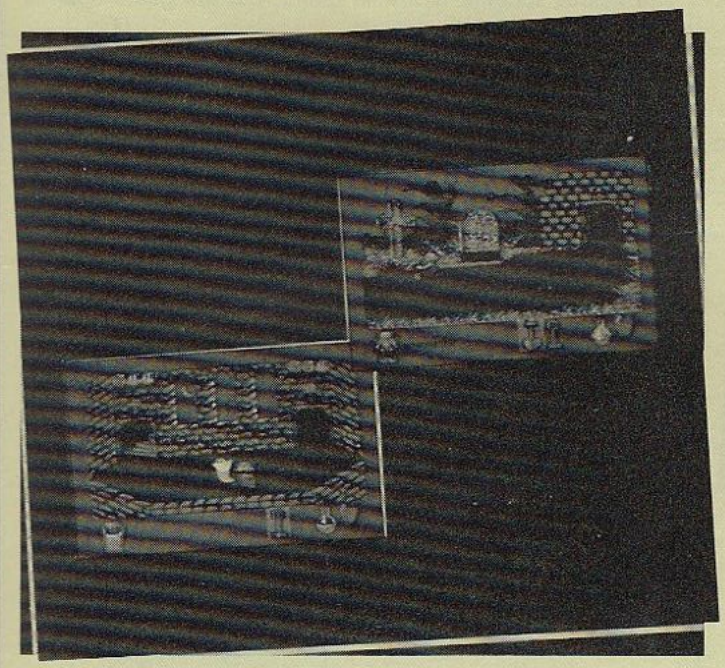
Her fiendish adversaries are also depicted as large cartoon-style characters. By cleverly allowing them to move behind some objects and in front of others the scenes are given a very solid feel.

The 60-screen playing area covers castle rooms, courtyards, graveyards and crypts. Most of these contain doors, the majority of which are locked — surprise surprise!

In various rooms around the castle are keys (seven in all), but each one will only open specific doors and you can only carry one key at a time.

Below the main playing area there are four types of icon. The first is Frankie's coffin — as you acquire new bits they are fitted to his skeleton and a list made of those collected.

Next there are the icons which display what you are carrying (keys, spades, pickaxes and other implements) and you use the cursor keys to highlight the required object.



Icon number three is a glass flask containing green liquid — the elixir of life. The more you roam the castle, the more elixir you consume, but letting your stocks run low is not to be recommended.

Fortunately there are several store rooms within the castle where you can walk to a beaker and fill up. The final icon is a beating heart, but beware of high blood pressure.

Ghosts and skeletons roam the castle looking for Mrs Stein. Contact with these sends her heart rate soaring — prolonged contact will kill her.

Certain areas of the castle have lower ghoul populations than others, but until you locate them you will find that the game doesn't last more than a few minutes.

Ghouls are notoriously difficult to shake off, especially since your speed of movement is reduced when a nasty is in the room.

There are two main areas in which you can settle your heart rate: The first is the pathway to the graveyard, the second the sanctuary room within the castle.

This contains a plentiful supply of elixir but appears to be built on a rift in the time/space continuum. Every time you use the room there is a good chance that you will exit to a different part of the castle, which can be used to your advantage.

The spade and pickaxe are essential tools for the amateur grave robber. Just stand in front of a headstone and dig — three shovelfuls will usually be sufficient to unearth some of Frankie's innards.

The pickaxe is used for opening crypts and clubbing ghouls. Many of the tombs are unlit so a lamp is a definite advantage, unless you like fumbling about in the dark.

The inlay card states that the dungeons contain shackled prisoners and goes on to say that you could set them free ... or use them in other ways — their spare parts are probably much fresher than the ones I've just dug up in the graveyard!

Ariolasoft has got the level of difficulty just right. The game appears almost impossible at first, but with a reasonable amount of perseverance you will soon find your lifespan increasing. This is a first rate arcade adventure game for the horror fanatic.

Carol Barrow

Presentation 86%

Good story line, well laid out instructions.

Graphics 88%

Colourful but chunky.

Sound 66%

The title tune is a little ditty by Bach — played by a tone deaf baboon.

Playability 87%

Ghosts are a major problem until you learn how to avoid them.

Addictive qualities 85%

Has a tendency to be more frustrating than addictive.

Value for money 84%

Good value for this entertaining game.

Overall 86%

Excellent arcade/adventure game.

ACE OF ACES

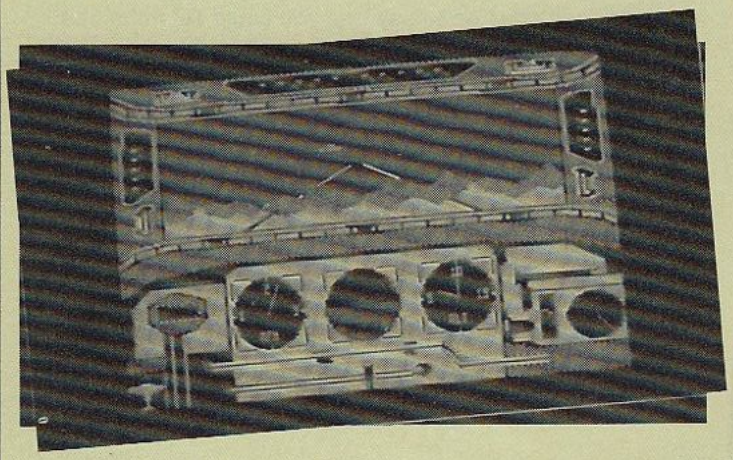
US Gold, $24.99 cass, $42.50 disc, joystick and keys

Ace of Aces puts you at the controls of a Mosquito speed bomber with the objective of knocking out enemy fighters, bombers, V1 rockets, trains and U-boats.

The game starts in the briefing room, where the commander points to a small blackboard from which you must choose to fly a mission or have a practice run.

You can equip your plane with rockets, bombs, cannon or an extra fuel tank for the longer missions — the selections reflect the type of mission chosen.

In the second mission you make your attack runs on U-boats before they dive to safety.



Enemy bombers are heading for London in the third mission. You must destroy them all using rockets — which are not known for their accuracy.

In the final mission your aim is to dispose of V1 rockets while being attacked by enemy fighters.

Unlike many other arcade simulators, the game is divided into five separate elements. The main pilot's display has all the controls which allow you to fly the aircraft, including artificial horizon, altimeter and radar.

The engineer has two views out to the port and starboard wings. From here he must control the throttle and boost settings of the engines.

The navigator's view shows the position of the plane on a map of western Europe, along with the position of any enemey activity.

The bombardier's view includes a full graphical status report of all weapons available and allows the ditching of empty fuel tanks to reduce the plane's weight.

The colour scheme is a little bland, but the game itself is very enjoyable, if a little difficult.

Much of the game seems a little rushed, especially the pilot's screen which is quite cluttered and detracts from what could have been a great game.

Instead this is an above average arcade/strategy/simulation which has a generous amount of mass appeal.

Anthony Clarke

Presentation 87%
Interesting way of selecting options, but the view screens could have been better.

Graphics 67%
Bland and unimaginative, with little colour and depth.

Sound 55%
Whirrr!!! Beeooo.

Playability 63%
Difficult to get the coordination of changing screens and keeping the plane in the air.

Addictive qualities 75%
A game which has that "Just one more go" appeal.

Value for money 64%
A bit steep for this quality game.

HEAD OVER HEELS

Ocean, $22.50 cass, keys and joystick

When Batman appeared all those months ago, it was heralded as one of the best 3D games available. Jon Ritman, author of Batman, has now written *Head Over Heels*, yet another 3D maze runabout, but it is quite a different game.

The Blacktooth Empire has been growing steadily for many years, taking over other planets and ruling by oppression.

The creatures of the peace-loving world of Freedom are becoming worried that their planet will soon succumb to the Blacktooth threat, so they have sent two of their top spies, Head and Heels, to counter the threat.

These two characters are a strange breed of symbiotic animal which can join together and become a larger animal or separate to control tricky situations.

Head, as the name implies, sits on top of Heels. Over the years he has lost his legs and developed strong arms and the remnants of wings which allow him to jump long distances and have a limited amount of controlled flight.

. . .*continued p 37* ►

SIREN SOFTWARE

"Without a doubt Siren Software have produced some of the best disc utilities ever seen on the Amstrad range of computers." Amtix! January 1987

★ NEW ★ DISCOVERY PLUS ★ NEW ★

The ultimate tape to disc transfer program

"Discovery Plus must be the most advanced and probably most efficient tape to disc transfer utility to date" Amstrad Action December 1986

This program will transfer more games to disc than any other transfer program. The first person who can prove otherwise will receive twice his money back!!

Discovery Plus consists of 4 easy to use programs that together will transfer an extremely high proportion of your software onto disc.

Also included is details on how to transfer over 100 games.

Silver Screwdriver Award Amtix! January 1987

Discovery Plus only $39.95 on disc for the 464/664/6128

★ NEW ★ HANDYMAN ★ NEW ★

FORMAT YOUR DISCS TO 416K

Handyman the unique disc enhancement package allows you to manage, use and get more from your discs. Look at these unique features:

- Format your discs to 416K (208K per side on a standard CF2 disc)
- Save unwanted discs onto tape to release expensive disc space
- Full disc/file search and edit. Find and alter messages in programs
- Superb menu maker puts a menu selection system on your discs
- Filemate displays ASCII files, finds text in files, prints files etc etc

"Siren has come up with another marvellous piece of software" Amstrad Action December 1986

"This is just about the best disc utility that I have had to use" Amtix! Jan 87

Amtix! Golden Screwdriver Award Jan 87

Handyman on disc for the 464/664/6128 only $34.95

MASTER DISC +HANDYMAN ON SAME DISC ONLY $57.95

★ ★ MASTER DISC ★ ★

THE DISC USERS UTILITY

Master Disc contains a disc copier, directory editor, fast formatter, sector editor, deprotector, disc and tape header readers, trans disc, trans tape, disc map, typefile, dumpfile & zipdisc.

"The package seems to work very well on the full range of machines" Amtix! June 86

"Each section is fully documented with clear and precise instructions" Amtix! June 86

"This Siren package really does offer you quite a lot for your money" Amstrad Action June 86

"So far we have yet to find a disc that it cannot copy from, it even copies unformatted discs" Amtix! June 1986

Master disc available on disc only $34.95 for the 464/664/6128

TAPE UTILITY

464 OWNERS, LOAD IN YOUR SOFTWARE AT UP TO 4 TIMES THE NORMAL SPEED

Tape Utility will allow you to make back up copies of your tape base software that will load at up to 4 times the normal speed.

. . . So easy to use, simple one key operation
. . . Handles up to 42K (Approx) in one go
. . . Will copy normal, headerless, speedlock & flashloaders
. . . Tests have shown that it will back up about 90% of all Amstrad software
. . . Choice of 10 speeds up to 4000 baud
. . . Removes protection from basic & speedlock programs

"Simply the best, the tape to tape backup copier to beat all tape to tape backup copiers". AMSCLUB

. . . Written specifically for the 464, this is not a Spectrum conversion

TAPE UTILITY ON TAPE ONLY $17.95. AMSTRAD CPC464 ONLY

PRINT MASTER

The printer utility and enhancement package. No printer owner should be without this.

This unique suite of programs will allow you to make the most of your DMP2000 or any Epson compatible printer.

- Superb large 16 shade printer dump of any mode 0 screen
- Large black and white dump of any screen in any mode
- Fast character dump of screen
- Amazing 16K interrupt driven printer buffer
- Print out files from most wordprocessors (Protext, Tasword etc) in a variety of fonts, sizes and styles. Include screen dumps as illustrations
- 10 great fonts included New! Latest version 20 fonts
- A terrific font designer allows you to create your own fonts

This spectacular package is available on disc only for your Amstrad 464/664/6128. Only $39.95 on disc.

To order, please use the form on centre page

ORDERING INFORMATION All prices include postage & packing

SOFTWARE ON TAPE

CAT#	TITLE	PRICE
1001	TASWORD	$36.95
1006	TOOLBOX	$19.95
1007	FLEXIFREND	$19.95
1008	GRASP	$19.95
1009	CHAOS FACTOR	$15.95
1010	MUZICO	$17.95
1011	DRUMKIT	$16.95
1012	MUSIC COMPOSER	$17.95
1013	EASIDATA II	$29.45
1014	EASIDATA III	$41.45
1015	DATABASE/MAIL LIST	$29.45
1022	QWERTY	$14.95
1024	MYRDDIN FLIGHT	$17.95
1030	AMS-FORTH	$25.00
1056	TAPE UTILITY	$17.95
1201	PLAN-IT (CPC)	$36.95
1202	MINI-OFFICE II	$36.95
1203	MAGIC SWORD	$21.95
1204	FUN SCHOOL (2-5)	$15.95
1205	FUN SCHOOL (5-8)	$15.95
1206	FUN SCHOOL (8-12)	$15.95
1207	CHARTBUSTERS	$15.95
1208	CLASSIC GAMES	$15.95
1209	RED ARROWS	$27.95

SOFTWARE ON DISK

CAT#	TITLE	PRICE
2001	TASWORD	$48.95
2007	FLEXIFREND	$31.95
2008	GRASP PLUS	$29.95
2009	CHAOS FACTOR	$27.95
2012	MUSIC COMPOSER	$29.95
2014	EASIDATA III	$53.45
2015	DATABASE/MAIL LIST	$41.45
2016	EASI-WORD COMBO	$49.95
2017	GENESIS	$39.95
2018	EASY MUSIC	$34.95
2019	POT POURRI VOL. 1	$19.95
2020	POT POURRI VOL. 2	$19.95
2022	QWERTY	$26.95
2024	MYRDDIN FLIGHT	$29.95
2051	DISCOVERY PLUS	$39.95
2052	HANDYMAN	$34.95
2053	MASTER DISC/HANDYMAN COMBO	$57.95
2054	MASTER DISC	$34.95
2055	PRINT MASTER	$39.95
2201	PLAN-IT (CPC)	$48.95
2202	MINI-OFFICE II	$48.95
2203	MAGIC SWORD	$33.95
2204	FUN SCHOOL (2-5)	$27.95
2205	FUN SCHOOL (5-8)	$27.95
2206	FUN SCHOOL (8-12)	$27.95
2207	CHARTBUSTERS	$27.95
2208	CLASSIC GAMES	$27.95
2209	RED ARROWS	$39.95
2301	PLAN-IT	$57.95
2401	MT-BASIC (CPC)	$125.00
2402	MT-BASIC (PCW)	$125.00

BOOKS

CAT#	TITLE	PRICE
3001	AMSTRAD HANDBOOK	$ 9.95
3002	AMSTRAD COMPUTING	$17.95

PUBLIC DOMAIN DISKS

CAT#	TITLE	PRICE
2801	PD VOL. 1	$21.95
2802	PD VOL. 2	$21.95
2803	PD VOL. 3	$21.95
2804	PD VOL. 4	$21.95
2805	PD VOL. 5	$21.95
2806	PD VOL. 6	$21.95
2807	PD VOL. 7	$21.95

TAPES

CAT#	TITLE		PRICE
7008	DIAMOND DIG	(8/86)	$7.50
7009	ICE FRONT	(9/86)	$7.50
7010	da BELLS	(10/86)	$7.50
7011	DISCMAN	(11/86)	$7.50
7012	ROGBOT RON	(12/86)	$7.50
7101	OTHELLO	(1/87)	$7.50
7102	SPACE BASE	(2/87)	$7.50
7103	GALACTIC INV.	(4/87)	$7.50
7104	SMILEY	(3/87)	$7.50
7105	BACKGAMMON	(5/87)	$7.50

DISKS

CAT#	TITLE		PRICE
7051	7008/7009/7010	AS ABOVE	$19.95
7151	7011/7012/7101	AS ABOVE	$19.95
7152	7102/7103/7104	AS ABOVE	$19.95

Computing With The Amstrad SUBSCRIPTIONS

CAT#	TITLE	PRICE
5001	MAGAZINE ONLY	$45.00
5002	MAGAZINE + TAPE	$90.00
5003	MAGAZINE + QUARTERLY DISK	$110.00

Computing With The Amstrad BACK ISSUES

CAT#	TITLE	PRICE
6008	CWTA PREMIERE EDITION	$4.95
6009	CWTA SEPTEMBER ISSUE	$4.95
6010	CWTA OCTOBER ISSUE	$4.95
6011	CWTA NOVEMBER ISSUE	$4.95
6012	CWTA DECEMBER ISSUE	$4.95
6101	CWTA JANUARY ISSUE	$4.95
6102	CWTA FEBRUARY ISSUE	$4.95
6103	CWTA MARCH ISSUE	$4.95
6104	CWTA APRIL ISSUE	$4.95
6105	CWTA MAY ISSUE	$4.95

How to order

1. MAIL ORDER

Please include all information requested on the order form, and make sure you have enclosed your name and address (you'd be surprised!) and the correct amount for the goods you require.

If sending a cheque or ordering using Visa, Mastercard or Bankcard please ensure that the date on your cheque is valid (ie 1987 not 1986) and that your credit card has not expired.

2 TELEPHONE ORDER [008] 030930

Please follow the instructions below carefully ***before*** ringing. This number will only be answered by a machine and cannot be used for general enquiries or messages. Anything other than an order will be ignored — you have been warned!

A) Complete the order form at right as though you were going to order by mail. Do not wait until ringing before deciding which titles you require or trying to find your credit card. The answering machine is voice activated and any pause over a couple of seconds will result in the machine hanging up on you.

B) When the machine answers, read the order from your order form slowly, clearly and distinctly giving all the information you have written down. Where possible leave a telephone number just in case we can't understand or hear your order.

C) This service will be in operation 24 hours a day, every day of the year.

D) Allow 28 days for the delivery of your order — orders which we cannot despatch within 2-3 days of receipt will be advised of the likely delay by mail. Back issue orders will be mailed at the same time as the current issue.

MAIL ORDERS TO: STRATEGY SOFTWARE
P O BOX 11, BLACKMANS BAY, TASMANIA 7152
Enquiries (002) 29 4377
Orders, including subscriptions – local call fee – (008) 030930
Use your VISA, MASTERCARD or BANKCARD for mail orders.

ORDER FORM

CAT #	TITLE	PRICE
	TOTAL	

Bankcard ☐ Mastercard ☐ Visa ☐

☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

Signed ..... Expiry ☐☐☐☐

Name .....

Address .....

..... Postcode .....

Phone ..... Date .....

Please read *How to order* carefully before using our 008 toll-free phone number

$AVINGS ON $OFTWARE

$AVE $2 this month only on every copy of Toolbox for the Amstrad CPC 464 (tape only)
Cat # 1006.
This original coupon must accompany your order. Limit one coupon for each client per piece of software. Offer closes 15 July 1987. Valid for mail orders only.

$AVE $5 this month only on every copy of Database for the Amstrad CPC 464/664/6128 (tape or disk)
Cat # 1015/2015.
This original coupon must accompany your order. Limit one coupon for each client per piece of software. Offer closes 15 July 1987. Valid for mail orders only.

$AVE $10 this month only on Myrdin Eight Simulator for the Amstrad CPC 464/664/6128 (disk only)
Cat # 2024.
This original coupon must accompany your order. Limit one coupon for each client per piece of software. Offer closes 15 July 1987. Valid for mail orders only.

When the press use such words as 'Phenomenal', 'Outstanding', 'Ideal' and 'Worth Every Penny', they've obviously discovered something rather special.

But when that something special turns out to be a product in which they are already expert, then it must be something very special indeed.

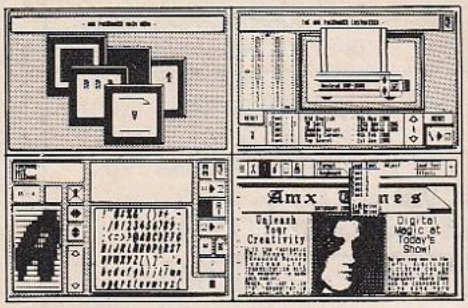
The object of their enthusiasm is AMX Pagemaker – a revolutionary software program that will produce newspapers, posters, leaflets, notices and hand-outs – in fact anything where text and graphics are required, to an extraordinary professional standard.

It's a complete graphics design system and word processor rolled into one. It has real time graphics with fast continuous scrolling up and down an A4 page and uses Mode 2, the highest graphics resolution on the Amstrad CPC computers.

READ ALL ABOUT IT.

You can type directly onto the screen, with any of the 16 typefaces supplied or design your own, alternatively, you can load in any ASCII file or a word processor file, from programs such as Tasword, Amsword, Maxam, or Protext, with fully automatic on-screen text formatting during loading.

'Word processing' facilities such as centering, ragged right and literal justification are all available. There is full pixel resolution control over text and graphics. Also included is a micro spacing facility.



The program that's making front page news.

EXTRA, EXTRA.

There are outstanding facilities for drawing, spraying and painting, using either the patterns supplied, or your own pattern designs. A screen conversion routine is included allowing screens created in Mode I and 0 to be used within the Pagemaker. The cut and paste facilities include copying, moving, rotating, stretching and a fantastic zoom is also available.

The previewer allows you to view three A4 pages at any time before work is output to a wide range of dot matrix printers including: Amstrad DMP 1000-2000, Epson FX/RX/LX/LQ, Canon PW-1080, Kaga KP810, Mannesman Tally MT-80+, Seikosha SP-1000A, Star Delta, Star SG10 and any that are compatible with the above.

The AMX Pagemaker requires: a) Amstrad CPC618 or b) Amstrad CPC664+64K Minimum add-on Ram or c) Amstrad CPC464+64K Minimum add-on Ram + disc drive, DK 'tronics Ram boards or compatible.

Let's leave the last word to the press.

*"Pagemaker" is phenomenal – it lends itself to creating anything where text and graphics are involved – notices, posters, leaflets, hand-outs, newsheets. Packages like this have been the province of the 16-bit micros until now, this product is worth every penny.

AMX MAGAZINE MAKER– WE THOUGHT IT WAS ABOUT TIME WE PUT YOU IN THE PICTURE.

A combination of AMX Pagemaker and the AMX video digitiser. Using any video that provides a composite signal and the digitiser, images from a camera or TV can be converted into a graphic screen on the Amstrad Micro. They can then be used within AMX Pagemaker to illustrate magazines or newsletters. The digitiser connects into the expansion port and scans a complete picture in only 5 seconds.

A special print dump routine is also included with the driver programs. This is specially designed to produce fast, correctly proportioned pictures, with reduced 'Contouring' resulting in a very accurate reproduction of the image.

Features offered by this package include:

- Dot resolution 256 by 256
- Standard 1 volt composite video input
- 10 bit A/D convertor gives 32 grey scale output
- Low IC count
- Contrast and brightness control
- No external power unit required

These packages are your opportunity to join the desktop publishing revolution.

The AMX Pagemaker costs only $125.00 software is supplied on 3" disc and a fully illustrated operating manual, AMX Digitiser only $225.00 including software on 3" disc, and AMX Magazine Maker (including AMX Pagemaker and AMX Digitiser) at any $325.00

Now available direct from Strategy Software at U.K. prices!

CAT #	PRODUCT
4501	AMX MOUSE $175.00
4502	AMX PAGEMAKER $125.00
4503	AMX DIGITIZER $225.00
4504	MAGAZINE MAKER $325.00
4505	AMX MAX $49.95

Please use order form on centre page

BANKCARD, MASTERCARD OR VISA ORDERS CALL 008 29 4377

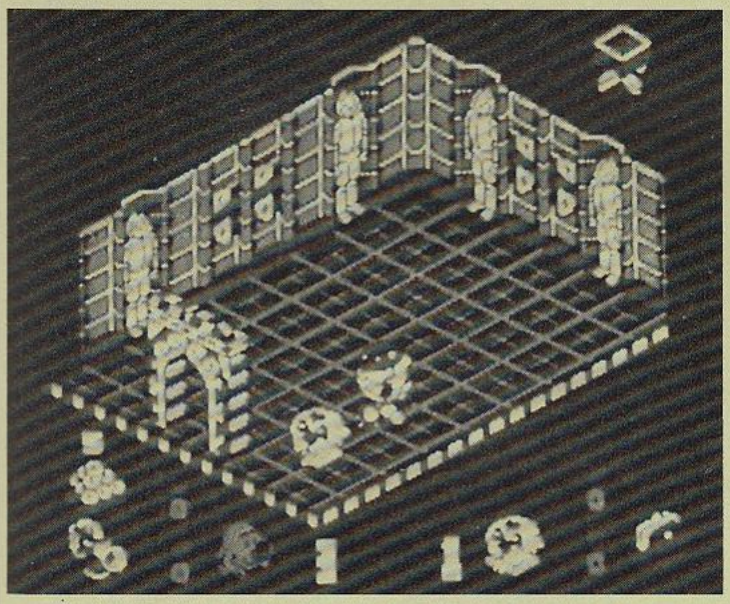
HEAD OVER HEELS

. . .continued from p 32

Heels has also changed, losing his arms and developing strong legs with which he can run very fast.

The differences between them mean that only together do they have any hope of completing their mission — to begin revolution on all of the Blacktooth Empire's auxiliary planets.

The game starts with the two characters separated and in different parts of a large gym, making the first order of business to bring them together while collecting two pieces of vital equipment.



From here they must then teleport to each of the four planets in turn and collect the Crowns of Blacktooth with which to start the revolt.

The first planet in the system is Egyptus, a strange world where civilization seems to revolve around people wrapping corpses in bandages and putting them in pyramids.

The Penitentiary is the Empire's prison planet, from which few return. It also holds the secret of the Pit.

A densley-vegetated planet, known at Safari, is the third in the system. The natives live in wooden houses and set traps for unwary animals.

The final planet is Book World. This is a huge Western Library used only by the Emperor and his minions, who are keen on the old West.

The task before the heroic pair is not as impossible as it first seems, and you will find several objects to aid your quest.

A holdall used by Heels gives the creature a limited ability to carry objects around the current screen.

You must collect a crown from each of the four planets before you can embark on the final quest. Teleports are scattered over the landscape, and these provide the only access to certain rooms.

Springs, switches and conveyor belts make up the rest on the landscape, making progress easier, but at times leading the two characters into a trap. In a similar way to Batman, you will find magic objects which give a character limited special powers, but they do run out with time and do not always work on both characters in the same way.

Joining Head and Heels together and picking up a special power will result in both characters gaining the new abilities.

There are options to change the sensitivity of the controls, the volume of the sound, redefine the keys and choose a joystick.

Head and Heels have to be controlled separately by switching between the two, though if one of the pair loses all his lives the other will be left on his own.

The game features 25 pieces of music, which herald different actions, such as finding a crown or picking up a bunny. There is a greater variety in the graphics and backgrounds than Batman, making it a much larger game.

The graphics are of the same high quality as Batman, though the souind is a great improvement. Even if you are not one to buy 3D games, take a look at this one — somehow it feels very different from any other game of this type I have ever played.

Anthony Clarke

Presentation 92%

Plenty of logical options.

Graphics 93%

Beautifully-defined characters that glide around the screen without a glitch.

Sound 75%

A little spectrum-like, but otherwise very good.

Playability 91%

Instantly playable, though some of the traps are unfair.

Addictive qualities 92%

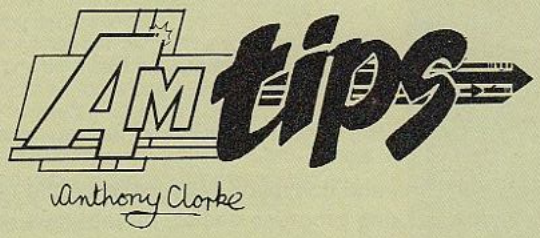
Should appeal to all, including those who normally find 3D games boring.

Value for money 94%

More of a game for less pennies than the competition.

Overall 92%

Should grace the shelves of every CPC owner.



For those of you new to AMTIX!, Amtips is our monthly section that helps you get more out of computer games. Just when you thought you had reached a hopeless situation, along comes a bright spark with all the answers,and a POKE or two to get you out of a jam.

MISSION OMEGA

(Mind Games)

The best robots to build are spheres with IR sights, a large LASER and the second power supply. They are fast, use few raw materials and run for long periods before having to re-charge.

When you enter a room with a teleport system, move on to the flashing square when the colour for the sector you require comes up. The colours relating to each sector have to be worked out by trial and error, but once found are good for the whole game.

The Fourth Protocol

(Century Communications)

I only began to play this game a short while ago, but here are some tips for section one that should slow down the decision to send you to the West Falklands.

Thorn is not the real problem — try putting 10 watchers on the person complaining about him and see what develops.

Only one person has access to all the NATO papers, so put watchers on him by at the latest 10 days into the game or things will start to fall apart.

Pasternak should be answered positively and swiftly. Though an umbrella may be his undoing, it is all for the best really.

When Telecom offers help, take it quickly. Try a voice print on the traitor who has access to the NATO papers.

When a person goes missing there is no point in putting watchers on him, but his wife, on the other hand, is a different matter.

Put at least 25 watchers on ABBS. Best not to feel her collar — bad publicity does wreck the image, and all she may bring another lost sheep back to the fold.

Building 17 security is simply a matter of common sense — who would want to steal top-secret loo rolls. Also remember that emergency exits must be kept clear at all times.

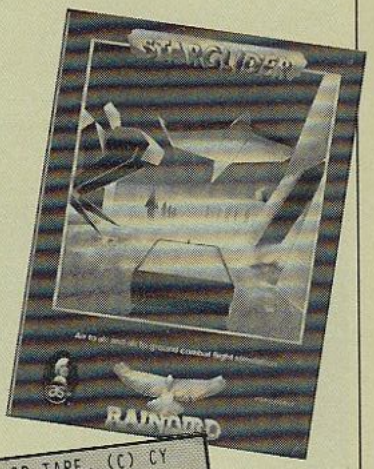
When a high-ranking official gives you a call, don't hang about. Call him back at once, but you had better have all the answers.

Get all the information possible from Blenheim, and if a foreign data exchange is possible then use that as well.

Having a problem with hackers? 3421 is a possible way to stop them.

STARGLIDER

(Rainbird)



For those of you not used to disc pokes, here's how to use them. Type in the listing as shown. Save it to a separate disc, insert the game disc, type Run then press Enter or Return. The program should now do all the rest.

This poke from Cy Booker should make Starglider a pushover. It's best if you only include the pokes which you think you need, so the game doesn't get too easy.

```
100 REM STARGLIDER DISC OR TAPE, (C) CY
BOOKER 1987
110 REM
120 MODE 1
130 INK 0,26:INK 1,0:INK 2,6:INK 3,2
140 LOAD "LOADPIC.SCR"
150 MEMORY 8191
160 LOAD "K32"
170 LOAD "K16"
180 POKE &6334,0: REM INFINITE SHIELDS
190 POKE &6307,0: REM INFINITE POWER
200 POKE &6370,0: REM INFINITE LASERS
210 POKE &6504,0: REM INFINITE MISSILES
220 POKE &2d79,0: REM ALWAYS GET A MISSI
LE WHEN DOCKED
230 POKE &2d81,0: REM MORE THAN TWO MISS
ILES ALLOWED
240 POKE &64F4,0: REM ALWAYS HAVE MISSIL
E MK16
250 POKE &588C,0: REM REAR VIEW
260 POKE &6CF1,24:POKE &6CF2,&F7: REM CA
N'T BLOW UP YOUR EQUIPMENT
270 A$="*CB#."
280 FOR loop=1 to 5
290 addr=addr+10^(loop-1)*(ASC(MID$(a$,l
oop,1)) AND 7)
300 NEXT loop
310 CALL addr
```

SENTINEL

(Firebird)

Sentinel must be one of the simplest, yet most challenging games ever to have graced the CPC. Just to put you on the right course, here are a few tiplets worked out by the AMTIX! gang.

When dealing with just a single Sentinel on the map, it is best to check the direction in which it is moving and then jaunt around the map, sucking off all the trees and deftly avoiding the Sentinel's stare until the map is cleared.

When a jump has been made to another shell, the robot should be facing the shell you left behind, though you may have to look up or down to see it, as the robot looks straight ahead after an energy transfer.

Always, when energy permits, place a boulder before a robot, so reducing the chances of not being able to see the base of the robot and losing the energy it contains.

Some squares may seem perfectly in view but the object there will not be sucked off. Try placing the cursor to the far left, right of the square or one pixel below.

It is helpful to learn in what situations an object can be sucked off and so increase the amount of energy you have when hyperspacing to the next map. If the level contains Guardians, try to remove them as soon as possible as they tend to cover one another, scanning you in unison.

Remember that when you first enter a screen the Sentinel or Guardians do not move until you try to upset the energy balance of the plane.

Use the fact to assess the possible moves you can make and try to engineer a way of gaining height quickly.

Finally, always turn the cursor off when you want to move the robot's head left or right as this speeds up the process, as does using the U-turn facility in otherwise difficult situations when there seems to be no option but to use hyperspace.

Level	Code
0681	70065775
0707	27292626
0744	84481593
0764	02744138
0798	98449867
0811	14444428
0814	31875385
0835	57060738
0858	99309875
0890	85245687

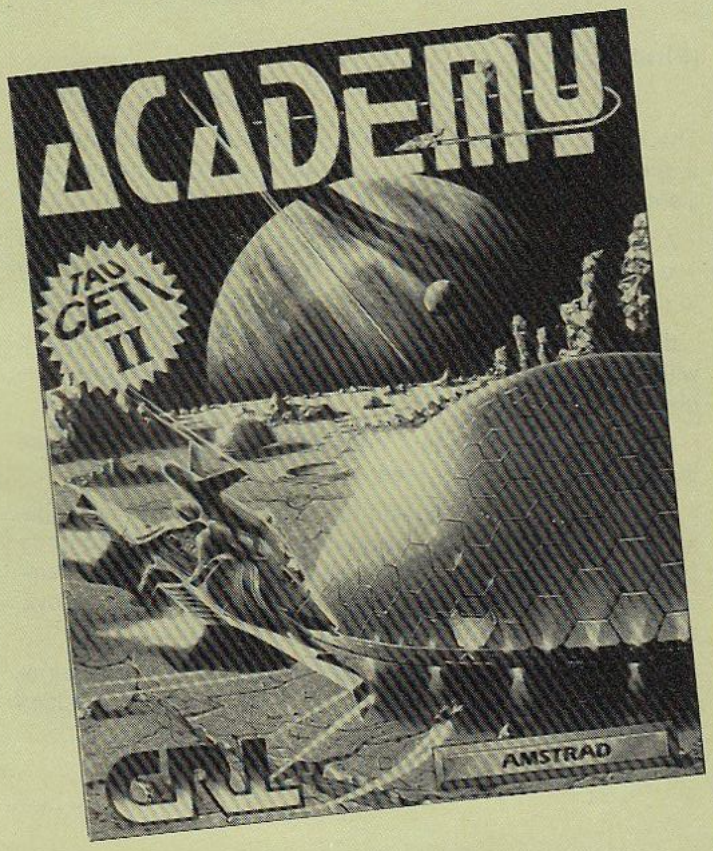
Some of you may be a little impatient to get through the 10,000 levels, so here is a booster to those dizzy heights.

We are currently in the 2000 range, more codes to come next month, but for now the codes listed should have a mix of different situations that will put you in fine form for the later levels.

Academy

(CRL)

Try to build a craft with the correct characteristics for the mission selected. Always go for a good shield and laser power, with steering being the next most important.



An infra-red unit is lighter than flares. The IR unit does not have definition of the flares but with a little practice it becomes easier to use.

ALWAYS take an ADF unit, or you will lost before you know it, and the chances of happening upon the base again are quite slim. It is possible, however, to navigate by the stars, but just like the real thing this takes months of practice.

IKARI WARRIORS

(Elite)

Just a quick hint for survival on the bridges when you have no tank. Walk along the gap between the start of the bridge and the wall so that you are standing on the supports.

Walk along this support with the toggle set facing in towards the centre of the bridge and keep firing. You should be safe from enemy fire, while still within reach of your gun. Grenades are useless here as they just sail over the enemy's head.

ELITE

(Firebird)

The key to anticipating which type of craft is attacking you, and whether to shoot back, can mean the difference between life and death.

Any ships that appear in your forward view are safe, unless you happen to be a fugitive. In this case a fer-der-lance will try to destroy you for the bounty.

If, when the ship is more than just a blob, you see a point with three other points radiating out from it at 90-degree angles then shoot away.

After all, bounty hunters are the scum of the universe so there is no legal penalty, even if you shoot first.

Any ship appearing outside the forward view is free range, so blast away. BOAs have soft underbellies — they seem to leave themselves open for a long time after flying by. If damaged a BOA will break off the attack. Blast them for the alloys, but you have to be quick as they don't hang about.

Once you are a competent pilot, make sure you have a galactic hyperspace at all times, as the first mission empties your fuel tanks.

RESCUE ON FRACTALUS

(Activision)

Try to perfect shooting saucers at long range, or fly in fast, tight circles until they leave the area. If you see a pilot with a green helmet, don't waste time — just turn the systems back on.

When searching at night it is better to fly high and slow. When a pilot comes into range, decrease height carefully, watching the ground level indicator. Flying at night is an essential skill, so once you get the hang of flying, jump straight to level sixteen.

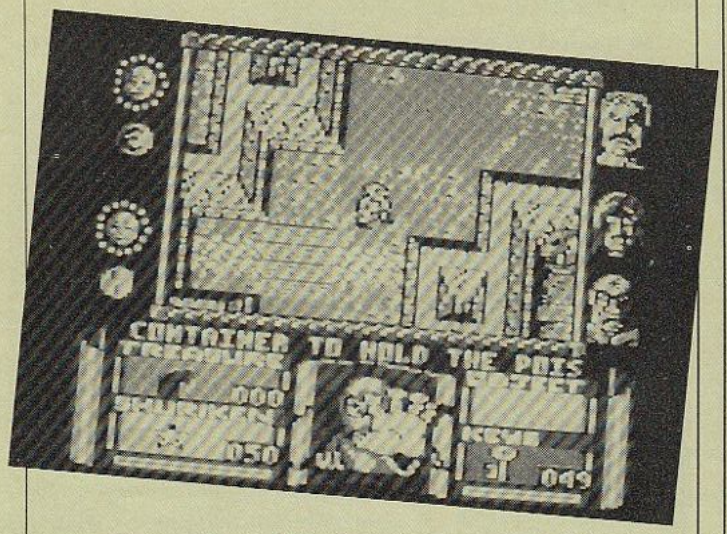
If you are just going for a high score stay on the lower levels, collecting as many pilots as possible before returning to the mother ship.

AVENGER

(Gremlin Graphics)

This time Cy makes it much easier to complete the game by allowing you to always call Kwon in your hour of need, among other things. Just use the same technique to enter them as with Starglider, but this time the poke is for the tape version only, so save the poke to a separate cassette.

```
100 REM AVENGER POKES, TAPE (C) CY BOOKE
R 1987
110 REM
120 MEMORY &1FFF
130 addr=&BE80
140 READ A$:IF A$="-1" THEN 170
150 POKE addr,VAL("&"+A$)
160 addr=addr+1: GOTO 140
170 LOAD "AVENGER"
180 POKE &2083,&80: POKE &2084,&BE
190 CALL &2000
200 REM
210 DATA AF,32,C1,6C : REM CAN ALWAYS CA
LL KWON
220 DATA AF,32,63,B1,3E,B7,32,36,AA : RE
M FIRE POWER
230 DATA 3E,B7,332,8E,A2,AF,32,1E,AA: RE
M KEYS
240 DATA 3E,18,32,E6,A2: REM SPIDERS DON
'T TAKE KEYS
250 DATA C3,D8,65,-1
260 REM
270 REM NB QUIT GAME = CTRL 2
280 REM THIS GAME CONTAINS A RANDOM NUMB
ER ROUTINE COPIED FROM A BOOK BY D WEBB.
290 REM SUPER CHARGE YOUR SPECTRUM
```



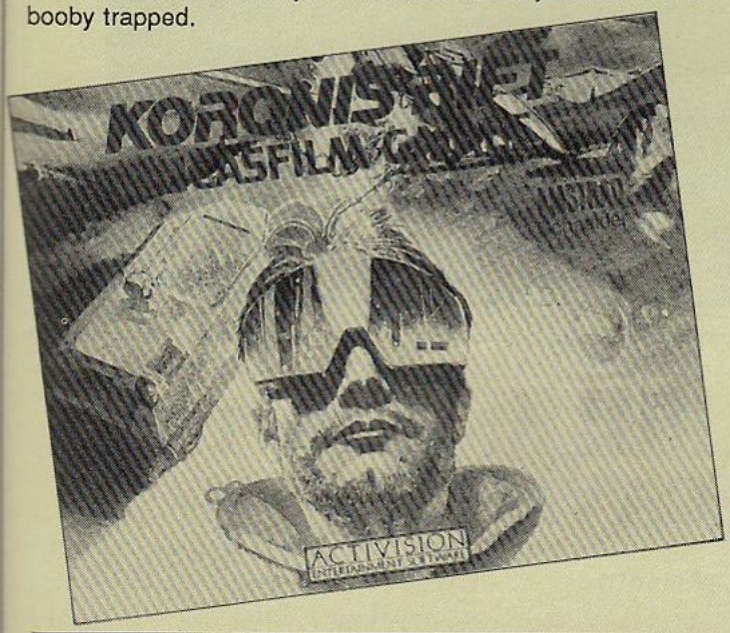
KORONIS RIFT

(Lucasfilm Games)

When collecting artifacts, don't dismantle them until you know you have no use for them. This way an armoury can be built up quite quickly, and eventually you should have a complete cargo of differing lasers and shields.

The manual is right about the better equipment available in the opposite direction to the one you are facing on landing, but make sure you already own a fairly good laser and shield as you will be attacked by two saucers (yellow and blue) the moment you land.

On the first rift, destroy the first brown hulk you find as it is booby trapped.



WIZARD'S LAIR

(Bubble Bus)

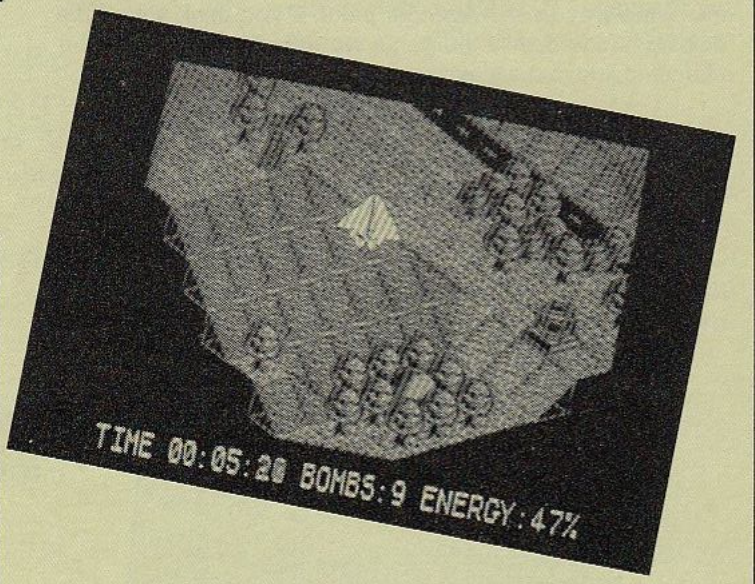
If you are still stumped on how to use the magic lifts, try using the codes listed here, and make the journey less hazardous.

LEVEL	LOCATION
1	CAIVE
2	HAWLO
3	CRYPT
4	DUNGN
5	VAULT
6	LIAYR
7	LYONS

GLIDER RIDER

(Quicksilver)

To deactivate a laser, simply run into the nearest pylon. The laser should now start to fire randomly into the air and give you a chance to take off and destroy a reactor.



HARDBALL

(Amsoft/Accolade)

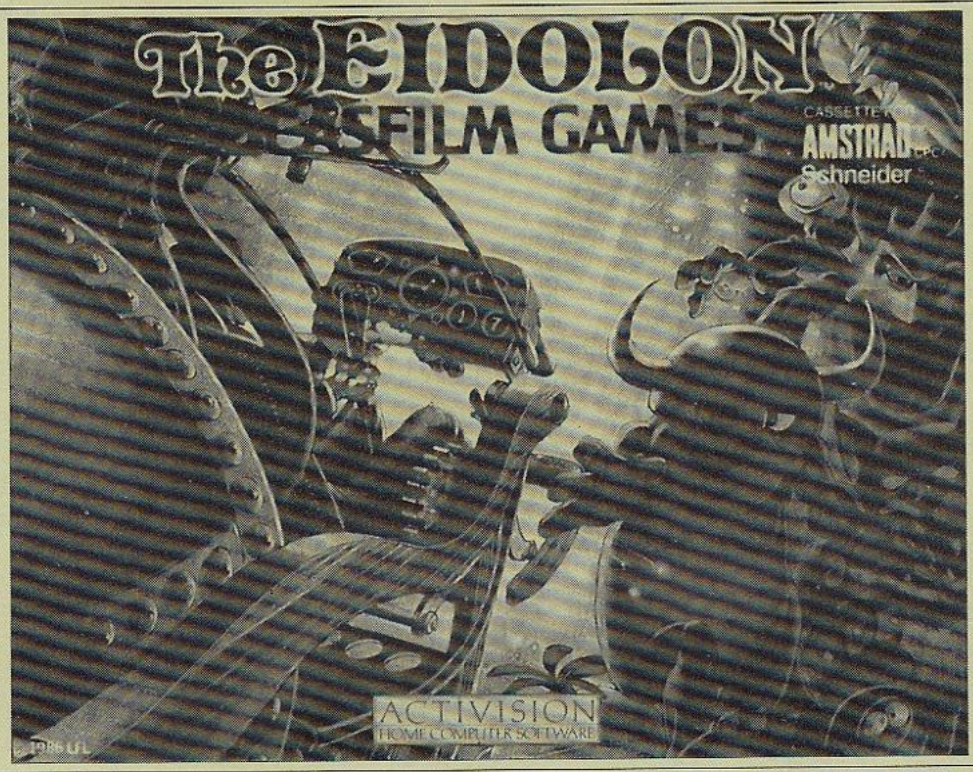
Keep bunting until the bases are loaded. Then swing for all you're worth and you should pick up four runs. Most people have a tendency to swing too early — just slow down a little and you should hit most balls.

You can gain an idea of where the ball is going to be pitched by watching the backstop. If his gloves move in a particular direction then swing at the point the glove moved to.

THE EIDOLON

(Lucasfilm Games)

A fairly obvious tip this one, but one that many people fail to spot. On the early levels always collect all the jewels as at some stage of the game you may come across a level that seems impossible. But if you already have the jewels just carry on to the next level.



DANDY

(Electric Dreams)

Our final poke also comes from Cy Booker. Use it in just the same way as the Avenger poke. If you think any of the helping hands makes the game too easy then leave out the offending line, but always keep the last line in.

```
100 REM DANDY pokes tape, (C) CY BOOKER
1987
110 REM
120 MEMORY &3333
130 addr=&77a0
140 READ a$: IF a$="-1" THEN 170
150 POKE addr, VAL("&"+a$)
160 addr=addr+1:GOTO 140
170 LOAD"!"
180 POKE &416d,&9d
190 POKE &41ba,9
200 POKE &4703,&77
210 CALL &4890
220 REM
230 DATA 21,20,00,22,c5,29: REM 8000 odd
energies when you eat.
240 DATA af,32,9b,36: REM Can walk throu
gh monsters.
250 DATA af,32,39,4f: REM Monsters don't
bite.
260 DATA af,32,91,30: REM Keep Keys.
270 DATA af,32,99,4e: REM Keep flashes.
280 DATA c3,a0,0f,-1
```

ROLAND IN THE CAVES

(Amsoft)

An oldie but goodie here, reprinted because several people have written in asking for it.

When the game starts, hold down the DOWN cursor key and you should receive a hefty bonus and progress on to the next level.

Next months issue will have

MORE

AMTIX reviews

and

AMTIPS tips

Pair those registers for more micro power

Part VI of MIKE BIBBY's introduction to machine code

If you thought we learned quite a lot last month, wait till you see what we've got lined up for you now!

Let's recap a little though. We've learned that the Z80 has several 8 bit registers — that is, internal memory locations that can hold one byte. We refer to them by the letters, H, L, B, C, D, E and A.

We've also met quite a few instructions involving these registers.

For example:

LD r,n

where r is one of the internal registers and n is a number in the range O to 255. This allows us to load a register with a number or constant.

Also:

LD r,r'

where r' is another of the registers. This allows us to transfer the contents of one register to another.

INC r

increases the contents of a register by one while:

DEC r

decreases the contents of a register by one.

So far these instructions work for all registers. We did find, however, that some demanded the use of the A register. For instance, we could peek and poke memory via the A register.

To poke we used:

LD (pq), A

and to peek:

LD A, (pq)

where pq is a two byte number specifying an address in memory. These two instructions are restricted to the A register only. There's no such instruction as:

LD B, (pq)

for instance.

A similar restriction applies to the ADD and SUB commands — they only apply to the A register.

They appear as:

ADD A, r

and

SUB r

(Notice you don't have to specify the A register for SUB. The micro knows you're subtracting r from A).

And you can also ADD or SUB constants to or from the A register,

So:

ADD A, n

SUB n

exist.

Lastly, we learned that increasing the contents of a single byte past 255 caused "the clock to start again".

That is:

255 + 1 => 0

255 + 2 => 1

Similarly:

0 - 1 => 255

0 - 2 => 254

We're really doing quite well with single byte registers. There's one difficulty though: the numbers are too small. After all, if we wanted to specify a part of the graphics screen, we could be dealing with numbers well over 255.

We're used to handling bigger numbers than can be held in a single byte every time we specify an address. All our addresses take two bytes, as we've seen. For example, &3000 consists of two bytes — a hi byte (&30), followed by a lo byte (&00). We've used this every time we've used CALL.

This is the instruction that's similar to a GOSUB — except we follow it with the start of a machine code address, not a line number.

The routine we call ClrText, which clears the text screen, starts at address &BB6C. Again, two bytes — hi byte &BB, lo byte &6C. When we translate the mnemonics into hex though, the lo byte comes before the hi byte.

CALL &BB6C

becomes:

CD 6C BB

So to handle things like addresses and numbers bigger than 255 we need two bytes. Single byte registers, therefore, tend to be a bit restrictive.

To overcome this, the Z80 allows us to pair our registers to give 16 bit registers:

H combines with L to give 16 bit register HL

B combines with C to give 16 bit register BC

D combines with E to give 16 bit register DE

This means we can do things such as LD HL, &AAFF.

This will load the register pair HL with the two byte, 16 bit number &AAFF. H will contain the hi byte (&AA) and L will contain the lo byte (&FF). That's how they get their name: H for HIgh, and L for LOw.

Let's prove it with Program 1.

So what are we up to? If you believe what I've said so far LD HL, &AAFF will store &AAFF in the register pair HL. The opcode for this is 21.

```
address  hex code  mnemonics
 3000    21 FF AA  LD HL,&AAFF
 3003    7D        LD A,L
 3004    32 F8 2F  LD(&2FF8),A
 3007    7C        LD A,H
 3008    32 F9 2F  LD(&2FF9),A
 300B    C9        RET
```

Program I

Notice that the value &AAFF, like an address, is translated into lo byte, hi byte form so LD HL, &AAFF translates as 21 FF AA.

To sort out which holds the hi byte, H or L, we then copy the contents of the L register into A with LD A, L and then poke the first byte of Hexer's "workspace" with LD (&2FF8), A.

To have a look at what's in the H register, we LD A, H and poke it into the next byte of workspace with LD (&2FF9), A.

Then, of course, there's our obligatory RET.

We're forced to do the transfers to A, by the way, because we can't poke the other registers into memory.

Incidentally, if you haven't already typed in our hex loader and simple monitor Hexer from our March 1987 issue, might I suggest you do so now. These articles have been written with it in mind.

If you run Program 1 and examine the workspace afterwards, you'll see that &2FF8 (where what was in L was stored) contains &FF, and &2FF9 (which holds what was in H) contains &AA.

So it's proved, a 16 bit load (that's what we call things like LD HL, &AAFF) stores the hi byte in H and the lo byte in L. Notice, by the way, I've arranged for the bytes to be stored in our workspace in the standard lo byte, hi byte form.

Program II should be familiar from last month. It puts an asterisk on the screen.

```
address  hex code  mnemonics
 3000    CD 6C BB  CALL ClrText
 3003    26 14     LD H,&14
 3005    2E 0C     LD L,&0C
 3007    CD 75 BB  CALL PosTCur
 300A    3E 2A     LD A,&2A
 300C    CD 5A BB  CALL CharOut
 300F    C9        RET
```

Program II

We could replace the instruction LD H and LD L (8 bit loads as they are known) with a single 16 bit LD HL as in Program III.

We still want to load H with &14 and L with &OC. Remembering that the hi byte goes in H and the lo byte in L the instruction we want is:

LD HL, &140C

which translates as:

21 0C 14

in lo byte, hi byte fashion.

If you run it you'll see that Program III does in fact place the asterisk where we want it — column 20 (&14 in the H register) row 12 (&0C in the L register).

```
address  hex code  mnemonics
 3000    CD 6C BB  CALL ClrText
 3003    21 0C 14  LD HL,&140C
 3006    CD 75 BB  CALL PosTCur
 3009    3E 2A     LD A,&2A
 300B    CD 5A BB  CALL CharOut
 300E    C9        RET
```

Program III

We can load our other register pairs with constants, as you can see from Table 1. Program IV uses a 16 bit load into the DE register pair as well as into the HL register pair.

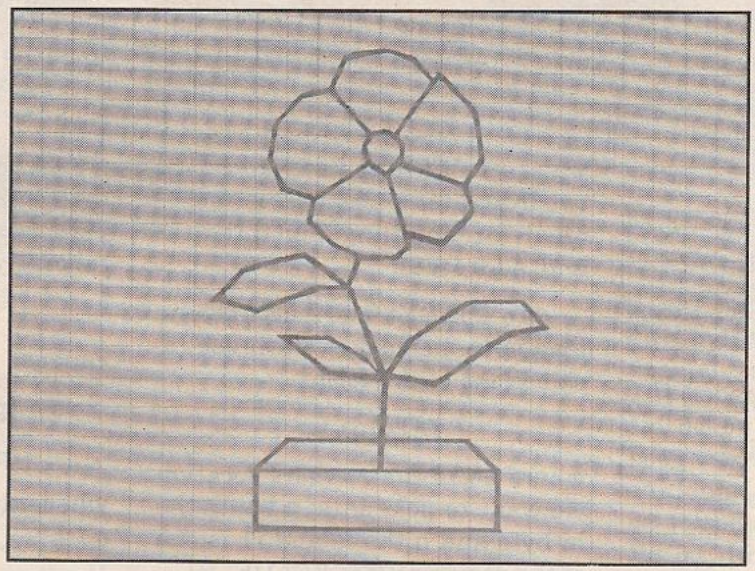
```
mnemonics  opcodes
LD BC,nn   01 n n
LD DE,nn   11 n n
LD HL,nn   21 n n
```

Table 1:Opcodes for loading register pairs with constants

It uses a routine called GrafLine (&BBF6) to draw a line on the graphics screen which we've previously cleared with ClrGraf (&BBDB). GrafLine uses register pair DE to specify the X coordinate, and register pair HL the Y coordinate, of the line's endpoint.

Its starting point is, of course, the last point visited by the graphics cursor, as in Basic's DRAW command. Since we've cleared the screen, this will be at (0,0).

As you'll see from the listing, our X coordinate is 639 (&27F) and our Y coordinate 399 (&18F), so GrafLine will draw a line from the bottom left to the top right corner of the screen. Run it and see.



```
address  hex code  aneamonics
 3000    CD DB BB  CALL ClrGraf
 3003    11 7F 02  LD DE,&27F
 3006    21 8F 01  LD HL,&18F
 3009    CD F6 BB  CALL GrafLine
 300C    C9        RET
```

Program IV

The return to Hexer (or Ready message if you've called it from Basic) will cause the line to scroll off the screen. Can you alter Program IV so that it waits for a key before returning? You'll find the Charln routine at &BB18 useful.

Now you've encountered GrafLine, why not use it to draw some pretty pictures? The routine corrupts all the registers we've met but the Amstrad's firmware, or operating system, keeps track of where the graphics cursor has reached, so this needn't worry you if you're simply plotting on from where you've just reached.

You might find the routin GrafMove at &BBCO useful while you're at it. This is the machine code equivalent of Basic's MOVE, with DE specifying the X coordinate and HL the Y coordinate. Again, our registers are corrupted.

```
address  hex code  aneamonics
 3000    CD DB BB  CALL ClrGraf
 3003    16 02     LD D,&02
 3005    1E 7F     LD E,&7F
 3007    26 01     LD H,&01
 3009    2E 8F     LD L,&8F
 300B    CD F6 BB  CALL GrafLine
 300E    C9        RET
```

Program V

Incidentally if we'd loaded H, L, D and E separately our code would appear as in Program V.

Admittedly, the code is only two bytes longer in this version. However Program IV is easier to read, and by loading DE and HL directly conveys the idea of coordinates better. In machine code every improvement in clarity, no matter how slight, is an advantage.

Take a look at Program VI. It uses several routines we haven't encountered yet.

What we're doing here is to create a text window, then clear that window to PAPER 3, which is red, providing we're still in our default colours in Mode 1 (the way the Amstrad is at switch-on).

TextWin (&BB66) allows you to create a text window. H and D contain the two text column numbers that you want the window to be between, while L and E contain the rows. It also corrupts all our registers.

In the above case our window is between columns 10 (&OA) and 30 (&1E). and rows 10 (&OA) and 16 (&10). It will also be window #0, since we haven't specified any other stream.

We also use the routine SetPaper (&BB96) to set the text background colour. A must contain a legal ink number for the mode you're in, and both it and HL are corrupted by the routine.

```
address  hex code  aneamonics
 3000    CD 6C BB  CALL ClrText
 3003    26 0A     LD H,&0A
 3005    16 1E     LD D,&1E
 3007    2E 0A     LD L,&0A
 3009    1E 10     LD E,&10
 300B    CD 66 BB  CALL TextWin
 300E    3E 03     LD A,&03
 3010    CD 96 BB  CALL SetPaper
 3013    CD 6C BB  CALL ClrText
 3016    C9        RET
```

ProgramVI

Notice that we then call ClrText again to clear the window to the paper colour.

The Basic equivalent of the process would be:

```
10 CLS
20 WINDOW 10,30,10,16
30 PAPER 3
40 CLS
```

In Program VI I've loaded H,L and D,E separately — no 16 bit loads here! To prove just how good you are becoming at machine code, I want you to convert it so you load HL and DE with 16 bit loads. That is, get rid of those four single register loads.

And while you're at it, that window is an awkward size — when you use the Examine Code option in Hexer it won't quite fit will it? So you can adjust the window size as well.

So far we've concentrated on loading our register pairs with constants. Our instructions have been in the form:

LD rr,an

where rr is the register pair and mn is the two byte constant to be loaded, m the hi byte and n the lo byte.

There are lots more things we can do with the register pairs, but for the moment we'll learn how to use them to peek and poke into memory.

So far we've done our peeking and poking into memory 8 bits at a time via the A register.

The instructions were:

LD A,(pq)

and

LD (pq),A

Notice the brackets around the pq. These show that we're concerned with the *contents* of the memory location specified, pq.

Using register pairs, however, we can manage to peek and poke two bytes, that's 16 bits at a time — ideal for handling big numbers and addresses.

To poke, or more accurately, load in to memory, we can use instructions of the form:

LD (pq), rr

where pq is the memory location we want to poke into, p being the hi byte and q the lo and rr is one of the register pairs BC, DE, HL.

Table II gives the opcodes. For the first time we're dealing with two byte opcodes, so together with the address, an instruction such as:

LD (&3000), BC

will take four bytes in all to specify.

mnemonics	opcodes
LD (pq),BC	ED 43 q p
LD (pq),DE	ED 53 q p
LD (pq),HL	ED 63 q p
LD (pq),HL	22 q p

Table II: Opcodes for 16 bit "pokes"

You might have seen that there are two versions of:

LD (pq),HL

one of which has a single byte opcode — that's the one we'll use for preference.

Fine, we know the opcodes, but what exactly do they do? Well, since we're loading *two bytes* into memory from the register pair, we'll naturally have to load them into two separate memory locations.

If the instructions were:

LD (&3000),HL

the effect would be to load &3000 with the contents of L and &3001 (the next memory location up) with the contents of H.

Notice it goes into memory lo byte (L) followed by hi byte (H) — very convenient. To put it more formally,

LD (pq),rr

causes

(pq) <== rr lo byte

(pq+1) <== rr hi byte

where the arrows mean is *loaded with*.

Think about it! pq+1 is the next memory location after pq. Both pq and pq+1 are in brackets since it's their contents we're changing, not the number pq or anything weird!

A bit of practical work should make this clear:

```
address  hex code   mnemonics
 3000    21 34 12   LD HL,&1234
 3003    22 F8 2F   LD (&2FF8),HL
 3006    C9         RET
```

Program VII

Program VII simply loads register pair HL with &1234, and then pokes the contents of HL into the memory locations starting at &2FF8, Hexer's workspace.

Run the program and then examine memory from &2FF8. You should see that &34 (the contents of L) is stored in &2FF8 and &12 (the contents of H) is stored in &2FF9.

Ring the changes in the program by loading the other register pairs — BC,DE — with constants and then poking them into the workspace. As you'll see, poking with register pairs stores their contents in lo byte, hi byte order in two adjacent memory locations —

'Register pairs are ideal for handling large numbers and addresses'

the one specified and the next higher up in memory.

We can also peek memory using the register pairs:

LD HL, (&3000)

would place the contents of memory location &3000 in the L register and the contents of &3001 in the H register (notice the hi byte, lo byte order in memory yet again).

We have the brackets around the address to show that we're dealing with the *contents* of memory. After all, if we left them out we'd have:

LD HL,&3000

which is just a straightforward 16 bit load, which will load the *constant* &3000 into the HL registers. H will then contain &30 and L will contain &00:

LD HL, (&3000)

on the other hand is our peeking instruction. After this instruction, what's in H and L will depend on the contents of memory locations &3000 and &3001.

If you like, in Basic,

LD HL, &3000

would translate to:

L = &00

H = &30

whereas:

LD HL, (&3000)

would be:

L = PEEK (&3000)

H = PEEK (&3001)

Our general description of 16 bit "peeking" with register pairs BC, DE, HL is:

LD rr, (pq)

which causes:

rr lo <== (pq)

rr hi <== (pq+1)

Table III contains the opcodes. We've got four byte instructions again, save for the "duplicate" involving HL. The reason for the extra, shorter instruction involving HL is that this tends to be our favourite and most frequently used register pair, in much the same way as A is among our 8 bit registers.

mnemonics	opcodes
LD BC,(pq)	ED 4B q p
LD DE,(pq)	ED 5B q p
LD HL,(pq)	ED 5B q p
LD HL,(pq)	2A q p

Table III: 16 bit peeks with register pairs

At this stage in the game it is a little awkward giving you a decent example of a 16 bit peek — though believe me they're very useful instructions. Anyway Program VIII gives a rather contrived example.

```
address  hex code  mnemonics
 3000    2A 00 30  LD HL,(&3000)
 3003    22 F8 2F  LD (&2FF8),HL
 3006    C9        RET
```

Program VIII

All this does is to copy the first two bytes of our program (which starts at &3000) into Hexer's workspace at &2FF8.

LD HL, (&3000)

puts the contents of &3000 into L, and the contents of &3001 into H.

Our next instruction:

LD (&2FF8), HL

is the poke instruction we've already met and puts the contents of HL into workspace, lo byte first.

When you now examine the workspace (from &2FF8 onwards), you should find the first two bytes are:

2A 00

corresponding to the first two bytes of our program (stored at &3000 onwards).

Try rewriting the program using the other register pairs we've met to peek and poke with.

Well, that's plenty for this month. A final point — you can't combine any two single registers to give a register pair. There isn't a register pair HB, for instance, even if there's such a pencil!

And that word "combine" is important, too. The register pair HL, for example, isn't independent of changes you make to the single registers H and L. A change in either will affect the value of the two byte constant or address in HL.

10 LINERS

Ten liners is a new feature intended to contain short simple programs sent in by our readers. The routine can be anything you want — utilities, mathematical listings, graphics demonstrations. All are welcome here.

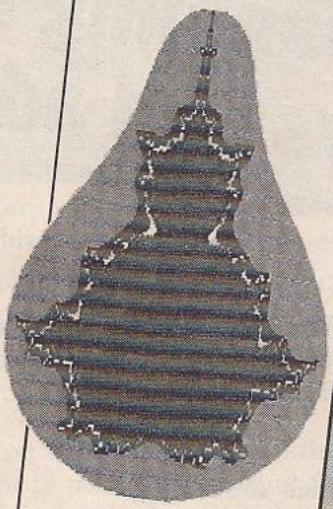
The only requirement is that they're no longer than 10 lines, so it's a real brain teaser.

Ten lines may not seem a lot, but it is surprising what you can do with them, plus a little imagination. Just to whet your appetite, here are some contributions from our editorial office.

So come on, get your thinking caps on. We'll pay $25 for the submission we consider the best, and $12.50 for any others we publish.

You'll stand a better chance of getting your masterpiece in print if you send it in on tape or disc, but make sure you include a stamped

DIY fractal design



FEATURED elswhere in the magazine this month is an article on fractals and the program to go with it allows you to create your own designs. When you input certain figures the result is a pattern known as the Mandelbrot set.

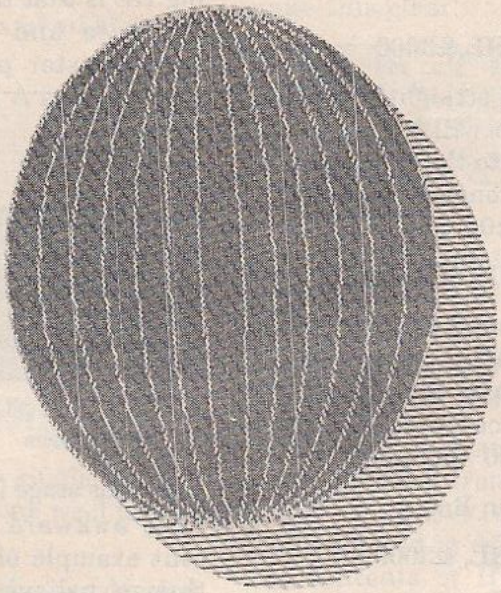
Here's Roland Wallilove's version of that particular little effect in just 10 lines

```
10 REM Mandelbrot Set
20 REM By R.A.Waddilove
30 MODE 0:INK 13,0:ORIGIN 474,200
40 FOR i=-2 TO 0.7 STEP 2.7/160
50 FOR j=-1.4 TO 1.4 STEP 2.8/200
60 n%=-1:x=0:y=0
70 x2=x*x:y2=y*y:y=2*x*y+j:x=x2-y2+i:n%=
n%+1:IF n%<48 AND x2+y2<4 GOTO 70
80 PLOT i*237.04,j*142.857,1+n%\4
90 NEXT:NEXT
100 SAVE "screen",b,&C000,&4000
```

Ball control

HERE Ian Sharpe has provided the biggest bongiest bouncing ball (complete with shadow) that you've ever seen on an Amstrad.

```
10 DEFINT a-z:ENV 1,10,-1,2:ENT 1,10,-60
,2:r=160:s=r^2:DEG:PLOT 1000,1000,1
20 MODE 1:BORDER 26:INK 0,26:INK 1,0:INK
 2,6:INK 3,15:ORIGIN 200,200
30 z=4:GOSUB 90:PLOT 1000,1000,2:ORIGIN
160,238:z=2:GOSUB 90:PLOT 1000,1000,3
40 FOR i=16 TO 160 STEP 24:GOSUB 100:NEX
T:xd=1:yd=1:x=35:y=28
50 WHILE 1:OUT &BC00,2:CALL &BD19:OUT &B
D00,x:OUT &BC00,7:OUT &BD00,y
60 y=y+yd:IF y=21 OR y=35 THEN yd=-yd:z=
RND*800+600:SOUND 129,z,0,15,1,1:SOUND 1
32,z+10,0,15,1,1:GOTO 60
70 x=x+xd:IF x=22 OR x=50 THEN xd=-xd:SO
UND 129,z,0,15,1,1:SOUND 132,z+10,0,15,1
,1:GOTO 70
80 WEND
90 FOR y=0 TO r STEP z:x=SQR(s-y*y):MOVE
 x,y:DRAWR -2*x,0:MOVE x,-y:DRAWR -2*x,0
:NEXT:RETURN
100 MOVE i,0:FOR a=0 TO 360 STEP 10:DRAW
 i*COS(a),r*SIN(a):NEXT:RETURN
```



By altering the values in certain registers of the video display control chip, the entire screen window is made to move about. With the same background ink and border, the ball appears to rebound from the sides of the tube.

Type it in, and amaze you friends.

Increasing workspace

MANY expansion roms have a command to switch themselves and other roms off. This is useful if commercial softwaree wants to use the rom's workspace and the rom itself is not required.

If you aren't lucky enough to have a rom with such a command this program from Ian will do the job on any CPC. With the machine code installed enter:

CALL &e000,a,b,c

where a,b,c and so on are roms you want to leave on, for instance rom number seven — the disc rom. Basic will be reinitialised with only the selected roms in action.

Be careful, as any program in memory will be lost. CALL &A000 by itself will leave no active expansion roms at all and the sme HIMEM as a bare CPC464.

```
10 MODE 2:MEMORY &9FFF:FOR i=&A000 TO &A
046:READ c$
20 c=VAL("&"+c$):POKE i,c:s=s+c:NEXT
30 IF s<>7962 THEN PRINT"ERROR!!!!":LIST
 50-
40 PRINT"call &a000,a,b,c ... etc to lea
ve selected rom numbers ON"
50 DATA 21,46,A0,77,B7,CA,14,A0,47,23,DD
,7E,0,77,DD,23,DD,23,10,F5,E,0,21,1C
60 DATA A0,CD,16,BD,3E,C9,32,CB,BC,3A,46
,A0,B7,CA,3F,A0,47,DD,21,47,A0,11,40
70 DATA 0,21,7F,AB,C5,DD,4E,0,CD,CE,BC,C
1,DD,23,10,F4,DF,42,A0,6,C0,0,0,0
```

Faster disc drives

IAN'S third program will speed your disc drive up by 5 to 12 per cent. A short machine code routine peeks into the disc rom to find where the parameters controlling the step rate, motor start up, and switch-off delays are stored.

The default values are 12 miliseconds, 50*1/50 second and 250*1/60 second respectively. You can alter these with the CP/M 2.2. Stetuputility and now with this, from Amsodo.

Here the values are set to 11,40 and 200 which are still reliable. Some drives are happy with a step rate of 10 which will improve speed by nearly 20 per cent on some files.

```
10 DEFINT a-z:FOR a=&50 TO &64
20 READ v$:POKE a,VAL("&"+v$):NEXT
30 d=0:CALL &50,,d
40 POKE d,40:POKE d+2,200:POKE d+6,11
50 DATA e,7,cd,f,b9,ed,5b,0e,c6,dd,6e,0,
dd,66,1,73,23,72,c3,f,b9
```


Business Contents

TYPE SANS TEARS

Articles in recent issues of *Computing with the Amstrad* showed you how it is entirely possible to typeset from an Ascii file generated by LocoScript. However, this approach has one disadvantage — the creation of an Ascii file strips from the text all the control characters originally put there to indicate the different typestyles to be used.

As a result, it's necessary to send to the typesetters, along with the text on disc, hard copy of the material to indicate where these changes should occur.

Such a procedure means that the typesetting company has to "intervene" to make those changes, which takes time, and therefore costs money. If you are producing typesetting work on the PCW regularly and you're prepared to go to a typesetting company equipped with the relevant software — TypeSet from Wordsmiths — you can short circuit that intervention.

You can produce a file which can be typeset directly, with you specifying the faces, sizes, position of text, in fact all the parameters that affect the final appearance.

All this for the cost of one piece of Amstrad software — WordStar. It's available from many dealers who supply the PCW system, and generally costs around $175.

It is the single most popular word processing program in the world, and is available for virtually all MSDos and CP/M computers, so the process described here is equally applicable to the Amstrad PC1512. In fact, once you've set up the program on either machine, the procedure for using it is identical.

Having files typeset commercially can pose many tricky problems. Richard Elen explains the WordStar solution

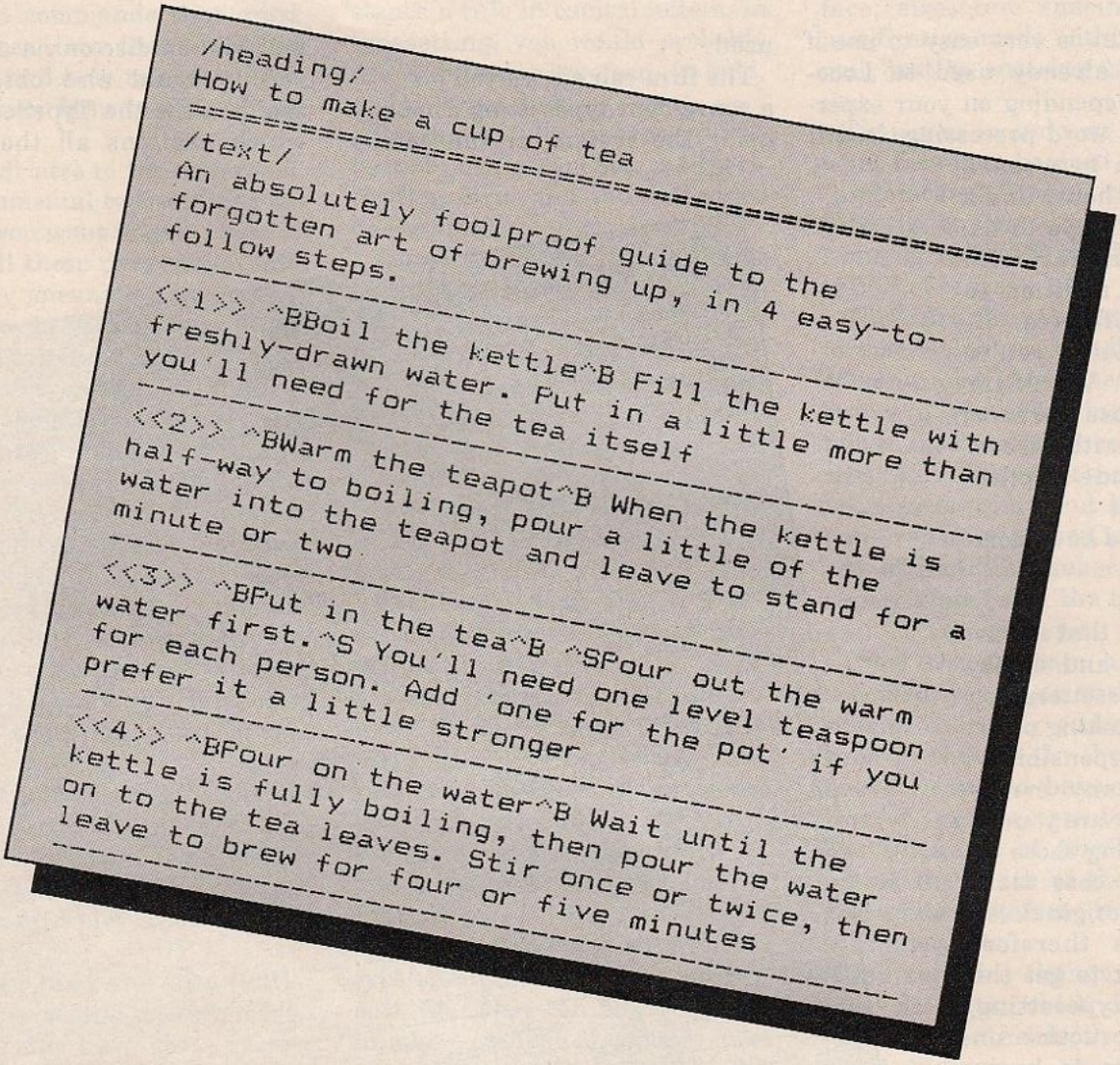
WordStar for the PCW is a slimline version called Pocket WordStar, leaving out some of the more sophisticated functions available on the MSDos version. But it does include mail merge facilities, and you can also obtain SpellStar, the spelling checker that goes with the package.

Pocket WordStar is entirely capable of utilising all the functions of the PCW standard printer (with the exception of true proportional spacing in fully justified text, which is normally handled by introducing extra space characters between words), or any Epson-compatible.

It will also handle a number of daisywheel and other printers if you use the supplied Install program to configure the system before you make up your work disc.

LocoScript files *can* be used in a similar fashion to that discussed here, but Wordsmiths recommend the use of WordStar wherever possible, as it is more amenable to direct setting. This is partially because of the way LocoScript stores typeface specification data in the file.

New World, which is compatible with WordStar files and is available for the Amstrad, can also be used for typesetting, and it has certain advantages, for example, showing underscored and bold areas of text. However, it has been known to suffer problems with large files on full discs.



```
/heading/
How to make a cup of tea
==========================================
/text/
An absolutely foolproof guide to the
forgotten art of brewing up, in 4 easy-to-
follow steps.
------------------------------------------
<<1>> ^BBoil the kettle^B Fill the kettle with
freshly-drawn water. Put in a little more than
you'll need for the tea itself
------------------------------------------
<<2>> ^BWarm the teapot^B When the kettle is
half-way to boiling, pour a little of the
water into the teapot and leave to stand for a
minute or two
------------------------------------------
<<3>> ^BPut in the tea^B ^SPour out the warm
water first.^S You'll need one level teaspoon
for each person. Add 'one for the pot' if you
prefer it a little stronger
------------------------------------------
<<4>> ^BPour on the water^B Wait until the
kettle is fully boiling, then pour the water
on to the tea leaves. Stir once or twice, then
leave to brew for four or five minutes
------------------------------------------
```

Figure I: The word-processed text including typesetting commands

WordStar is available on so many machines that the setter doesn't need to own an Amstrad to use a file generated on one. It's a simple matter to convert the file from one disc format to another.

In WordStar a file contains the words of the text, with the highbit set on the last Ascii character of every word. Changes of typestyle are embedded in the text by means of control codes, and are often toggles. For instance, ^S in the text (meaning Control+S) indicates a change to underscore mode (usually interpreted as italic by the typesetter). The same character indicates a return from underscore mode to whatever the style was before.

It is entered into the text by pressing f6, or Alt+P then S. The latter is the general form of entering a print command in WordStar — Alt+P followed by another letter, such as S for underscore, B for bold, and so on.

The multiple spaces which pad out the text for justification are ignored by the setter, as are line spacings except in special cases.

How to make a cup of tea

An absolutely foolproof guide to the forgotten art of brewing up, in 4 easy-to-follow steps.

1 **Boil the kettle** Fill the kettle with freshly-drawn water. Put in a little more than you'll need for the tea itself

2 **Warm the teapot** B When the kettle is half-way to boiling, pour a little of the water into the teapot and leave to stand for a minute or two

3 **Put in the tea** *Pour out the warm water first.* You'll need one level teaspoon for each person. Add 'one for the pot' if you prefer it a little stronger

4 **Pour on the water** Wait until the kettle is fully boiling, then pour the water on to the tea leaves. Stir once or twice, then leave to brew for four or five minutes

Figure II: The typeset result

WordStar is very easy to use if you are already used to LocoScript. Depending on your expertise with word processing, it will take you between a couple of hours and a couple of days to become familiar with the new addition to your software library. Once you've learned WordStar you can use the same program with ease on a multitude of other machines.

It should be remembered,

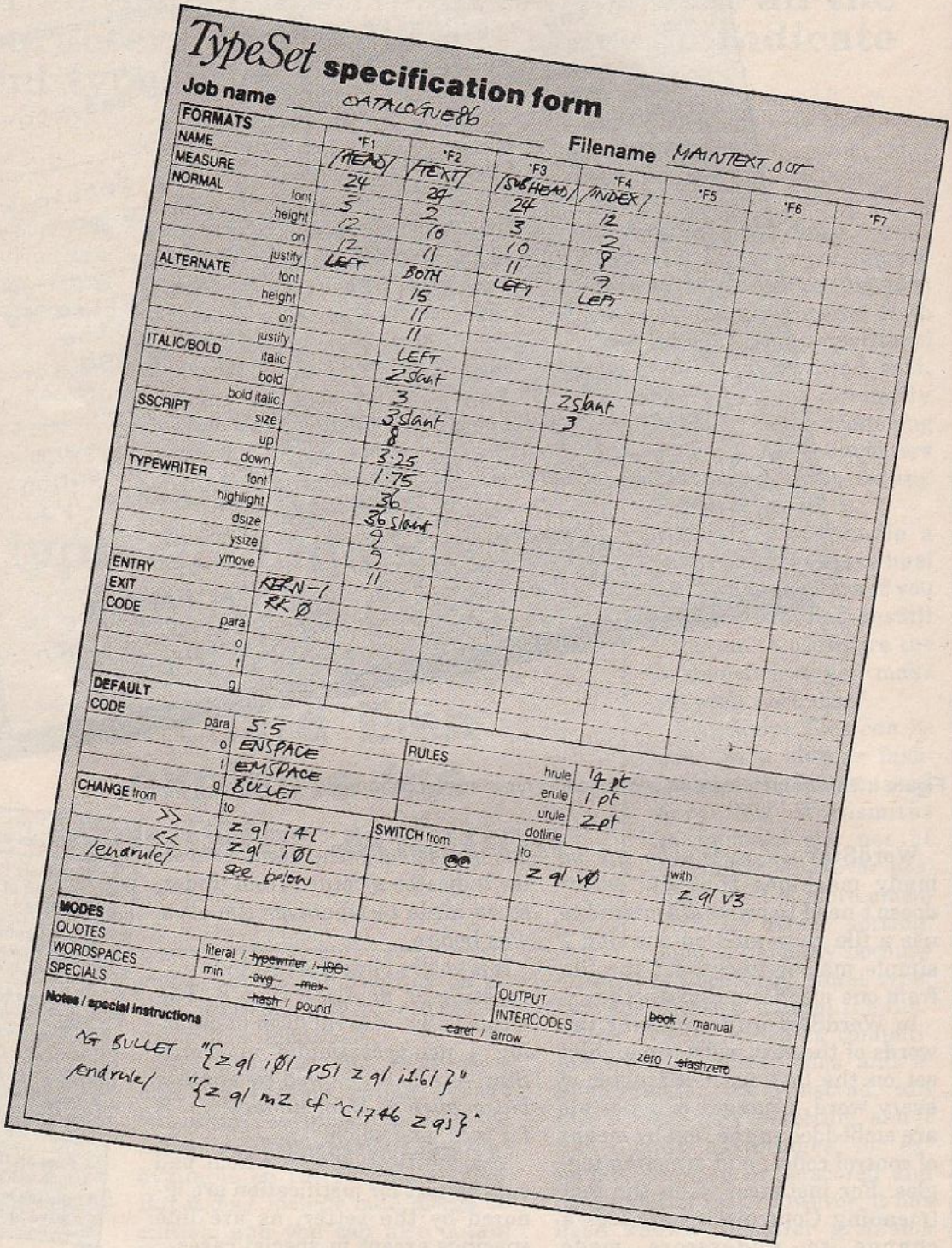
of course, that in saving time and money at the typesetters, you are also taking on an added responsibility, in that any errors that occur are yours.

Correcting those errors can cost more than the original setting. It's therefore important to get the hang of typesetting, and to practice on small jobs to begin with to avoid expensive mistakes.

To my knowledge, there is currently only one fully implemented direct conversion typesetting system designed to work with WordStar files, and that's TypeSet from Wordsmiths Typesetting. Wordsmiths was one of the originators of typesetting from computers in the UK, and its system is quite widely used.

The firm can advise you on a convenient typesetting company using the system, or can accept material on disc or via a modem.

You should also obtain from Wordsmiths the TypeSet booklet, which explains all the instruc-



tions and commands that must be inserted into the WordStar text file, and how to construct a specification form to accompany your file.

This indicates to the setters all the fundamental typesetting parameters you want to use.

You call those parameters into action by means of embedded characters in your file.

Also you get a chart demonstrating the various faces available from your typesetters.

There are fundamental differences between the normal output from a word processor or a typewriter, and typeset text in a book or a magazine. For a start, many word processors, including WordStar, and most typewriters, use monospaced typefaces in which every character takes up the same width on the paper.

Typesetting, on the other hand, uses proportional spacing. Examine the text on this page and compare the amount of space taken up by an i and an m, for instance.

A word processor also limits the number of type styles you can use. You may have some variation on a modern dot matrix printer, with about two or three basic faces, and the capability to use underlining, boldface, expanded and perhaps italic versions. You may even be able to use a couple of different sizes.

In typesetting there are literally thousands of faces at your disposal, and a wide range of sizes. That welter of possibilities can be confusing to begin with, as can the apparent complexity of the way in which typesetting information is specified.

In addition, some things that work on a typewriter or word processor don't necessarily look that good when typeset, for instance a title in capital letters. In typesetting, you would probably use a different typeface, in bold, and a larger size.

Fundamentally, there are two parts to using TypeSet, the specification form and the embedded commands.

Figure 1 shows just how simple typesetting from WordStar can be. It is taken from the Typeset booklet and describes a simple, yet essential process. It shows some of the commands and special character combinations that instruct the typesetter to use different specifications. Alongside in Figure 11, is an example of how it might look when typeset.

You'll notice from the word processor file that two types of commands are embedded in it. There are the regular WordStar ones, like ^S*Sitalic*^S and ^B**bold**^B, which are used in exactly the same way by TypeSet as they are by WordStar. But there are also some other commands which relate to parameters determined in the specification sheet.

Up to seven completely different type specifications can be used in each file. They can be called by a number embedded in the text (like *F1 and *F2) but it's a good deal more understandable to use a name such as /heading/ and /text/. Each format covers every aspect of typesetting.

Within each format, you can determine:

- The line-length in picas (1/6in) and points (1/72in).
- The name of the normal font — used when a new format is called, or if the control character ^N is placed in the text.
- The type size in points.
- The line spacing.
- The justification mode (left, right, centre or both).
- Alternate font specs, which enable you to call up a different face, size, line spacing and/or justification mode within the format, just by entering ^A in your text.

Then you include on the form what face is to be used when the "underscore" (italic) instruction (^S) is encountered, and the same for bold (^B) and bold italic (^B^S of course).

There are also other aspects of a format which can handle the use of non-proportional faces, for instance if you are using the system to set a computer manual and wish to represent on-screen responses or command lines in a non-proportional face, and methods of handling subscripts and superscripts (with the toggles ^V and ^T respectively).

Once the formats have been specified, you go on to define any special character codes or rules used in the file. In the example above we would have defined a line of = characters to produce one thickness of line (a "rule") across the width of the column, and a line of hyphens to represent a thinner rule.

You can also change the standard defaults, which determine, for example, the depth of a paragraph indent.

Then you come to special instructions which must be handled with care, as they represent a good deal of typesetting "intercode" (a programming language used by the typesetting system) in just a couple of unique character combinations. In the example above we would have defined >> to mean "indent the text by so many picas", while << would mean "cancel the previous indent".

Special instructions can do more or less anything, and are best discussed with the typesetter first. For instance, you may want to set a "bullet" list, with text

indented from the left, and each paragraph preceded by a blob. A simple method would be to define a special instruction, or tag — let's call it [blob] — to do the work for you.

In intercode, [blob] might translate into something quite bizarre, like {z ql i01 p51 z ql i 1.61).••• You may not need to know that, but your typesetter will need to know what you can expect to happen. (Intercode can in fact be entered directly in the text if you wish, but it is understandably complex, although it may be something you wish to use eventually.)

Finally, you detail the various modes of operation you require for certain aspects of the conversion. A good example here is quotes. You can embed them in your text in three ways, and must indicate which your file uses.

Most word processors do not offer true opening and closing quotes and double quotes, so conventions must be established as to what you put in your file to generate them.

Wordsmiths recommend you use the grave accent to open quotes, and the standard apostrophe to close them, using them twice for double-quotes. Alternatively, you can use "typewriter" mode, with " or ' at each end of the quoted word or phrase.

TypeSet deals with them "intelligently", and converts them to the correct opening or closing characters, but there are difficulties with abbreviations like '86 and 'phone.

The third option is the ISO standard, which requires entry in the file of " to open quotes and ' to close them (just to confuse you). You use them twice for double quotes.

Another conversion specification is that of paragraph breaks. The file can contain blank lines between paragraphs (journalistic style), or you can use WordStar's standard paragraph indent to generate a typeset paragraph indentation.

Assuming you're using book mode (for normal setting) rather than manual mode (for setting documentation), you can generate fractions like this: '33^Y1/3^Y%' will come out as '33 1/3%'.

Special conventions exist in TypeSet to cover the indentation of whole paragraphs. At the front of the file you determine the type of indent you want — left, right or both — and how many picas the indent is to be.

Typically, you might enter *IL*12 to indent left by two picas.

But nothing actually happens until you call the indent by including the characters -> (hyphen followed by greater-than, forming a right arrow) in the file. You cancel the indent by entering <- and the indent type and amount can be changed at any time with further *I instructions.

If you intend to print out your file before sending it for setting — a good idea, as you should check, for instance, that toggles have been turned on and off properly, otherwise you may end up with half your text in bold italic because you used ^B at the start of a phrase and ^S by mistake at the end — you may need to be able to display the typesetting instructions.

Embedded control characters will usually cause an effect, rather than being visible, but you can make them print by turning off WordStar's formatting during the print setup sequence.

Tags and format names can be suppressed in normal printouts by preceding them with two dots and placing them on their own line — this is WordStar's method of indicating non-printing comments.

Again, they can be seen by defeating the page formatting.

On the subject of page formatting, remember that WordStar's centring command does not work in typesetting, you use *JC (justify centre) instead.

But forced page breaks in WordStar (using the command .pa) do have an effect in that they cause a half-inch gap to be left in the setting, making it easier to identify blocks of text when it comes to laying out the setting and pasting it up as camera-ready artwork.

Alternatively, .pa can be used with a pagination program at the setters to force the end of a page, as in WordStar, which can also perform the setting of running heads and page numbers.

The fundamental point about TypeSet is that it is a method of conversion at the typesetters — you need no extra software (other than WordStar itself) or hardware to include control characters (all of which, incidentally, are entered into the file as print commands, by pressing Alt+P followed by the letter).

So, to produce ^Y, you type Alt+PY) ••• •• or names, formats and special instructions. You do, however, need to learn what they do. Luckily, this is relatively easy.

Once you have grapsed the technique — and most importantly, once you begin to think typographically — you are well on the way to being able to orgininate text files complete with all the information needed to typeset them directly, to your specifications.

It's cheaper, it gives you more control, and the results are excellent.

JUST YOUR TYPE?

Niels Reynolds inspects a package which claims to

Supertype is a package offering PCW users a choice of eight typefaces which may be used in either LocoScript or CP/M. As such, it could prove useful for a wide range of applications.

To set up Supertype you have to copy the relevant parts of CP/M and the program disc on to a common working disc. The procedure is relatively straightforward, though the instructions may be slightly confusing to users new to such things.

This mild rebuke is aimed at the accompanying booklet, as there must be many PCW owners who have come to the machine via the typewriter rather than the home micro and who have little or no knowledge of CP/M.

Digita has set the instruction booklet using a dot matrix printer, but strangely not using one of their own types.

On loading the working disc you are offered a choice of the eight type faces, or fonts as they are known in the trade. They are Business1-4 (all sans serif), Old English, Outline, Vaudeville (a style with a 1920s feel to it) and Stencil.

Having made your selection you install the type face on a disc, specifying whether it is LocoScript or CP/M — a simple enough procedure.

After selecting, say, Stencil, and choosing a LocoScript disc, any

Program: Supertype
Price: $49.95

document on the disc will now print out in Stencil. However, having selected a particular font, the instructions don't tell you how to reselect the original LocoScript typeface.

The pack reads: "Simply install Supertype on your current work disc and choose from any of these eight fonts". This suggested to me that, while working on a document within LocoScript, any or all of the styles can be selected.

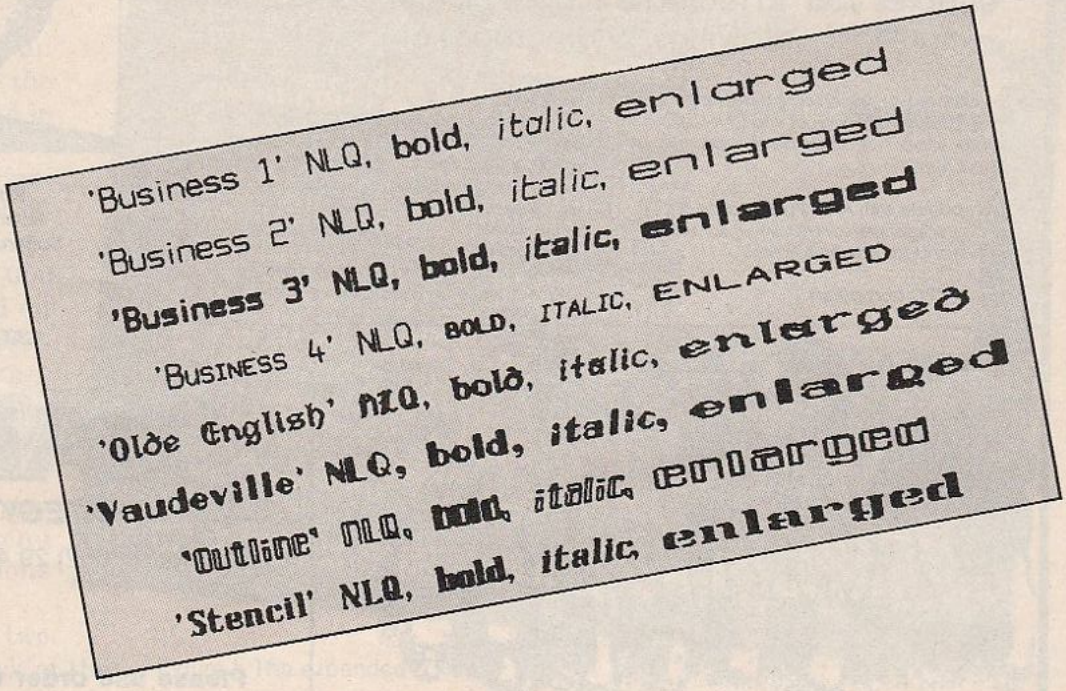
This is not the case, only one style can be used at any one time and cannot be mixed within the disc, let alone a specific document.

This impression is further backed by an illustration on the packaging that shows a printer rolling out a printed sheet with all eight type faces displayed. This is just not possible.

To change type faces you must finish editing, replace the work disc with Supertype, call up the required font and install it on the work disc. Once familiar with the procedure this takes three or four minutes, which though not a long time in itself, can be annoying.

I have said that there is no explanation in the manual as to how the original LocoScript style can be recalled. If there are only a few doucments on the affected disc, they can be moved (f4) to another disc through drive M.

If you feel this is too risky, or there are too many documents to move, there is another way. Return to the management screen with the Supertype font installed



and call up hidden files (f8).

Erase (f6) then file MATRIX.STD in group A. Then, changing to the original LocoScript system disc, or preferably a backup if you have one, copy (f3) file MATRIX.STD to drive M, then move it to group A of the previous work disc.

Digita warns that the system is designed to work under proportional spacing. Too true. Some of the letter spacing in other pitch sizes is awful, especially those involving i and I.

It is claimed that all printer modes are supported — Enlarged, bold emphasised, italic, proportional and so on. Yet pitch sizes 15 and 17 are not supported and, because of the letter spacing problems, pitch sizes 12 and 10 can become awkward to read.

Although some fonts are worse than others in this respect, to my mind that makes them unusable in those particular pitch sizes. However, there are none of these problems if proportional spacing is used — even if italicised or enlarged — although some letters may fill in when used in bold.

Despite these compalints, I liked Supertype and could imagine using it for various applications — club newsletters, college magazines, whatever.

I must admit for day-to-day use I would stick to LocoScript, but would enjoy having Supertype around just for the choice.

Digita's packaging and advertisements suggest that Supertype is rather more versatile and simpler to use than it acutally is, and having read all the blurb, my expectations were perhaps set too high. This is a shame, as Supertype has a lot to offer for the price. The additional hype is totally unnecessary.

I think people may well buy this package on the strength of Digita's rather loose sales pitch and could possibly be disappointed. I for one would happily buy Supertype on its own merits.

Digita has just announced an upgrade to this package, which as well as including several pitch, style and proportional spacing improvements, makes the software compatible with LocoSpell and LocoMail, and provides a redesigned Business 1 font for use with SuperCalc.

HELPING YOU TO SPELL IT ~~BITE~~ W~~RITE~~ RIGHT

Unless you're an expert proofreader and a perfect typist, you'll certainly benefit from using a spelling checker. LocoSpell is the official, Amstrad-approved checker for LocoScript.

Michael Sterne reviews a spell checker with the

To use it you must prepare a start-of-day disc and the method is fully described in the clear and comprehensive manual.

Your start-of-day disc, if you are using the PCW8256, will have to contain the LocoScript software, your PHRASES.STD and TEMPLATE.STD files, the LocoSpell software and the 32,000 word dictionary.

Everything you need is provided and takes up the whole of a 3in disc. Indeed, you may not have room for all your templates. With an 8512, or a second drive, the dictionary can be on a disc in drive B.

When you boot, the dictionary, whether on the start-of-day disc or in drive B, is copied to drive M. The 77,000 word dictionary, supplied on the reverse side of the software disc and requiring 158K, is only available for use in this way with the 8512.

You will find it easy to use LocoSpell. Pressing f7 provides an expanded menu with the additional options shown in Figure I.

Choose either of the first two options for a spelling check of the whole or part (a very useful facility) of the document currently being edited and displayed on the screen.

As usual, you can choose the option either by using the cursor keys or by typing the code represented by the capital letter.

When the program encounters a word it doesn't recognise, the menu shown in Figure II is displayed, showing the word and a suggested replacement.

This feature applies most frequently to the correction of typing errors or minor mistakes where the word typed is reasonably similar to the correct word: It is a considerable convenience.

Marking the word as correct places the code "SiC" in contact with it so that the checker will accept it, even if it is not in the dictionary. You can also place the code from the set menu or by using the set key and the alternative "SC".

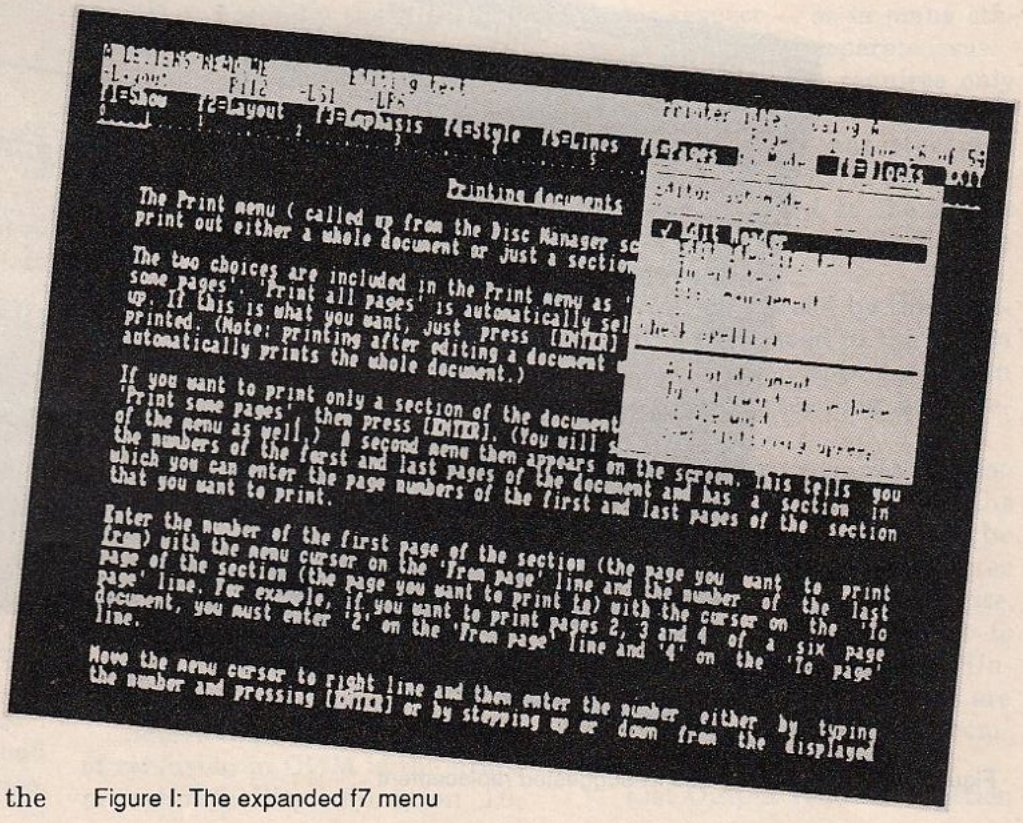


Figure I: The expanded f7 menu

Consulting the dictionary accesses 18 words at a time — you move the menu to the appropriate section by entering a word or some initial letters at the head of the menu.

You can look up a word while you are editing by typing all or part of it and choosing the single word option from the Modes menu. A single key-press then inserts the word.

The program has a facility for creating and amending user dictionaries. You choose the final option to store an unrecognised word in a specially created dictionary from which it will be recognised from then on: It is not saved permanently until the completion of the check.

The final menu gives a word count, an analysis of the spelling check and the option of updating the user dictionary.

You can amend user dictionaries but not system ones. And although you can add a user dictionary to a system dictionary you can then no longer amend it.

All user dictionaries use the same file name and extension so that the only way of distinguishing between them is by their location in different groups — which can cause complications.

As the manual indicates, by recommending an upgrade, the 8256 is not really suitable for LocoSpell. The software uses 34k of drive M and the dictionary another 68k so that "disc/drive full" messages abound and you will have to erase and move files in many cases to get LocoSpell to work.

The major advantage of a resident spelling checker is simplicity and with LocoSpell the document is checked on the screen with three keystrokes.

It also has some very convenient features, but shares with LocoScript the problem of slow scrolling (and is excruciatingly slow using the dictionary on disc).

I found with documents longer than about 15k that my CP/M-based spelling checker was quicker, even taking into account the time needed to boot CP/M and the program. The time spent in erasing files to free drive M would accentuate this.

If you have a PCW8256, LocoSpell is probably not the most suitable spelling checker for you unless you're planning an upgrade. With the 8512 you'll find it easy to use and well worth buying, though more expensive than some.

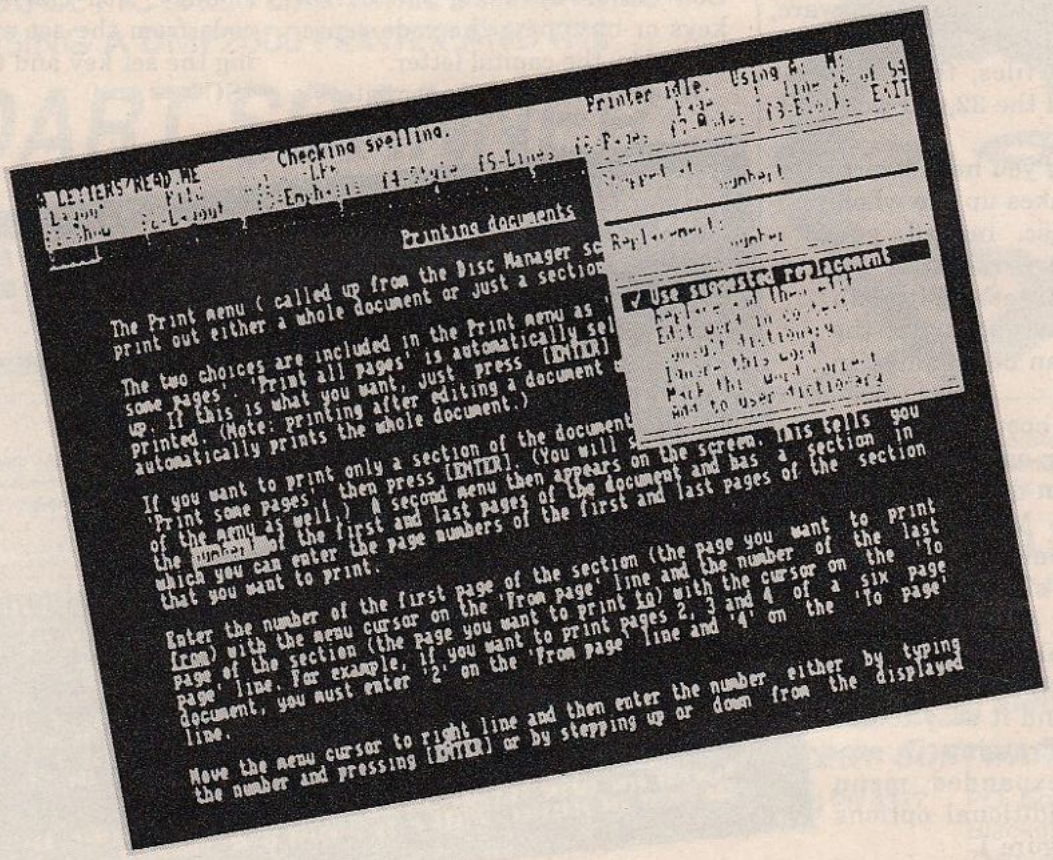


Figure II: The misspelling and its suggested replacement

number 5 in register C and Ascii value for the character is sent to the logical LST: device — normally the Amstrad's Centronics printer port. Again no checks are made for keyboard input.

Direct Console I/O is where things start getting a bit complicated. This function can be used to let us check the keyboard to see whether a character has been typed, intput a character from the keyboard or print one of the screen. In all cases we must call it with function number 6 in register C.

If we wish to print a character on the screen we must also put the Ascii code for the character into register E, just as for the normal Console Output function. However this one does not check for pause or abort commands from the keyboard.

To read from the keyboard we must call the function with the value &FFin register E. It then checks to see whether a character has been typed on the keyboard, and if so it returns it in register A, just like the normal Console input.

However if there is not a character ready it also returns immediately, but this time has the value 00 — an Ascii null character — in register A. Unlike the normal Console Input this does not wait until a character has been typed before returning.

This function is extremely useful if we wish to check for input occasionally — for example, an abort command — while doing something else. If we used the normal Console Input for this our main program would stop until someone pressed a key.

Get IOBYTE requires only the function number in register C, and returns the current value of the OIBYTE bit field in register A. This is a particularly useless function as the IOBYTE is always found at address 3 of the SPA in any CP/M system we might as well just get it direct with an instruction such as:

```
*
```

which is all the BDOS does anyway, without going through the rigmarole of a function call. The fuction was originally provided for compatability with early versions of CP/M which did not support the IOBYTE concept, but nowadays it is redundant.

Set IOBYTE is the companion function to the previous one, and is equally useless. Again it can be replaced simply by a single instruction such as:

```
ld (0003h),e
```

If you really do want to use it you need function number 8 in register C and the new pattern for the IOBYTE in register E. It returns nothing.

Print String is extremely useful and requires function number 9 in register C and the memory address of a string of characters in register pair DE.

It takes a string of Ascii characters starting at address DE and terminated with a $ character, and prints the string on the logical CON: device, which is normally the Amstrad's screen.

As it works internally by calling Concole Output for each character to be printed it obeys the same rules— typing Ctrl+S will pause the output, Ctrl+C will then abort and any other key will restart the output. If the Ctrl+P toggle has previously been set the string is also sent to the LST: device.

Read Console Buffer requires function number 10 in register C and the memory address in register pair DE of an input buffer. The layout of this buffer is shown in Figure 1.

This function lets us unput a string of characters from the keyboard, complete with line editing facilities like those available on CCP command lines. on an Amstrad these are as shown in Table 1.

We must have previously initialised the first byte of the buffer with a value for its masimum size, which is the maximum number of characters we are going to allow the person running the program to type at the keyboard before we cut him or her off. This can be any number between 1 and 255.

The function will return either when the buffer has filled to the size we have specified or before that if Enter in pressed to show end of input. When the function returns CP/M has worked out the length of the string which was typed in and has put this number into the second byte of the buffer.

Get Console Status requires function number 11 in register C, and returns a value in register A. If the value is 00 no character has been typed since the last character was read. If the value is &FF a new character has been typed and is waiting to be read.

The function is another pretty useless one. If you remember Direct Console I/O will scan the keyboard and return not only the status but also the character if one is waiting. This is quicker and easier than calling this function to check the status, and then having to check the value reutrned and call another function to actually read the character if one is ready.

Return Version Number requires function number 12 in register C, and returns values in registers H and L which specify

the version of CP/M present on the machine.

This allows programs to work out exactly what facilities and functions are available to them on different machines which may be running different versions of CP/M.

Older versions such as 1.4 and 2.0 do not have as many BDOS functions as we so with version 2.2, and some common functions work slightly differently.

Similarly newer or more powerful versions such as 3.0 (CP/M Plus) and MP/M (the multi-user system version of CP/M) can do many things which we can't.

If register H returns with a value of 01, then we are in an MP/M system. Otherwise register H will contain 00 to signify normal CP/M. In this case register L will contain 00 for versions before 2.0, and values of &20 or higher for newer versions. So our CP/M version 2.2 will return a value of 00 in register H together with &22 in the L register.

Ctrl+C	Aborts and warm boots if at start of line.
Ctrl+P	Printer echo toggle – typing this echoes everything printed on the screen from then on to the LST: device. Typing Ctrl+P again turns this off.
Ctrl+R	Retypes the line.
Ctrl+X	Deletes all characters already typed and repositions cursor to start of line to let you start again.
Del	Deletes last character typed.
Enter	Terminates input and returns the string to our program.

Table 1: Read Console Buffer line editing facilities

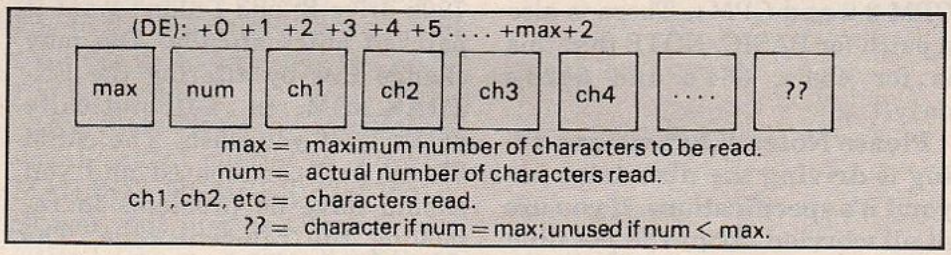


Figure 1: Console buffer structure

44 Tracks under CP/M

As I was re-reading last month's column about how the disc stores information, I connected it with a utility that I saw for sale in one of the U.K. magazines that increased the 3 inch drives' disc space by quite a bit. After some preliminary investigation it seemed that there was no reason for us not to have the 44 tracks talked about. So, this month we have a set of utilities that enable owners of the CPC range of computers to use up to 44 tracks under CPM 2.2 and CPM+. There is also a patch for BASIC. NOTE that this is for 464's, 664's and 6128's only!!

Please Note:- What we are doing is driving the disc drive beyond it's specifications. If you are at all worried about the possible effects of these modifications then don't do it!!!!

*****You have been warned*****

To allay your fears, I have been using this for about 8 weeks (or I will have by the time you read this!) and I have had no bad vibrations from the drive. The first step in the trek towards bigger discs for amsters is the program FMTEXTRA.COM. This is a 44 track formatter that I wrote for my own use about 18 months ago. It was then just a regular formatter, but I have altered it to do the extra tracks that we need. It is hereby placed in the public domain for all amsters to use as are all the programs that I have written for this disc. OK, thats that out of the way. Now you are free to use the program, here's what you do:- First get a disc that is either new, or near new as fluff tends to collect around the hub of the disc and as we are now going into that area, we must make sure that our chances of success are the best. OK, run the formatter and follow the prompts. It does not matter whether you chose system or data format, you will have an extra 18k per disc. The next step is to run one of the patch programs (either the BASIC or the CPM one). The EXTRAK.COM program automatically detects which operating system (CPM+ or CPM2.2) is in use, and patches it accordingly. If you have two drives attached, then you are asked if you want to patch B: as well. I did this because I have a 720k drive B: and I didn't want it patched automatically. You may change this by altering the EXTRAK.ASM to automatically patch B: if you wish. The .ASM file is well commented and you should have no problems in removing the relevant code. The patch stays put under CPM until you do a cold reboot. Under basic it stays until you do a reset.

OK, thats all for that part. Are any of you ELITISTS? No, not snobs but players of ELITE? I am, and I got tired of trying to remember where I could get those elusive extra credits for my cargo, so I wrote a short program that gives me a form to fill in for each planet that I visit. It lets me record all the details of the market prices on a particular planet and is quite handy in helping you decide what to carry. That program is ELITLIST.BAS. Use it, it helps. The other .BAS program is one that prints out a form for helping in a disassembly of a program. I find it helpful when I am doing disassemblies of short programs that would just be a chore under DASM.

The next programs are the contents of CPMUG Volume number 22 and is the original game that started all the space simulation games. Yep, you guessed it, it's STARTREK the game!! The original files from this compilation are there along with a modified version of STRTRK/2.ASC that will run under CPM+ only, due to the space restrictions under MBASIC running under CPM2.2.

If anybody would like to try to modify BIGTREK to fit into the available space go right ahead. But don't forget to put the modified version back into the public domain for everybody to use.

Now, just for a change of pace, we are going to tell all you novices out there what you do when you receive your public domain disc as there has been a number of inquiries to the magazine along the lines of 'thanks for the disc, but it won't run'. To understand what to do, we must go back to basics.

CP/M 2.2 is a generic operating system that will run on most Z80 based hardware that has at least 16k bytes of RAM and a disc drive. It is supplied to the customer (in our case the customer is AMSTRAD) as three separate parts: the CCP or console command processor, which is the bit that gets your commands from the keyboard and passes them along to the appropriate routines. The BDOS or basic disc operating system that handles all file and disc related requests from the user and finally the BIOS which is the basic input output system or the bit that gets it's hands dirty actually talking to the attached disc drive or printer or any other object that can be legally hung on the system. Now, when you type :CPM from basic a few things happen. Firstly, all RAM is cleared and the 'cold boot' routine sets up the BIOS to minimum working state and then loads in a boot program from the boot sector on the CPM SYSTEM DISC. For this to work

properly, there must be a disc that has been formatted in SYSTEM format in drive A:. Otherwise you will get the 'failed to load CPM — re-try, ignore, cancel' message. Once the boot program is active it sets up CPM to where it is able to run, but then loads a configuration section off the disc that will display a signon message, do keyboard translations etc. If the configuration sector is not there you get another 'failed to load...' etc. If you have formatted the disc as a SYSTEM disc, this configuration sector will be written for you, without you having to worry about it. You may alter the configuration sector by using the SETUP.COM utility that is supplied as part of your CPM package. After all this happens, the CCP and BDOS are then loaded from the SYSTEM disc and the CCP is entered. CPM is now active.

As you might have guessed, these sectors on the SYSTEM discs are copyrighted. So when you buy your public domain disc from this magazine, you will find that there is no CPM information on the first two tracks as we do not wish to be prosecuted for piracy. But this does not help you when you put your PD disc in drive A; and type :CPM.

There are two ways to get your disc ready for use. This first method is the recommended one:-

1. Boot CPM (either 2.2 or +, the one you normally use)
2. Get a new disc and format it in SYSTEM format
3. Copy all the files from your PD disc to the newly formatted disc.

Now you may run the programs according to the directions. The next method is for experienced CPM'ers only.

1. Run the SYSGEN program, following the prompts.
2. Run the BOOTGEN program, following the prompts.
3. The target disc in all the above is your PD disc.

Now you should have a bootable PD disc. Place it into drive A: and all should be well. If it's not, then read your MANUAL. The information is all there, it's just a matter of ferretting it out. OK, here's a rundown of all the files on this month's disc:-

The QWIKKEY.COM utility is quite handy if you have to keep keying the same sequence of keys in all the time. Simply define a key to the sequence you want and then use it anytime. See the .DOC file for details.

OK, thats all for this month. As usual, urgent queries on VIATEL MAILBOX NUMBER 534876510. Slower, but just as effective is a letter to the magazine. Keep those requests coming.

```
TITLE  2.2  3.0  COMMENT
FMTXTRA .ASM  Y  N  SOURCE FILE FORMATTER
FMTXTRA .COM  Y  N  .COM FILE ON ABOVE
EXTRAK  .ASM  Y  Y  SOURCE OF PATCH FILE
EXTRAK  .COM  Y  Y  .COM OF ABOVE
EXTRAK  .BAS  N  N  BASIC PATCH FILE
QWIKKEY .COM  Y  N  'HOT' KEY MEMORY RESIDENT KEY
   BOARD ENHANCED
QWIKKEY .DOC  -  -  DOC FILE FOR ABOVE
STARTREK.TXT
STRTRK/2.ASM
TREKMOD .ASM  SEE ZOSO.22
TRKINFO .DOC
BIGTREK .BAS
ZOSO    .22
STRTRK05.ASM  N  Y  NEEDS EITHER MBASIC OR MALLARD
   BASIC TO RUN
CEBUG      .COM  Y  Y  AN EASIER TO USE DDT TYPE OF
PRO GRAM
CEBUG. DOC  -  -  DOC OF ABOVE
```

By Roland Waddilove

The number of roms available for the Amstrad is increasing all the time and it's quite easy to forget what you've acutally got in your rombox. It's even harder to remember the names of all those extra commands, never mind what they actually do.

Help is a useful utility which when run adds two RSXs — IROMS and IHELP. The first prints a list of rom numbers and their type. They can be either foregound, background, extension or Basic. Basic appears in all empty sockets.

The rom type is found by looking at byte 0 of the rom at address &COOO. A 0 indicates a foregound, 1 a background and 2 an extension rom. Basic is marked by having bit 7 set.

The second command IHELP is a bit more intelligent than IROMS, printing out a list of the commands that the rom will accept. It needs an extra parameter to tell it which rom to look at. The disc rom is always number 7 so:

:HELP,7

will print the disc commands.

The command names are all stored in a table somewhere in the rom and the address of the table is found by looking at bytes 4 and 5 at &COO4/5.

The names are stored as Ascii strings with the last letter of each command marked by having bit 7 set. The end of the table is marked by a zero byte.

```
10 REM Help!
20 REM By R.A.Waddilove
30 REM (c) Computing with the Amstr
ad
40 PRINT "|ROMS for list of ROMS"
50 PRINT "|HELP,n for list of ROM c
ommands."
60 MEMORY &9FFF
70 address=&A000
80 FOR i=1 TO 21
90 sum=0:READ code$,check$
100 FOR j=1 TO 21 STEP 2
110 byte=VAL("&"+MID$(code$,j,2))
120 POKE address,byte
130 sum=sum+byte:address=address+1
140 NEXT
150 IF sum<>VAL("&"+check$) THEN PR
INT "Error in line ";170+i*10
160 NEXT
170 CALL &A000
180 DATA 3e00a7c03c3201a001cda0,422
190 DATA 21dea0c3d1bce17ecd5abb,730
200 DATA 23b720f8e90e073e3081cd,4ac
210 DATA 5abbcd15b9a7200fcd11a0,504
220 DATA 3a466f726567726f756e64,455
230 DATA 00fe80200acd11a03a4261,403
240 DATA 736963003d200fcd11a03a,363
250 DATA 4261636b67726f756e6400,400
260 DATA 3d200ecd11a03a45787465,3b9
270 DATA 6e73696f6e00cd11a00d0a,3bc
280 DATA 000df21da0c9a7c8cd11a0,572
290 DATA 4e616d653a00dd4e00cd0f,3c2
300 DATA b9c53a00c0e680280bcd11,4ef
310 DATA a042415349430018312a04,279
320 DATA c07ee67ffe20d45abbcb7e,6f3
330 DATA 2328f37ee67ffe203e0ad4,55b
340 DATA 5abb3e0dcd5abbeb3e15cd,54d
350 DATA 1ebb20f9eb7ea720d73e0a,541
360 DATA cd5abbc1c318b9d5a0c374,6e3
370 DATA a0c31ba048454cd0524f4d,4b5
380 DATA d3000000000000000000,0d3
```

Computing With The Amstrad

Published monthly by Planet Publications Pty Ltd under licence from Database Publications Ltd.
Advertising enquiries phone [002] 31 0130.
Subscriptions – see centre page order form.
General enquiries, phone [002] 29 4377.
Telex: AA 58134, Attn. HT163
Mail: P.O. Box 11,
Blackmans Bay, Tasmania, 7152
Annual Subscription (12 issues)
Australia $ 45
South Pacific (inc. NZ) $A65

'Computing With The Amstrad' welcomes program listings and articles for publication. Material should be typed or computer printed, and must be double spaced. Program listings should be accompanied by cassette tape or disk. Please enclose a stamped addressed envelope or the return of material cannot be guaranteed. Contributions accepted for publication by Database Publication or its licensee will be on an all-rights basis.

The rough with the smooth

About the last thing one feels like doing — and therefore attempts to do — after putting together a magazine, is to write an editorial.

No matter how well organised everything is, or appears to be, there's always last minute problems — missing stories, disc files labelled incorrectly, articles that just won't fit no matter how important they are ... and that's when everything *is* organised!

Can you imagine what it's like to take over a magazine in full flight? A magazine that's been growing at an incredible rate since its first issue, with the subscription list growing by twenty per cent a month, is sure to have growing pains, and *Computing With The Amstrad* is no exception. It was simply impossible to meet the deadlines for a June issue.

Kevin Poynton, the first editor, did an excellent job of establishing *Computing With The Amstrad* and I'm happy to take over the driver's seat from him. It will take a couple of issues for my small, but very dedicated and hard-working team to come to grips with all the ins and outs of which articles we should and shouldn't publish — your help is vital here. Let us know which articles you enjoy, and which (if any) you don't find so interesting.

From this issue we are featuring AMTIX and AMTIPS — two sections that I know will be appreciated by many of our readers. And somehow we've managed to find space for our regular series.

The taxation issue has been resolved to our satisfaction, with the magazine now being classified as tax exempt. It was pleasing to find that there is some logic in these laws after all.

Plans for the introduction of *PC Amstrad* are continuing, and we'll keep you up-to-date with developments, but don't hold your breath ... we're doing all that's humanly possible to organise it.

As I was starting to say at the beginning of *My Say*, the last thing to do for each issue of a magazine is the editorial — well if I don't finish it now the printer is likely to...

Yours in haste,
Rob McKenzie

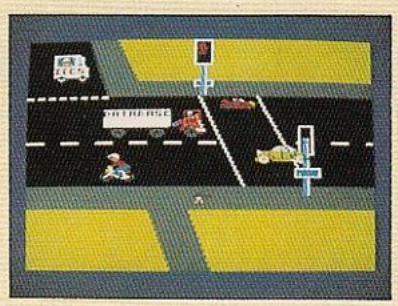
Learning CAN be fun

- Use your Amstrad to teach and amuse your children at the same time.
- Three packages crammed full of educational programs – and so easy to use!
- Each program has been educationally approved after extensive testing in the classroom.

ONLY
$15.95 - Tape
$27.95 - Disk

Ages 2-5

Alphabet
Colours
Counting
House
Magic Garden
Matchmaker
Numbers
Pelican
Seaside
Snap



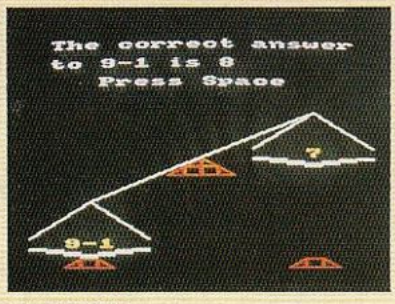
PELICAN
Teach your children to cross the road safely at a Pelican crossing

HOUSE
Select the colours to draw a house – hours of creative entertainment

Ages 5-8

Balance
Castle
Derrick
Fred's Words
Hilo
Maths Test
Mouser
Number Signs
Seawall
Super Spell

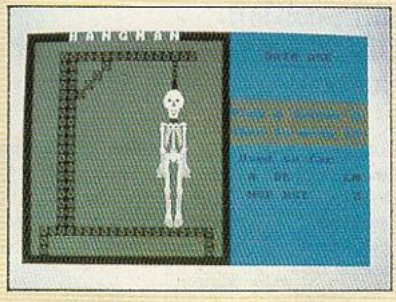
NUMBER SIGNS
Provide the correct arithmetic sign and aim to score ten out of ten



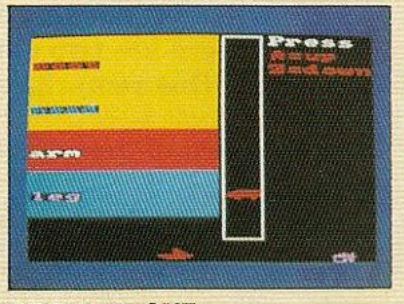
BALANCE
Learn maths the fun way. Type in the answer to balance the scales

Ages 8-12

Anagram
Codebreaker
Dog Duck Corn
Guessing
Hangman
Maths Hike
Nim
Odd Man Out
Pelmanism
Towers of Hanoi



HANGMAN
Improve your child's spelling with this fun version of the popular game



ODD MAN OUT
Find the word that does not fit – before your time runs out

To order, please use the form on centre page

... the COMPLETE personal organiser

Now there's a simple way to keep track of your money, plan your budgets, sort out your files and manage your time far more effectively.

PlanIt's three main modules – Personal Accounts, Financial Diary and Card Index – take care of all your day-to-day activities and help you rationalise your future financial position.

And there are two extra utilities – a Loan Calculator and a Calendar – to complete this remarkable package.

Personal Accounts Gives you up-to-the minute facts about your financial position at any time. Keeps separate accounts of your banking, cash transactions, credit card payments. Allows 24 individual accounts, up to nine different credit cards (and warns you when you reach your cash limit) and as many as 400 different transactions a month. Sets up your standing orders. Automatically updates relevant accounts with each transaction.

Card Index Create your own address book, phone directory, tape library title list. Use the flexible editor to enter or amend data. Sort and search. Call up detailed reports on contents in any form. Produce mailing labels on your printer.

Financial Diary All the features of the best desktop diary – plus much more. Enter up to 15 items per day and have them automatically sorted in time order. Add your expenses and have them totalled in separate categories. Speed search for entries, then mark them for future manipulation or replication.

DATABASE SOFTWARE

To order, please use the form on Page 72