

FOR THE BBC MICRO

BASIC EDITOR

- * DISPLAY BASIC PROGRAM
- * INSERT / DELETE CHARACTERS WITH CURSOR GIVING INSTANT CHANGE
- * INSERT / DELETE LINES WITH CURSOR GIVING INSTANT CHANGE
- * SCROLL UP AND DOWN

```

1000 DEFPROCsetvars (number, width%)
1010 lives=3:screen=1:score=0:score(20)
1020 score=0:S%=50:T%="Minter"
1030 DIM x(xmax),y(ymin),score(20)
1040 FOR set=1 TO 20
1050 READ x(set),y(set)
1060 NEXT set:PRINT"Data loaded OK."
1070 PROCtitle:PROCchars:ENDPROC
  
```

BEEBUG

VOLUME 4 NUMBER 7
DECEMBER 1985

GENERAL CONTENTS

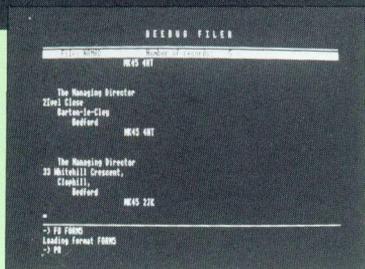
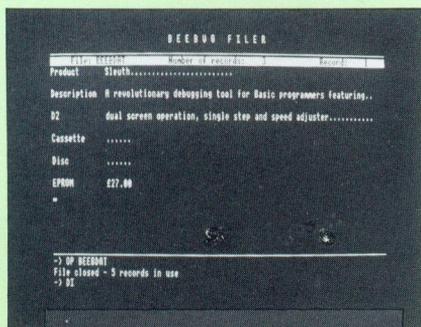
- 3 Editorial Jottings
- 4 Postbag
- 5 BEEBUGSOFT Forum
- 6 News
- 7 Acornsoft's Graphics Extension ROM
- 10 Post-Haste
- 12 Full-Screen Basic Program Editor
- 14 Points Arising
- 15 Build Your Own Database Manager
BEEBUG Filer Part 2
- 19 Upgrading Your Beeb
Acorn 1770 and ADFS
- 22 Personal Diary Manager
- 27 Sure Save Utility
- 28 Games Reviews
- 32 First Course
Colour Blending
- 34 Upgrade Your B+ to 128K
- 35 Programming with Wordwise Plus (Part 3)
- 38 BEEBUG Workshop
Sorting Revisited
- 40 Adventure Games
- 42 Programming Sideways RAM and ROM
Basic Programs
- 44 Book Reviews
- 46 Chinese Chequers

PROGRAMS

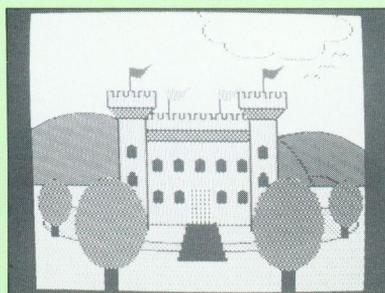
- 12 Full-Screen Basic Editor
- 15 BEEBUG Filer
- 22 Personal Diary Manager
- 27 Sure Save Utility
- 32 First Course
Colour Blending
- 35 Wordwise Plus Segment Programs
- 38 BEEBUG Workshop
Procedures and Demonstration
- 42 Programming Sideways ROM & RAM
Using Basic Programs
- 46 Chinese Chequers

HINTS, TIPS & INFO

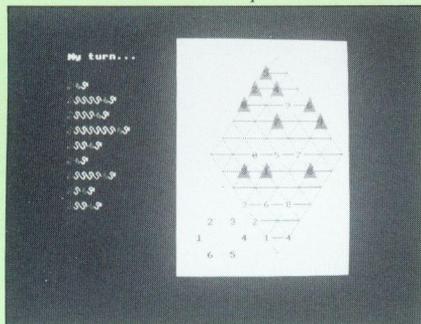
- 8 Simple Shadows
- 8 More User Defined Characters
- 9 New Characters in Mode 7
- 9 DNFS CATS
- 14 Testing OSBYTE Routines
- 33 Stippled Lines
- 33 DNFS Addresses
- 43 CLEAR Description



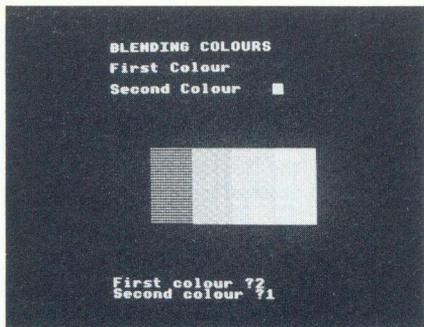
BEEBUG Filer



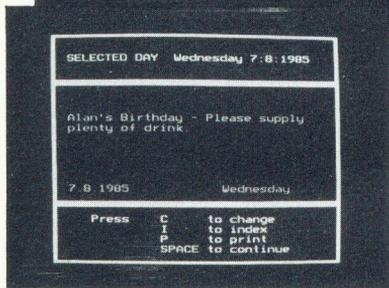
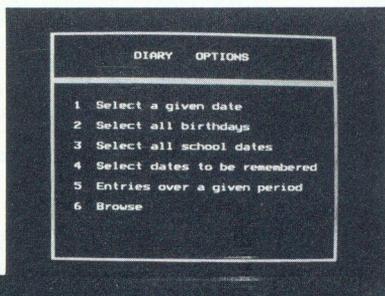
Chinese Chequers



Graphics Extension ROM

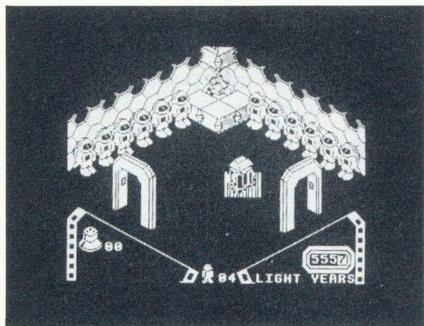


Colour Blending



Personal Diary Manager

Games Reviews



EDITORIAL JOTTINGS

Many of the articles and programs that appear in BEEBUG magazine originate as contributions sent in by members without any prior warning. Indeed some of our best and most exciting programs arise in this way. Other articles and reviews are produced by a small group of longstanding and valued contributors whose names you will see quite often.

Although we already receive a substantial quantity of material, we are always on the lookout for anything particularly novel or innovative, as well as other good well written material. If you would like to contribute to BEEBUG then let us have your programs, articles and ideas. It always helps us if you do include a written or printed summary of your submission, and enclose a copy of any programs on disc or cassette.

We do pay generously and promptly for everything published. We will also send you a copy of our "Notes of Guidance for Contributors" if you send us an A5 SAE. Please note that any contributions, and any other correspondence intended for the editorial office, should be sent to us at P.O.Box 50, Holywell Hill, St Albans AL1 3YS.

BEEBUG SHOP

We are very pleased to announce that we have now opened a shop/showroom in our St Albans premises (some of you will already know of this). This will enable us to offer a wider range of computer products to members and will, of course, also stock all BEEBUGSOFT software and other items. We hope that many BEEBUG members will take the opportunity to visit us, to browse, and to discuss their requirements. The full range of shop items is available to both mail order and personal callers. Full details of the products available and information about the shop can be found in the leaflet enclosed with this issue.

PROGRAM CLASSIFICATION

All programs in the magazine, and on the magazine cassette/disc, are marked with the symbols shown below. An unmarked symbol indicates full working, a single line through a symbol shows partial working (normally some modifications will be required), and a cross through a symbol indicates a program that will not work on that type of system. There is also a symbol for the B+ which includes the 128K version.

Basic I	I	Electron	⊙
Basic II	II	Disc	⊠
Tube	⊖	Cassette	⊠
Model B+	+		



POSTBAG



POSTBAG

Members' Discounts

Thank you for featuring our name and address in previous issues. We must now ask you to delete it as we have decided to withdraw from the computer market place.

When we started at the end of 1982 we considered that home computing would be a natural extension to our established business as stationers and office equipment dealers. After all, aren't we all going to have an electronic office in our own homes?

The Beeb was the first machine we supported and it continued to be the mainstay of our activities. By the summer of 1983 programs did not sell on their merits but on the outrageous 'hypes' of marketing men in the newly emerged computer magazines. BEEBUG alone has remained true to its readers in providing honest appraisals of software.

At the same time we found that returns of allegedly faulty software were reaching epidemic proportions. Adults and youngsters alike were prepared to bring other shop's software back and lie blatantly about the tape "they bought yesterday". In fairness, we must add that the percentage of Beeb software returned was consistently lower than for other user groups.

The market place has changed drastically and we are happy to leave it to

others. Our thanks for their custom to all the BEEBUG members who have written to us in the past.

Peter Southeran
Director
A.A.Southeran Ltd.

Mixed Reception to Watford

Thank you for returning the magazine disc for Vol.3 No.10 about which I had complained that certain programs did not run on my machine. I took the disc along to Watford Electronics and was told that the Watford DFS 1.2, the version I had, would not run these programs. I had to buy a DFS 1.43 chip for £7 before the 'Mixed Modes' program would run at all. I thought you might like to warn others.

Norman Barker

Although we are unable to test all magazine programs on all versions of other DFS chips, we do regularly use a Watford DFS 1.40. We have found that in general there are few if any problems in running programs with the Watford DFS 1.3 or later versions.

Larger Factorials

Surac's article on 'look-up' tables (Workshop in Vol.4 No. 4) reminded me of a way to allow the BBC micro to manipulate factorials greater than 33! (8.6833E36). Logarithms can be used to generate the value of the factorial, which can then be stored for use in any calculation. For example:

```
100!=10^(Log100+Log99+
Log98 + . . . . .)
```

The log series can be programmed as:

```
100 DEF FNlog(n)
110 LOCAL result,sub
120 IF n<=1 result=LOG(1)
    ELSE result=log(n)+
    FNlog(n-1)
130 =result
```

The results of this are perfectly satisfactory for statistical and many other calculations where the end result is not out of range, though the method will not help in calculating individual factorials greater than 33!

Andrew Farmer

This is a useful technique, though with limitations as described. In fact, many mathematical functions can be represented as a series. This is how Basic and other languages calculate such functions as SINE and COSINE.

Puzzled

The following 6 program lines might be of interest to those of your readers that get frustrated trying to solve their jigsaw puzzles using the program you published in Vol.4 No.3. Pressing key 'S' will now complete the puzzle.

```
2191 IF K%=83 PROCsolve
3000 DEF PROCsolve
3010 FOR j%=0 TO mN%-1
3020 IF T%(j%)=j% THEN 3060
3030 FOR k%=j% TO mN%-1:
3040 IF T%(k%)=j% u%=k%
3050 NEXT: PROCswap(j%,u%)
3060 NEXT
3070 ENDPROC
```

P.Grilli

BEBUG SOFT FORUM

SPELLCHECK UTILITY DISC

A utility disc is now available for users of Spellcheck II (or the new Spellcheck III), and offers two useful routines. The first will transfer words from a Spellcheck I dictionary disc to a Spellcheck II (or III) disc. The second will list the words in your dictionary disc to the screen or to a printer, alphabetically sorted by the first letter.

As a service to members, this disc is available at £4.00 plus 50p post and packing.

SLEUTH An Enhanced Version

A new version of Sleuth, our Basic debugging ROM, is now available and an upgrade service is offered to owners of versions prior to 1.06. The new version has the following, additional, facilities:

1) The ability to enter Sleuth directly from any stage of your own program. e.g.:

```
IF D%=10 THEN *SLEUTH  
or:
```

```
ON ERROR *SLEUTH  
and so on.
```

2) The new command 'OUT' to exit Sleuth and continue running your program outside Sleuth, from the current line.

3) Dual screen work-space may be located in sideways RAM.

4) Variable update now includes arrays.

5) Aries B-20 and Watford 32K Ramcard compatible.

Members wishing to upgrade should send £7.50 plus 50p post and packing with their old ROM and manual to the St. Albans address.

DUAL FORMAT DISCS

A number of our programs are now supplied on dual format (40/80 track) discs. Consequently it is not possible to copy them using the normal *BACKUP command. Should you wish to keep your original disc safe and work from a copy, you should use the following command:

```
*COPY S D *.*
```

(where S and D are source and destination drives).

This will abruptly terminate with disc fault 18, leaving uncopied two special dummy files (which unusually have filenames containing space and asterisk characters). Carefully check that all the other files have been copied and use

```
*COPY S D <filename>  
to copy anything missing. You will then have a new working copy of the disc. Remember to reset the auto-boot option with the *OPT command (i.e. *OPT 4,3).
```

We would remind you that unauthorised copying, apart from the production of one backup copy only for use by the purchaser, is strictly illegal.

EXMON II LABELS

To create labels in the Exmon II assembler/editor, enter the editor at the address required and, if necessary, press Tab to enter the assembler. Now type in the label preceded by a full stop and followed by <Return>. For example, to label the OSASCI and OSWRCH entry points follow this procedure:

```
*EXMON  
E FFE3  
.osasci <Return> <Escape>  
E FFE6  
.oswrch <Return> <Escape>
```

TECHNICAL QUERIES

Our team will be pleased to answer your queries concerning any of our products - their features, their use, or their compatibility with other hardware/software.

All inquiries should be addressed to:

The Software Manager,
PO Box 50,
St. Albans,
Herts., AL1 3YS.

Please ensure that you provide us with full details of your set-up, including OS and Basic version numbers, other ROMs in your machine, and details of all hardware attached.

If you prefer, technical queries may also be made by telephone between 2.00 and 4.30 in the afternoons. Our hotline number is 0727-40303.

News News News News News News News

New Concepts

Computer Concepts is to launch another speech ROM next month. The speech extension ROM is used in conjunction with Computer Concepts' previous speech ROM. It gives full text-to-speech conversion with semi-intelligent translation of the quirks of the English language into the phonemes recognizable by the older speech ROM.

Computer Concepts' other promised 'ROM-link' ROMs are due out in the next few weeks too. Inter-word and Inter-base are the word processor and database of the system, respectively. Inter-word offers full on-screen formatting in an 80 or 105 column format and costs £65. Inter-base comes on two 16K EPROMs (for £69) and provides an essentially language-based database management system with full control over the other ROM-link products as well.

Computer Concepts is also keen to point out that the Inter-chart business graphics package is compatible with the Plotmate plotter (see BEEBUG Vol.4 No.4). Computer Concepts is on 0442-63933.

Watford Speeds Up

The Delta Card is the latest offering from the ever-prolific Watford Electronics. This board contains a 4MHz 65C02 and 64K of fast RAM and enables your Beeb to run up to twice as fast as normal. The Delta Card costs £114 complete with its control ROM, and is claimed to be

compatible with all software written for the normal Beeb.

Yet another Watford Electronics DFS is available. This time it is a double density DFS for the B+. The B+, of course, already has a double density disc controller chip so this product is merely the DFS ROM to enable the B+ to use double density discs. The B+DDFS has all the features of the old model B version and costs £40. Unlike previous Watford DFSSs, this price includes the manual. Also in the sideways arena, Watford has a 16K sideways RAM module to tempt you. Costing £29 the module will plug into any ROM board or even the Beeb's main circuit board and can be fitted with write and read protect switches at an extra cost of £2. Further details from Watford on 0923-37774.

Cheap IEEE

As an alternative to Acorn's unit, the TE-4881A interface from Tussen Electronics enables you to connect any IEEE-488 standard instruments and other laboratory devices to the Beeb. The TE-4881A costs £130 from Tussen on 051-648 4488.

Kiss and Tel

BBC Publications has joined the throng of Viewdata services for the Beeb with a new service called OwlTel. The service operates 24 hours a day on 01-927 5820 and provides free information on BBC Soft products and the

Computer Literacy scheme for anyone with a suitable modem and software.

Ramifications

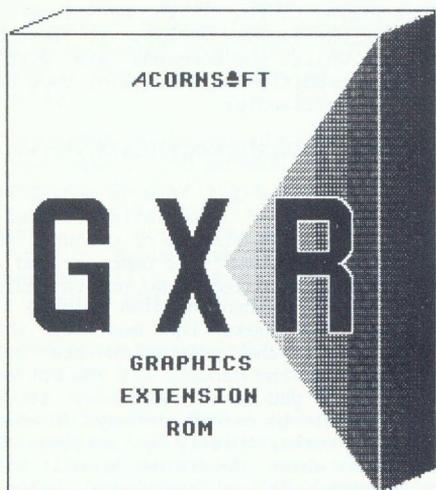
Promicro's 16K RAM module is a convenient method of adding sideways RAM to your Beeb. It is based around two piggyback 8K chips and can be plugged into any sideways ROM socket with a single flying lead. Unlike similar systems adopted by many Acorn users, Promicro's costs only £29.50. Further details from the company on 01-554 6219.

Top Tandata

Tandata has launched yet another modem. At £535 the Tm602 is not cheap but it offers 1200/1200 full and half duplex, 300/300, 1200/75, and 75/1200 standards, auto-dial, auto-answer, and auto-logout and is fully BT approved. Further details from Tandata on 06845-68421.

Designs on 3D

Interactive 3D is a package for designing 3D models on the Beeb's screen. Unlike most 3D graphics systems, I3D does not require co-ordinate entry but actually allows you to design solid objects on the screen by showing you four views of the object at once. Lines are created and moved into position with the cursor keys and completed models can be saved to disc or cassette. Interactive 3D costs £8.95 (cass.) or £12.95 (disc.) from Data Dynamics on 0525-402447.



If you want to transform your graphics displays then Acornsoft's long-awaited Graphics Extension ROM could be the answer. Mark Sealey has been putting the ROM to work.

Product : Graphics Extension ROM

Supplier : Acornsoft

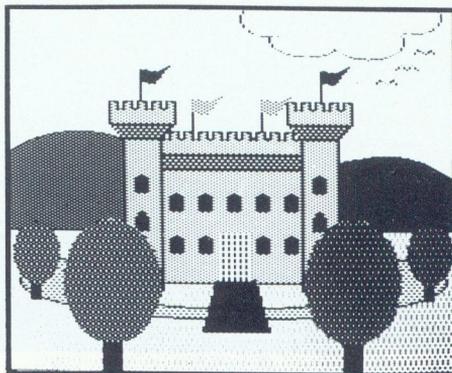
**Cambridge Technopark,
645 Newmarket Road,
Cambridge CB5 8PD.
0223-214411**

Price : £29.90

Acorn's Graphics Extension ROM (GXR) provides a set of additional graphics commands for spectacular, new and easy shape drawing and filling, copying and moving areas of the screen, and a variety of shadings. In addition, Acorn have provided a sprite graphic utility.

Unlike Computer Concepts' Graphics ROM which used a variety of new * commands, GXR mainly uses the PLOT extensions which, until now, had the famous 'for future expansion' description, and provides new VDU23 and GCOL parameters.

Any language that supports the PLOT, GCOL and VDU commands can make use of these routines provided the GXR ROM is



installed and activated. It is 6502 Second Processor compatible and will also work on the Electron as well as with the Aries B-20 RAM Board. However, for use with the B+, a special version of the GXR is available.

The ROM is activated by *GXR and can be turned off by *NOGXR. A rather hefty three pages of RAM are lost when the ROM is activated so PAGE is reset to &1C00 on a disc system. Other * commands available from within the ROM are an online HELP list of the codes, the sprite controls, and a *FLOOD/NOFLOOD option. Use is also made of *FX163.

Supplied with the package are example programs on cassette with a special tape-to-disc transfer utility. The copy of the pre-release manual was up to Acorn's usual standard in its clarity. Each section begins with a diagram of a sector, arc, ellipse or whatever, explains the simple codes to achieve the effect and then gives longer examples.

The PLOT calls (with the exception of the Copy routines) follow the old pattern of coding described on page 319 of the BBC User Guide (i.e. 1 is current graphics foreground, 2 relative logical inverse colour, etc.). PLOT may now take codes between 88 and 248 as its first parameter, to draw and/or fill rectangles, parallelograms, circles, ellipses, arcs, segments and sectors.

For example, using the old way to fill a rectangle with two filled triangles (two MOVE and two PLOT instructions) is prone to mistakes and about twice as slow as the



new method:

```
MOVE 200,800      :REM top left corner
PLOT 101,700,200 :REM bottom right.
```

To draw a filled ellipse (hitherto requiring almost advanced qualifications in geometry (see BEEBUG Workshop in Vol.4 No.3), you simply MOVE to the centre, MOVE to the right-hand point or wider radius, and PLOT (with first parameter in range 200-207) the top (or narrow radius). This gives an almost instantaneous block fill.

Many of these extended routines will be useful as utilities in, say, pie-chart routines for extended software written in Basic and really are impressively fast - presumably as it accesses the OS directly (hence the ROM's own version number, 1.2).

PATTERN AND SHADING STYLES

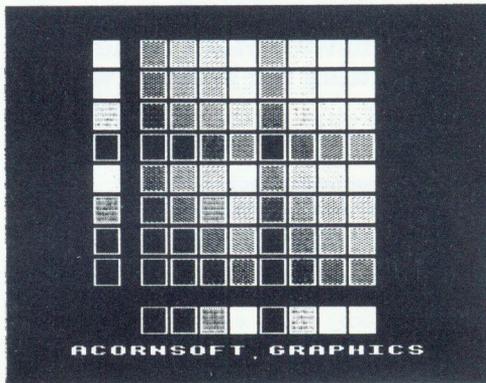
It has long been known that the GCOL command is capable of extension beyond what is outlined in the User Guide. Only months after the BBC micro's appearance Gaelsett released their Extended Graphics utility on tape. Now Acorn's own ROM takes

colour mixing to its limits by pixel/bit manipulation (using VDU23). It must be said that whilst the commands are within the true tradition of BBC Basic, they are very user unfriendly:

```
VDU 23,2,&1D,&2A,&1D,&2A,&1D,&2A,&1D,&2A
```

This redefines GCOL 16,0 in mode 2 to give a white and green stippled mix. A virtually infinite mixture of colours and hues is possible, and well demonstrated in the examples supplied. Some understanding of how the BBC micro handles pixels is essential to make the most of this facility, but the principles are well explained in the manual. All the options are there, if you seek them out. It is also possible to extend the ways in which flood fill works, chiefly by setting the conditions which determine when it will stop - which fore or background colour, etc. Again, it is a very versatile and fast set of commands.

Copying and/or moving any area of the screen to any other position could not be



HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

SIMPLE SHADOWS - Nicholas Ahmon

Text in modes 1 and 5 can be given a 3D effect using *FX155,N. A shadow is produced 'behind' each letter in the colour specified by the value of N. The whole screen is affected and the shadows can be cancelled either by changing mode or using *FX155,B with B the value of the background colour.

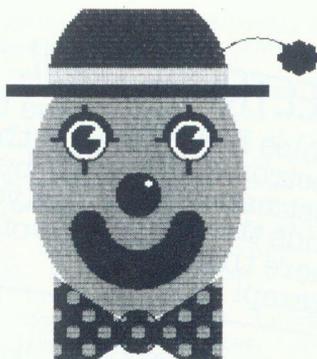
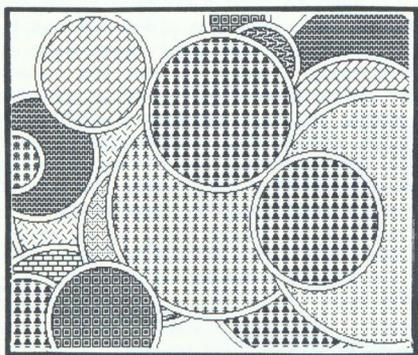
MORE USER DEFINED CHARACTERS - Jonathan Temple

Up to 64 user defined characters can be defined without using *FX20 or moving PAGE by first including ?&36D=&B. The 'extra' definable characters are numbers 192-223. The definitions reside from &0B00 but this can be altered by changing the right hand side of the statement.

easier. You first mark with two co-ordinates (bottom-left, top-right) the area to be copied from. Then PLOT (range 184-191), newx, newy where it is to be moved to.

Included in the example programs is a fairly simple CAD package that makes use of most of the facilities contained in the ROM. More sophisticated drawing programs abound, but the fact that this one is written in Basic and yet still plots, draws and fills so quickly is evidence of the Graphics ROM's own in-built speed.

The final feature of the Extension ROM is a sprite facility. It allows you to design and edit up to 256 sprites (with PAGE still at 8800 with a Second Processor). Each can then be driven from Basic by selecting it as number n: *SCHOOSE n followed by PLOT (range 204 to 211) and its x,y co-ordinates. A sprite is loaded and saved as a block of memory, and a very impressive feature is that of defining (another VDU 23 command) any area of the already plotted screen and actually moving this as a sprite.



The GXR's drawing commands for circles, ellipses, and so on are also available on Computer Concepts' Graphics ROM. However, this does not allow the easy passing of variables to the extra graphics commands as does GXR. In addition, although GXR has no text angling and enlarging facilities nor turtle graphics, its colour mixing and character filling abilities more than make up for this.

Now for the bad news. Whilst using the Graphics ROM it behaved perfectly but there was a definite clash with Wordwise Plus, causing this to do all sorts of unlikeable things. I understand that this is because the latter makes use of *FX163 calls reserved for Acornsoft ROMs. It is a major drawback for users of both and it remains to be seen whether Computer Concepts can modify this excellent wordprocessor to avoid these clashes.

However, this long-awaited ROM is cheap as ROMs go and makes fill, draw, copy and shade/colour routines both easy and very quick, using already familiar commands. Imagination - as they say - is the only limitation.

HINTS HINTS HINTS HINTS HINTS HINTS

DNFS_CATS - Eric Pope

Cataloguing a disc with the Acorn (or Watford) DFS produces a list of files in alphabetic order by ASCII code - lower case letters have a lower priority to upper case. The DNFS also arranges catalogues in alphabetic order but makes no distinction between upper and lower case.

NEW CHARACTERS IN MODE 7 - R.J. Walking

For a 'different' character set in mode 7 (the CRTIC chip registers are changed) type in: VDU23;8,144,0;0;23;9,9,0;0;0; To restore things to normal use: VDU23;8,147,0;0;23;9,18,0;0;0;



ELECTRONIC MAIL

James Fletcher shows how electronic mail beats the postman every time, but finds that it will cost you far more than the price of a stamp!

The very words 'Electronic Mail' conjure up visions of the high-tech society that we are all going to live in one day soon, and I have recently been trying out some of these systems for myself. I am pleased to report that it is possible to use your Beeb as a high-speed electronic postman today, but that there are just one or two little snags.

The modern business executive is somewhat spoiled for choice as far as electronic mail services are concerned, and several companies are making a respectable living by providing him with a whole range of electronic services. The home user is not quite so well off, but has several means of communicating with other computer users at a moderate cost.

The simplest and cheapest electronic mail service is available to Prestel and Micronet users, but there are some problems that only become apparent when you try to use the services in earnest. Prestel messages are actually 'free' once you have paid your subscription (£16.50 per quarter for Prestel/Micronet), your telephone bill, and the computer-connect time charge of 6p per minute if you are using the system during office hours. This latter charge is waived between 6pm and 8am on weekdays, after 1pm on Saturday and all day on Sunday.

The first problem that you come across is the rather fundamental one that you can only communicate with other Prestel users, and the same holds true for any of the other systems - both you and your correspondent must be members of the same electronic mail service. This is somewhat restricting when trying to write to dear old Auntie Flo, but is very useful when trying to talk to friends who are computer

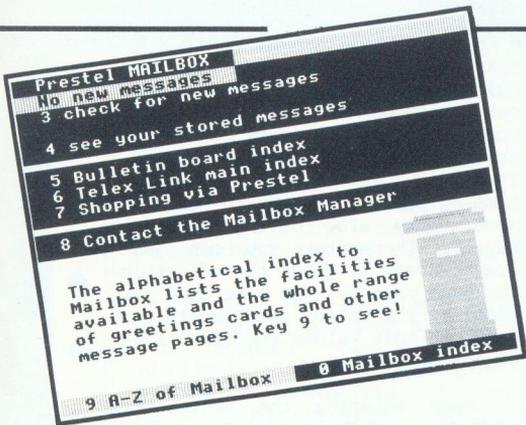
fans. Finding out whether or not your friends are system users, and their electronic mail numbers, is made easy by an on-screen directory, and if you really do want to hide your light under a bushel you can still be 'ex-directory'.

When using the Prestel message service you are provided with a simple teletext-type standard page on the computer screen which has blanks for you to type your message in. No more than about 100 words can be fitted onto a Prestel message page, which restricts you to short notes. The message service is extremely easy to use, with simple instructions on the screen.

The service proved extremely efficient, messages being received in different parts of the country, via different Prestel computers, within just a few seconds. When you first switch to Prestel the screen warns you if any incoming messages are waiting for you to read, and once you have read them you are offered the choice of storing the messages for further reference in the Prestel computer, or deleting them. If you need to send a message to an organisation that has the old-fashioned telex service, rather than your up-to-date electronic mail, Prestel allows you to send a short telex message anywhere in the UK for 50 pence, and has recently extended the service worldwide, a most useful service for a comparatively low charge, especially when you remember that there are more than one and a half million telex users.

There are a few points to bear in mind. You cannot, of course, find out if any messages are waiting





for you without dialling Prestel, and that costs money. Further, the 1200/75 baud rate used by Prestel means that sending your messages can be very slow, even more so between the peak times of 6pm and 8pm.

A far more sophisticated service is provided, primarily for business users, by British Telecom, under the label of 'Telecom Gold', and this service has just become available in a form suitable for domestic users thanks to an agreement between one of the major computer magazine publishers and BT, which has resulted in the formation of 'Microlink'.

With Microlink, the modem can be a simple Prestel type or the more expensive 300/300 or 1200/1200 baud type, offering faster input of your messages than Prestel. You also need some software on disc or ROM. This is usually provided by the supplier of your modem, or you can buy one of the sophisticated ROMs such as Communicator from Computer Concepts or Comstar from Pace. Microlink enables you to use the remote mainframe computer almost as if it were your own. You can not only send messages and telexes to other users, but you can set up your own filing system on the mainframe and perform vast calculations. If you give other people the keys to your electronic filing cabinet they will be able to access it too from anywhere in the country.

Telemessages, the modern version of the telegram, can be sent from your keyboard, and BT will ensure that they are delivered to any house in the UK by first post the next day. This service allows communications even with houses without a telephone line, and is available via Microlink for

only £1.25 for up to 350 words, quite a bit cheaper than normal telegrams.

Microlink has its own multi-user bulletin boards which have enormous potential for making friends of computer users the length and breadth of the land. It is also part of the international DIALCOM network which sends your messages to other electronic mailboxes around the world at very reasonable charges since local-call access has been specially arranged even for these long-distance systems. Huge databases of information, rather like encyclopaedias, can be searched at will, and any of the information can be displayed, stored, and printed out when required.

It doesn't take long playing with Microlink/Telecom Gold to realise that this is what home computing should be. An almost infinite database filing system permanently on tap and the ability to communicate world-wide really bring a whole new excitement to your hobby. The costs of Microlink seem reasonable. There is a joining fee of £5 and then a standing charge of £3 per month. It costs from 3.5 pence/minute outside peak hours to 10.5 pence/minute during the normal business day to use the service, and there is an extra charge of 2 pence/minute if you are outside London. There are also charges for storing information on the mainframe.

Just to send messages via electronic mail proves to be fairly cheap and is immensely convenient, but I suspect that most users, like me, would find that they couldn't resist the lure of the international bulletin boards and databases. Once you are on-line to these the minutes just fly by and the costs soar. Although I can think of 101 reasons for my high phone bills, deep down inside I know that it is something to do with that Beeb in the corner of my study!

Contacts:

- Prestel Microcomputing: Freefone 100 and ask for Prestel sales, or 01-583 7214.
- Micronet 800: 8 Herbal Hill, London EC1R 5EJ or phone 01-278 3143.
- Microlink: Database Publications, Europa House, 68 Chester Rd, Hazel Grove, Stockport SK7 5NY.
- Telecom Gold: 60-68 St Thomas St, London SE1 3QU or phone 01-403 6777.

Full-Screen Basic Program Editor

The Basic full-screen editor presented here is extremely fast and powerful for its amazingly short length. This must surely be the utility that all Basic programmers have been waiting for.

Although Basic's built-in editor is quite adequate, most programmers weary of excessive use of the Copy key after a while. What BBC Basic needs is a true screen editor that is fast and easy to use. That is just what the surprisingly short utility program listed here provides, and being written in Basic itself can be truly called a Basic Editor.

But why use an editor written in Basic, not assembler? The high level language makes the listing much more compact for you to key in from this article, and far easier for you to customise according to your personal needs and preferences.

The reason for NOT doing so, you might think, is that Basic could not possibly be fast enough. Try it out and you will be surprised!

The editor consists of a procedure and five functions, designed to be merged onto the end of the program under development. Merging by means of *SPOOL and

*EXEC is described in the User Guide, page 402. Once merged, they can be loaded and saved as one, until the program is finished when the editor lines can be deleted. The editor is invoked the first time by typing PROCED in command mode. This also defines function key 10. Thereafter you can just press Break to enter the editor in the same mode as that last used for editing. The editor also resets to their default values the *FX states set up at the time.

Upon entering the editor you find yourself in 'procedure mode'. This means that only the first line of the program and lines beginning with the keyword DEF (one leading space is allowed) are listed, to give an overall view of properly structured programs (note that this also lists functions). Move through the definition list using the up and down cursor keys, and if it is long enough you will find that it scrolls properly in both directions. To actually list the program you need to enter 'line mode' by pressing Return, but first move the cursor either to the top or to the particular definition you want to home in on directly.

In line mode the down cursor key produces a line-by-line listing, once again with true scrolling: whatever disappears off the top of the screen can be recalled with the up cursor key. The left and right cursor keys move the cursor within individual lines, so you can insert or delete characters at will. The Copy key gives a 'clear to end' function. Whatever changes you make are actually entered into the program only when you press Return, so if you accidentally mess up a line just press Escape or the vertical cursor key to recover the original version. When you do press Return the screen is cleared and the new version of the line is listed, with any abbreviations (like P. for PRINT) properly expanded.

The Return key is also used to insert a new line between the

current line and the next. The default line number is chosen, provided there is room, to be the normal 10 greater than the current line number. If this is not possible, a difference of first 2 then 1 is used. A warning beep means the line numbers are consecutive, leaving no room at all.

The fact that the line number itself can be edited as part of the line enables you to copy an identical, or modified, version of a line to a completely different part of the program. That then becomes the current line.

All this, with very little practice, becomes very natural to use. There are also just two other commands to learn. In line mode Ctrl-D is a quick way to delete the current line. Ctrl-S is a string search which asks for a target string, then lists downwards from the current line until it is found. The string can include keywords and/or line numbers. It can be used in procedure mode as well to search out a definition by name.

Space doesn't allow a full explanation of how the editor works, but the crucial technique is a means of tying up communication between a Basic program and command mode, required for listing and entering new program lines. It is done without the involvement of the input buffer (whose 32 character capacity is not sufficient for program lines of a reasonable length) and without calling routines in the Basic ROM directly. The RDCHV and WRCV vectors are redirected to code located in Zero Page at &70, which in turn uses Page &A (normally the RS423 input buffer) as a buffer for input and output. The effect of using the same buffer for both purposes is that PRINT statements can generate commands which are obeyed as soon as command mode is entered. Provided the command string ends with a Basic function reference, that function will be re-entered as soon as the commands have been carried out. The code is so short

it is poked directly with the ! operator, but an assembler listing would look like:

```
.WRCHentry STA &A00
      INC WRCHentry+1
      RTS

.RDCHentry LDA &A00
      INC RDCHentry+1
      CLC
      RTS
```

Surprisingly, such recklessness does not end in disaster. The command mode input causes echos, intended for the screen, which overwrite the common I/O buffer, but the RDCH pointer keeps safely behind the WRCH pointer until the vectors are restored.

As I said at the beginning, a strong reason for writing the editor in Basic was for it to be easy for you to modify. So I hope readers will send in their ideas for enhancements to the truly Basic editor described here.

```
32000 DEF PROCED
32010 MD=?&355:IFMD=0 ORMD=3 D%=80
ELSEIFMD=2 ORMD=5 D%=20 ELSESD%=40
32020 ?&8F=MD:*KEY100.|MMD=?&8F:MOD
EMD|MPROCED|M
32030 !&78=&1877E60A:??&7C=&60
32040 IFPAGE<6400M%=TOP+256ELSEM%=&
1100
32050 T%=PAGE+1:L%=M%:A%=&A10:I%=!&
20E
32060 K%=0:W%=0:S%=0:F%=TRUE
32070 VDUI2;;PRINT'CHR$32"Procedure
mode"
32080 *FX4,1
32090 *FX229,1
32100 IFFNY
32110 ENDPROC
32120 DEF FNW
32130 IFL%=M%=0
32140 !L%=0:L%=L%-2:T%=!L%:=TRUE
32150 DEF FNX
32160 =256*?T%+T%?1
32170 DEF FNY
32180 !&70=&E60A008D:!!&74=&AD6071:!!
&20E=&760070:IFW%PRINTT$'"IFFNZ":=0
ELSEPRINT"L.";FNX'"IFFNZ":=0
32190 DEF FNZ
32200 !&20E=I%:IFW%W%=0:VDUI2:=FNY
32210 R%=A%+LEN$A%+1:IFK%=139 PRINT
STRING$(0%+2+LEN$R%DIV D%,CHR$11);
```

```

32220 O%=LEN$R%DIV D%:R%?(LEN$R%-1)
=13:PRINT'CHR$32;SR%;
32230 IFFK%=GETELSEIFLEN$R%>LENS$:
IFINSTR(SR%,S$)F%=TRUE:GOTO32230
32240 IFK%=139:IFFNW:=FNY
32250 IFK%=138X%=T%:REPEAT:X%=X%+X%
?2:UNTILS%OR?X%=255ORX%?3=&DDORX%?4
=&DD:IF?X%<255!L%=T%:T%=X%:L%=L%+2:
=FNY
32260 IFK%=13 ANDNOTS%S%=TRUE:L%=M%
VDU12;:PRINT'CHR$32"Line mode"=:FNY
Y
32270 IFK%=19 AND?(T%+T%?2)<255F%=0
:PRINT'CHR$32"Target:";:K%=137:S$=F
NEL(""):K%=138:VDU11:GOTO32250
32280 IFK%=27THEN32360
32290 IFK%=138ORK%=139ORNOTS%F%=TRU
E:GOTO32230
32300 W%=TRUE:IFK%=4:T$=STR$FNX:VDU
7:=FNW*FNY
32310 IFK%=13PRINT:H%=POS:V%=VPOS:V
DU28;24,39,V%,11,26,31,H%,V%,32:N=2
56*?(T%+T%?2)+?(T%+T%?2+1)-FNX:T=FN
X+1-(N>2)-8*(N>10):PRINT;T;:K%=137:
VDU6-(N=1):T$=FNEL(STR$T)ELSET$=FNE
L($R%)

```

```

32320 T=VALT$:IFK%<>13ORT$=0T$="":VD
U7:=FNY
32330 IFT<FNX:REPEAT:UNTILNOTFNW OR
T>FNX:IFL%=M%T%=PAGE+1
32340 IFT>FNX REPEAT:IL%=T%:L%=L%+2
:T%=T%+T%?2:UNTILT<=FNX
32350 =FNY
32360 *FX4
32370 *FX229
32380 =0
32390 DEF FNEL(A$)
32400 $A%=A$:C%=A%+LEN$A%:VDU152,8;
REPEAT:B%=K%:IFK%=127:IFC%>A%$(C%-1
)=$C%:K%=136
32410 IFK%<135:IFK%>31$(C%+1)=$C%:?
C%=K%:K%=137
32420 IFK%=135:?C%=13:K%=0
32430 IFK%=136:IFC%>A%$C%=C%-1:VDU8
32440 IFK%<>B%H%=POS:V%=VPOS:PRINT$
C%;CHR$152;:V%=V%+(VPOS=24):VDU10,3
1,H%,V%
32450 IFK%=137:IF?C%<>13:C%=C%+1:VD
U9
32460 K%=GET:UNTILK%<32ORK%>137:=$A
%

```



POINTS ARISING POINTS ARISING POINTS ARISING POINTS

DYNAMIC MEMORY WINDOW (BEEBUG Vol.4 No.5)

In the function key definitions in this program the vertical bar characters (|) were inadvertently omitted. The lines involved and their correct versions are as follows:

```

1370 DEFFNturnoff:str$="K.7*FX13,4|M":=FNoscli
1390 DEFFNturnon:str$="K.8MO.7:V.31,0,8:*FX14,4|M":=FNoscli
1410 DEFFNbreak:str$="K.100.|M?&220=&"+STR$(code MOD256)+"?:&221=&"+STR$(codeDIV256)
+"|M":=FNoscli

```

FURTHER DISC MENU EXTENSIONS (BEEBUG Vol.4 No.5)

If you add all the extensions as published then you will encounter no problems. If you are just adding the extensions for Wordwise and View, or for Disc Doctor's *SWAP command you will also need to enter a modified version of line 1230 as follows:

```
1230 DIM NAME$(E%-1),P%(E%-1)
```

ROULETTE (BEEBUG Vol.4 No.6)

A number of lines in this program were quite inadvertently repeated. There is, however, no problem and all the program is listed correctly in the magazine. Just ignore the redundant lines.



HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

TESTING OSBYTE ROUTINES - Jonathan Temple

Although many OSBYTE routines are not implemented as *FX calls, they can be easily tested from Basic, to see their effect, with the following function key definition. When prompted, enter the value for A (the OSBYTE call number) and the X and Y registers. The routine is then called and the results printed out.

```
10*KEY0 I."A-A%X-X%Y-Y%:R=USR&FFF4:P."A=";R A.&FF:P."X=";(R A.&FF00)/256:P."Y=";(R A.&FF0000)/16^4|M
```



Build your own Database Manager

BEEBUG Filer Part 2

We continue our major new software project, BEEBUG'S own database program, by describing the file structure in detail and adding functions to provide flexible formatting and output of your records.

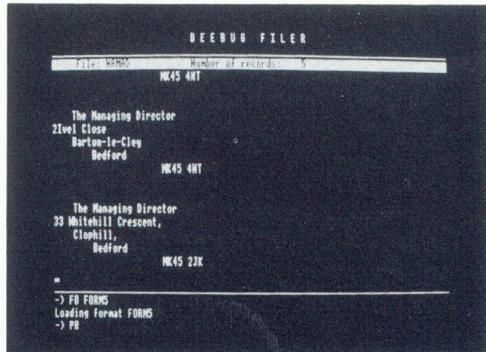
Last month we commenced a major software project in the magazine to construct your own database program. The Basic program, called BEEBUG Filer, was presented in the first instalment together with a general introduction and description. This month we shall look at the fundamentals of our database program in some detail, but first we will see how we can improve on some of our first steps, and add some exciting new features. The major addition will allow you to output your records in a flexible format of your own design.

GENERAL DESCRIPTION

The program uses mode 3 to allow the maximum amount of information to be displayed on the screen. We shall have to watch memory space carefully as the program develops, but in the meantime we must endeavour to keep the program as concise as possible and yet provide a clear and readable program. You can help yourself by omitting all unnecessary spaces when typing the program in (including the space between the line number and the instruction that follows). This is also the reason why you will find reals rather than integers used throughout most of the program. Although integers use less space, every reference to an integer uses an extra byte (for the '%') which often outweighs any advantages.

IMPROVEMENTS

This month I want to describe a number of minor improvements to the original program. These both help to make the program more foolproof and more efficient, and pave the way for the next major



additions. I have introduced three new utility functions called PROCread, PROCwrite and PROChead. The main purpose here is to separate the functions of:

- read record from file
- display record on screen
- write record to file

The procedure PROChead simplifies the task of updating the screen information header. These new functions, the other lines that are changed as a result, and a number of other minor improvements are listed below. The best way to update the original program is to type in these lines and save as a *SPOOL file which can then be *EXECed with the original program. You will also need to delete lines 6100 and 6120 from the original program.

```

1050 delete%=0
1060 DIM com$(20),record$(maxf,1),field
$(maxf),width$(maxf),fp$(2),os 40
3630 IF rec>recn THEN PRINT"File full":
ENDPROC
3660 PROChead(64,0,"Record: ",rec,4)
3680 FOR I=1 TO f:record$(I,0)=FNinput(
13,L*(I-1)+1,width$(I),"."):NEXT I
3700 PROCwindow2:IF FNask("Confirm (Y/N
): ")<3 THEN PROCwrite(rec,0):rec=rec+1:
PTR#F=0:PRINT#F,rec
3720 PROCwindow1:PROChead(47,0,"",rec-
1,4):PROCwindow2
4960 PROCread(I,0):PROCdisplayl(I)
5240 PROChead(64,0,"Record: ",n,4)
5280 FOR I=1 TO f:PRINTTAB(13,L*(I-1)+
1)record$(I,0):NEXT I
5620 PROCread(n,0):PROCdisplayl(n):PROC
window2
5660 IF A<3 THEN FOR I=1 TO f:record$(I
,0)=STRING$(width$(I),"."):NEXT I:PROCwr
ite(n,0):delete%=-1
6030 IF rec=1 OR NOT delete% THEN 6160

```

```

0600 PROCread(I,0):IF ASC(record$(I,0))
<>46 THEN J=J+1:IF I<J-1 PROCwrite(J-1,
0)
20700 DEF PROCread(n,R):LOCAL I
20720 PTR#F=256+recs*(n-1)
20740 FOR I=1 TO f:INPUT#F,record$(I,R):
NEXT I
20760 ENDPROC
20800 DEF PROCwrite(n,R):LOCAL I
20820 PTR#F=256+recs*(n-1)
20840 FOR I=1 TO f:PRINT#F,record$(I,R):
NEXT I
20860 ENDPROC
21000 DEF PROChead(x,y,msg$,n,w)
21020 PROCinv(1):PRINT#TAB(x,y)msg$;
21040 PRINT SPC(w-LEN(STR$(n)));n
21060 PROCinv(0):ENDPROC

```

```

"<text>" string or text literal
<text> fieldname

```

The contents of a line will normally be left justified on the form but may be right justified or centred by appending /R or /C respectively to the end of a format line specification. For example, a form for printing address labels might be specified as:

```

Header 8,30,2
Line1 "The Managing Director"/C
Line2 ADDRESS1
Line3 @4,ADDRESS2
Line4 @8,ADDRESS3
Line5 POSTCODE/R

```

This would print 8 forms per page consisting of 5 lines of 30 characters, and with 2 blank lines between forms.

ADDING TO THE PROGRAM

This month we are going to add two new features to the program involving three new commands and a modification of an existing command. These are:

```

COMMANDS
CREATE <filename>/<n>
FORMAT <filename>
PRINT <n>

```

The first of these is quite simple and allows a list of all recognised commands to be displayed on the screen. The procedure will automatically cope with any additional commands.

The other three commands are concerned with printing (and displaying) formatted records. Any number of formats can be set up for use with a data file. Each format (or print form) is a special one-record file created with a variation of the CREATE command as above. The value of 'n' specifies the number of lines that the form is to contain (e.g. CREATE form1/5). Once the format file has been created, the format record is generated by using the standard OPEN and ADD commands as for any datafile. You will find that the format record consists of a 'header' line followed by the number of lines specified at create time.

The header line should contain three numbers separated by commas, the number of forms per page, the width of a form (in characters), and the gap or number of lines between forms. Each of the form lines that follow should contain any, or none, of the following elements, in any order, separated by commas:

```

<n>          number of spaces
@<n>         at position n (i.e. tab n)

```

Of the other two new commands, FORMAT is used to select a previously defined format, while PRINT will output records according to the format selected. As with the DISPLAY command, PRINT can be used to print all records or a single specified record, useful for checking a format for correctness. If you haven't a printer immediately available enter *FX5,0 to see the effect of the PRINT command. The printer output then just disappears into a 'sink' (see User Guide page 423).

The additional and amended lines for these enhancements are listed below and may be added to the original program as described for the earlier improvements, which are an essential pre-requisite. The magazine cassette/disc contains the complete program as so far developed.

Next month we will look in more detail at the functions and procedures used to implement the FORMAT and PRINT commands. We will also add further functions to the BEEBUG Filer including a much needed edit record facility. For those interested in programming there is, following the additional lines listed below, a more detailed technical description of the whole program which explains some of the basic ideas and describes how the program has been implemented.

```

1040 maxf=12:X=0:Y=0:format%=0:*FX4,2
1100 DATA 10
1160 DATA DELETE,END,COMMANDS,FORMAT,PR
INT
1670 IF C=5 THEN PROCdisplay(pm$,0)
1700 IF C=8 THEN PROClistc
1710 IF C=9 THEN PROCformat(pm$)

```

```

1720 IF C=10 THEN PROCdisplay(pm$,1)
2550 I=INSTR(p$,"/"):IF I THEN PROCf1(I):GOTO 2820
3000 DEF PROCf1(i):LOCAL I
3020 f=VAL(MID$(p$,i+1))+1:IF f>maxf THEN PRINT"Too many lines":ENDPROC
3040 recn=1:p$=LEFT$(p$,i-1)
3060 FOR I=1 TO f:field$(I)="Line"+STR$(I-1):width$(I)=64:NEXT I
3080 field$(1)="Header":ENDPROC
3100 :
3320 FOR I=1 TO f:INPUT#F,field$,width$(I):field$(I)=FNstrip(field$,"."):NEXT I
4800 DEF PROCdisplay(p$,flag)
4890 IF flag AND NOT format% THEN PRINT "No format":ENDPROC
4900 PROCwindow1:IF flag=0 PROCrecord
4910 IF flag VDU28,0,19,79,4,12,2:form=0
4960 PROCread(I,0):IF flag THEN PROCprint1(I) ELSE PROCdisplay1(I)
4980 IF I<end AND flag<1 THEN G=GET
5000 NEXT I:IF flag VDU12,3
5020 PROCwindow2:ENDPROC
6300 DEF PROClistc:LOCAL I
6320 PROCwindow1:CLS
6340 FOR I=1 TO N:PRINT com$(I):NEXT I
6360 PROCwindow2:PRINT"Press any key to continue";
6380 I=GET:PROCheader:PRINT:ENDPROC
6400 :
6500 DEF PROCformat(p$):LOCAL F,I
6520 IF p$="" THEN PRINT"No file given":ENDPROC
6540 F=OPENUP(p$):IF F=0 THEN PRINT"No such file":ENDPROC
6560 INPUT#F,frec,I,I,ff:IF frec=1 THEN PRINT"No format specified":ENDPROC
6580 PRINT>Loading format ";p$
6600 PTR#F=256:FOR I=0 TO ff-1:INPUT#F,p$:record$(I,1)=FNstrip(p$,"."):NEXT I
6620 p$=record$(0,1)+","":FOR I=0 TO 2:p$=INSTR(p$,""):fp%(I)=VAL(LEFT$(p$,p-1)):p$=MID$(p$,p+1):NEXT I
6640 CLOSE#F:format%=-1:ENDPROC
6660 :
7000 DEF PROCprint1(n):LOCAL I,p,p$,l$,s$
7020 FOR I=1 TO ff-1
7040 l$="":s$="":p$=record$(I,1):IF p$="" THEN p=0:GOTO 7180
7060 p=INSTR(p$,"/"):IF p s$=RIGHT$(p$,1):p$=LEFT$(p$,p-1)
7080 p$=p$+",":REPEAT:p=INSTR(p$,"")
7100 IF ASCp$=34 THEN l$=l$+MID$(p$,2,p-3) ELSE IF ASCp$>48 AND ASCp$<58 THEN l$=l$+STRING$(VAL(LEFT$(p$,p-1))," ") ELSE IF ASCp$=64 THEN l$=l$+STRING$(ABS(VAL(MID$(p$,2,p-1))-LENl$)," ") ELSE l$=l$+FNstrip(FNfield(LEFT$(p$,p-1),0),"")

```

```

7120 p$=MID$(p$,p+1):UNTIL p$=""
7140 l$=LEFT$(l$,fp%(1))
7160 p=fp%(1)-LENl$:IF s$="C" THEN p=p/2 ELSE IF s$="R" ELSE p=0
7180 PRINT SPC;p;l$
7200 NEXT I:PROCline(fp%(2)):form=form+1
7220 IF form=>fp%(0) VDU12:form=0
7240 ENDPROC
7260 :
21200 DEF FNstrip(p$,c$):LOCAL I:I=LENp$+1
21220 REPEAT:I=I-1:UNTIL MID$(p$,I,1)<>c$
21240 =LEFT$(p$,I)
21260 :
21400 DEF PROCline(n):LOCAL I
21420 IF n>0 FOR I=1 TO n:PRINT:NEXT I
21440 ENDPROC
21460 :
21600 DEF FNfield(p$,R):LOCAL I:I=0
21620 REPEAT:I=I+1:UNTIL p$=field$(I) OR I>f
21640 =record$(I,R)
21660 :
31010 IF ERR=17 THEN PROCheader:PRINT"Command aborted":GOTO 140

```

THE FILE STRUCTURE DESCRIBED

A database file will consist of a series of fixed length records, and each record will contain a number of fixed length fields. Field and record lengths need to be fixed in any given data file so that they can be accessed using the PTR# instruction. We also want our program to be able to handle a variety of different data files, each with its own field and record lengths.

One solution to this problem is to reserve space at the start of any file in which to store a description of that file. By always keeping the file description in the same format, the one program can readily handle many different files.

BEEBUG Filer reserves the first 256 bytes in any file for the file description. This is then followed by the records



Diagram 1

themselves as shown in figure 1. The file description contains the position of the next free record in the file (or, if you like, the number of records in use plus 1), the maximum number of records that the file can hold, the size in bytes of each record, the number of fields in each record and the name and length of each field in turn. The structure of the file description record (FDR) is shown in figure 2. Field names, the only variable length items, are limited to 12 characters (14 characters of file space) and the 256 byte FDR provides enough space for up to 12 fields, which is therefore the limit for any BEEBUG Filer database file.

You could increase this by allocating, say, 512 bytes for the FDR but there are other limitations too. The area of the screen used to display a record has a maximum of 17 lines available. This would be the maximum number of fields that could be displayed together. If you want to change this or experiment, then you will need to alter the value of 'maxf' in line 1040 to whatever limit you wish to set, and change 256 to 512 (or whatever) where referenced with PTR#.

COMMAND MECHANISM

BEEBUG Filer is command-driven, with the number of commands and the commands themselves read from DATA statements in PROCsetup at lines 1080 to 1160. These are read into an array for subsequent use. This makes the addition of further commands very easy, as described earlier.

The program operates a simple loop, repeatedly calling the main command procedure (PROCcommand). This procedure, at lines 1500 to 1880, inputs the next command from the keyboard, checks its validity, and depending upon the command entered, calls one of several key procedures.

KEY PROCEDURES AND UTILITY PROCEDURES

The key procedures implemented in the

rec	recn	recs	f	field\$(1)	width%(1)	field\$(2)	width%(2)	field\$(3)	width%(3)	field\$(4)	width%(4)	field\$(5)	width%(5)	field\$(6)	width%(6)	field\$(7)	width%(7)	field\$(8)	width%(8)	field\$(9)	width%(9)	field\$(10)	width%(10)	field\$(11)	width%(11)	field\$(12)	width%(12)	4
6	6	6	6	14	5	14	5	14	5	14	5	14	5	14	5	14	5	14	5	14	5	14	5	14	5	14	5	4

(Up to 12 field names & widths)

Diagram 2

basic program are as follows:

- PROCcreate - create a new datafile
- PROCOpen - open a datafile for use
- PROCadd - add a record to the file
- PROCdisplay - display one or more records on the screen
- PROCdelete - delete a specified record
- PROCclose - close a data file

These key procedures are listed from line 2500 upwards and include a number of supporting procedures and functions. All the key procedures contain a number of checks, displaying appropriate error messages as needed.

There are also several general utility functions and these are listed from line 20000 upwards:

- FNask - seek a Y/N answer
- PROChheader - display main header
- PROCwindow1 - set record display area
- PROCwindow2 - set command display area
- PROCinv - toggle inverse video effect

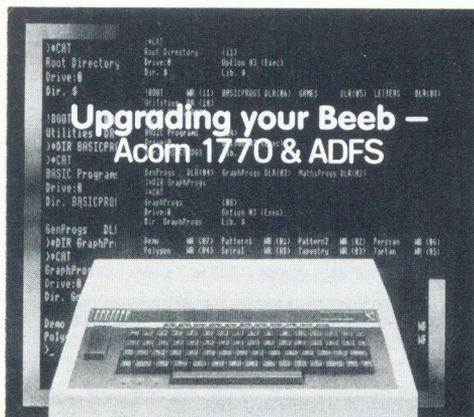
GLOBAL VARIABLES

A number of variables are used throughout the program and these are listed below. All variables that are logically local to any procedure or function are declared as LOCAL to avoid any unforeseen clashes.

<u>Global</u>	<u>Function</u>
rec	Next free record
recn	Maximum number of records
recs	Size of a record in bytes
f	Number of fields
field\$()	Fieldnames
width%()	Fieldwidths
record\$()	Contents of fields for any record
com\$()	List of commands
maxf	Maximum number of fields
open%	Flags if a datafile open
F	Datafile channel number
F\$	Current datafile name
exit%	Flags if exit from program
N	Number of commands in use
C	Number of current command
X	Used to record the position of the cursor in window 2
Y	Set to indicate single or double line spacing depending on number of fields
L	Set to indicate single or double line spacing depending on number of fields
delete%	Flags if any record logically deleted

More technical information will appear next month covering the later additions.





Upgrading your Beeb - Acorn 1770 & ADFS

Acorn now provides upgrades that allow all BBC model B micros to be upgraded with the 1770 disc controller, used in the B+, and with Acorn's Advanced Disc Filing System (ADFS). Is the cost and effort worthwhile? Peter Rochford reports.

Acorn 1770 Disc Interface price £49.95

Acorn ADFS Upgrade price £29.95

**Acorn Computers Ltd, Cambridge Technopark,
645 Newmarket Road,
Cambridge CB5 8PD. 0223-214411.**

THE ACORN 1770 FLOPPY DISC CONTROLLER

The recently introduced BBC B+ is supplied with a disc interface already fitted which utilises the Western Digital 1770 FDC (Floppy Disc Controller). This chip has the advantage of being relatively cheap and enables the use of both single and double density formats.

This interface is now available from Acorn for the Model B, and much cheaper than the older 8271 FDC, but still retaining compatibility with the Acorn DFS.

The kit consists of a small PCB about 2.5 inches square, on which is mounted the 1770 FDC and several support chips. Supplied with this are two chips, two wire links, a 16K DFS ROM, fitting instructions and the Acorn Disc User Guide with extra information on the use of the 1770 DFS.

Fitting the kit involves plugging the PCB into the 8271 (IC78) socket by means of two rows of pins on the top of the board. The two chips supplied are then plugged into their appropriate sockets and the two small links plugged into two other IC sockets. The 1770 DFS ROM must also be installed in a vacant paged ROM socket.

The whole process is quite straightforward thanks to Acorn's excellent documentation, but I did find that the PCB required a fair amount of force to make it secure and was very close to the speech chip sockets when fitted. Acorn tell me that they may include an extra 40 pin DIL socket with future releases of the kit to raise the board up to provide more clearance. If you have an ATPL ROM board, by the way, the kit can be fitted without any problems.

Those who have the 8271 disc interface fitted can achieve installation of the kit in much the same way after the removal of the 8271 FDC from socket IC78. However, if IC78 is soldered in, as in some Beebs, the job would be best left to a dealer unless you are deft with a soldering iron and solder sucker. Issue 2 & 3 boards, fitted with an interface for the first time, need a further modification to the main PCB.

The operation of the 1770 DFS is identical to the DFS 1.2 used with the 8271 FDC with the exception of the *DRIVE command which allows the reading of 40 track discs by an 80 track drive. There are also some additional commands however, which are shown below:

- *CLOSE Closes all sequential files.
- *EX Display info about the specified directory.
- *FORM Format a disc.
- *FREE Display bytes free and files free.
- *MAP Display map of disc free space.
- *ROMS Lists paged ROMS and their type.
- *VERIFY Verify sectors of a disc.

The only problem you may possibly encounter with the 1770 DFS is that of protected commercial discs which write to the 8271 FDC direct. Acorn have provided several types of 8271 emulation in the 1770 DFS to cope with this but give no guarantees that all protected software will be readable.

For those who don't already have a disc interface the new kit will be most

welcome being half the price of the old 8271 interface. It still has all the drawbacks of the DFS, and of course one must bear in mind the question of protected discs. As far as I can see, there is little gain for existing 8271 owners changing to the 1770 FDC, except that they will need it fitted if they want to use Acorn's other new release for the Beeb.

THE ADFS UPGRADE

The Advanced Disc Filing System has been around for some time and is used on the Electron Plus 3 and with the Acorn Winchester. It is now available to Model B owners with the 1770 interface fitted, and of course B+ owners.

For your £29 you get the 16K ADFS ROM, fitting instructions, a user guide and utilities disc. Fitting just requires putting the ADFS ROM into one of the paged ROM sockets with a priority to suit whether you require it as the default DFS on power-up.

It can be used with the 1770 DFS still in the computer and you can move from one to the other when required. However, there are certain problems with this arrangement which I will come to later.

I should mention before going any further, that this review of the ADFS was written without the use of the manual or utilities disc which were both not ready at the time of writing. I did, however, get hold of the ADFS (Advanced Disc Toolkit) ROM from Advanced Computer Products (reviewed BEEBUG Vol.4 No.6) which has versions of the main ADFS disc utilities on it. The only reference

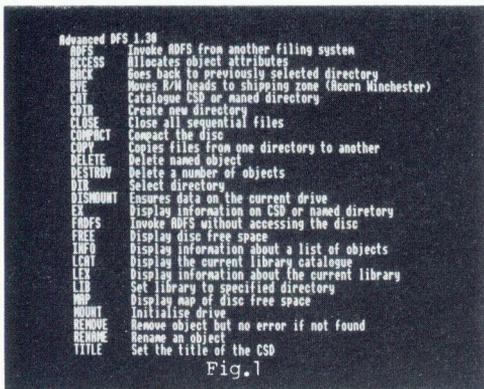


Fig.1

available to me was the Acorn Winchester manual which gives all the ADFS commands. A list of these is shown in Fig.1.

In contrast to the standard Acorn DFS, the ADFS is double density with sixteen sectors per track. This means an increase of 60% in storage but makes the term 'double' density not quite appropriate! If you are using a double-sided 80 track disc you will now have 640K as opposed to 400K with single density.

The limitation of a fixed number of filenames does not occur with the ADFS. It has what is known as a hierarchical filing structure, much the same as that of MS DOS used on many business micros.

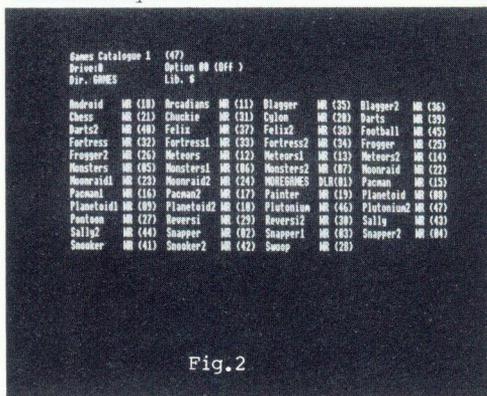


Fig.2

When an ADFS disc is formatted, a root directory \$ is created. This directory may contain up to 47 objects which can be files or directories. Each directory that you create in the root directory can hold 47 objects and again they may be files, or directories. You can carry on nesting directories inside one another to a depth of 127, but there is one snag with this. Getting too carried away can make the whole tree of files and directories rather unwieldy and complicated. The example in Fig.2 shows the catalogue of a directory with 46 files and one directory, which is named in upper case.

Filenames and directories in the ADFS may be up to 10 characters in length. Directories are identified with an R alongside when created and may be given a title of up to 19 characters. This is shown when the disc is catalogued with that directory as the CSD (Currently Selected Directory).

To make things clearer as regards the organisation of an ADFS disc, look at Fig.3. All the objects in upper case are directories and the others are files. To load the file Memol for example with the CSD as \$ would require a file 'pathname' LOAD"LETTERS.BIZLETS.Memol". This is rather cumbersome of course, but if you make the CSD 'LETTERS', you will only have to type LOAD"BIZLETS.Memol". Taking it a stage further, if you now make 'BIZLETS' the CSD you will only have to type LOAD"Memol". Fortunately, the ADFS allows the extensive use of wildcards so you can substitute for the first example LOAD"L*.B*.Memol".

Looking at the list of commands in Fig.1, you will notice some of them are identical to those used with the DFS. There is the absence though of a *DRIVE command. This is because in the ADFS you use the *MOUNT command to select the current drive. With twin drives, there are only two drive numbers, 0 and 1. If the drive is double-sided both sides of the drive are configured as one contiguous surface which can be a distinct advantage, particularly for database work.

Some of the commands in the ADFS, although the same as the DFS, operate in a

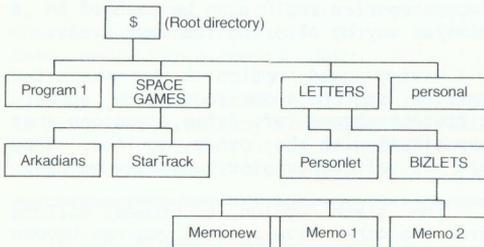


FIG.3 The filing system structure

different way. There are also some obvious omissions such as *BUILD, *WIPE, *LIST, *TYPE, *DUMP and *BACKUP. The bad news is that these have to be loaded from a utilities disc. The formatter and verifier are also on this disc, as are several other programs, one of which is for transferring DFS files to ADFS and vice versa.

This brings me on to a disadvantage of the ADFS that will cause most concern. PAGE with the ADFS is moved up by 1K bringing it to &1D00. Unlike the DFS where you could lower page to &1100 if you did not have any files open, the ADFS will not

allow this. In effect, you have actually lost access to 3K of usable memory. A further complication arises if you have the 1770 DFS in the computer as well as the ADFS. PAGE then goes up to &1F00!

Many of these problems can be overcome with the ADT ROM that I used for this review. This has versions of many of the ADFS utilities instantly available.

CONCLUSION

The ADFS is certainly a very powerful filing system and has many advantages over the DFS. It is fast, provides a greater amount of disc storage, allows two surfaces of a drive to be configured as one and has a structured method of storing and retrieving an unlimited number of files. The ability to have meaningful file and directory names along with the use of wildcards for many of the filing operations, helps to make it a lot more user friendly.

Unfortunately, there are some serious disadvantages, which I have outlined in the course of this review. Personally, I found the ADFS very difficult to get used to at first and I have a suspicion that others will too. With perseverance however, I have reached the stage where there is no way I would go back to the DFS. But...I would not contemplate or advise using the ADFS without the use of a ROM such as the ADT, nor without the use of a second processor, sideways RAM or a shadow RAM board. You will need at least one of the extra RAM options to help you get over the memory problem.

Reduced memory will be your biggest headache with the ADFS and mean a great deal of your existing software will need modification to work. Some will not work at all, particularly those that are memory tight and need to access the filing system. For these you will have to revert back to using the DFS. Don't forget too, that View and Wordwise will have 1K less memory available, particularly important in View if working in 80 column mode.

You will have to weigh up very carefully the pros and cons of the ADFS from your own personal point of view before committing yourself to it as a filing system. Even if you decide against for now, it is clearly the direction in which Acorn are moving.

Personal Diary Manager

Get ready for 1986
with this personal
diary manager
described by
Chris Wragg.

This program provides you with a computerized desk diary, with quick access, plenty of space to record information, and a variety of routines to ensure important dates are not forgotten. As it relies on random access, it is only suitable for disc users.

Before the program can be used, two dummy data files should be set up using the short program, listed on page 23. Running the main program prompts the user to enter the year (1984-9999 only). It will then automatically load the index file and display the main menu. All options but 'Browse' lead to a prompt asking if you want printout.

SIX OPTIONS AVAILABLE

- Select a date
- Select all birthdays
- Select all school dates
- Select all dates to be remembered
- Entries over a given period
- Browse

Initially your diary will be blank, and you will need to use option 1 to make some entries. Enter the date as prompted and the record is loaded. Press 'C' to change the entry, and enter your data.

All the options use the same record display routine, and within this routine you can change, index, or print the record, or press the space bar to continue. There is also one hidden facility - pressing M followed by the space bar will take you back to the Menu.

The index option is very important. The index file stores a record of the entries memorable for some reason. To index an entry, press I and the index keys required. Similarly, to delete a record, you will need to use the index routine to cancel the index record for that day.

There are three index categories: birthdays, school dates, and dates to remember. Not everyone has children at school, and so some users will want to rename this category. To do this, alter the PRINT statements in lines 1250, 1730, and 2750. You may also wish to change the letter pressed from S to another. In this case the INKEY statement in line 2790 would need to be changed too. The other two categories could also be changed in a similar way by altering the same areas.

Having used option 1 to make a few entries, and the index routine to specify different types of dates, you can then experiment with the other options. They are all self-explanatory and easy to use.

The sixth option, to browse, differs from the other five in that you can browse through the diary in either direction, starting from a specified date. To change the direction of browsing just press R while the current page is being displayed followed by the space bar.

SETTING UP THE DATA FILES

In addition to the main program, the short set-up program on page 23 is also required to create the diary and index data files on disc (X.DIARY and X.INDEX). These are 90K and 1.5K, respectively, so readers with 40 track discs may have to use a separate disc for these files.

USING A PRINTER

Readers without Epson-compatible

printers may need to modify line 3230. This reduces line spacing to 32/216 inch, sets the left margin to 6, and the line width to 68 characters.

PROGRAM NOTES

Options 5 and 6 are very similar, and so the same control procedure, PROCprd, has been used for both, the differences being controlled by the value of C%.

The date is calculated with a base of 1984 because this is a leap year, and it conveniently starts on a Sunday, the first day of the diary week. The date entered is used to calculate Y% equivalent to the first day of the given year. L% is used as a flag to indicate a leap year.

Records are held on disc in date order, and 29th February is there as record number 60 whether it is a leap year or not. It is possible to use the same file for any year, but the date of entry is always recorded as the last 10 characters of the record, so there should be no danger of your turning up at a meeting a year early or late!

For brevity and simplicity the index file is composed of single integers made up from the sum of appropriate powers of 2. 8 indicates that a record exists, 1 that it is a birthday, 2 for a school date, and 4 for a special date.

The display format procedure is based on the PROCbox utility from BEEBUG Vol.2 No.8. The entry routine is based on the BEEBUG Workshop of Vol.3 No.1.

```

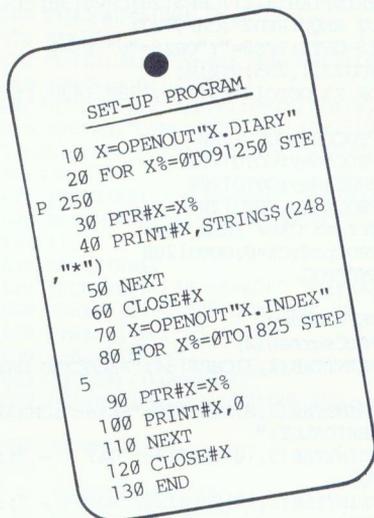
10 REM PROGRAM DIARY
20 REM VERSION B0,5
30 REM AUTHOR C.WRAGG
40 REM BEEBUG DEC 1985
50 REM PROGRAM SUBJECT TO COPYRIGHT
60 :
100 MODE7
110 ONERROR VDU3,7:MODE7:REPORT:PRINT"
at line ";ERL:CLOSE#0:END
120 DIM Index%(366),Day$(6)
130 R$=STRING$(248," "):R1$=STRING$(24
8," ")
140 FORX%=0TO6:READDAY$(X%):NEXT
150 DATA Sunday,Monday,Tuesday,Wednesd
ay,Thursday,Friday,Saturday
160 PROCtitle
170 PROCopenfile:PROCprinter
180 REPEAT

```

```

190 PROCmenu
200 UNTIL Z%=0
210 PROCend
220 :
1000 DEFPROCtitle
1010 PROCbox(2,9,1,4,21)
1020 FORX%=1TO2:PRINTTAB(0,X%+1)CHR$141
;TAB(13,X%+1);CHR$131;CHR$157;CHR$132;"
DIARY ";CHR$156;:NEXT
1030 PRINTTAB(2,8)CHR$130;"Enter year (
not earlier than 1984)""
1040 PRINTTAB(16)CHR$130;:INPUT""Year%
1050 IF Year%<1984 OR Year%>9999 CLS:GO
TO1020
1060 IF Year%=(INT(Year%/4))*4 L%=1 ELS
E L%=0
1070 Y%=Year%-1984
1080 F%=Y%/4;Y%=(Y%+F%-(1<L%))MOD7

```



```

1090 PRINTTAB(6,18)CHR$131;"New Year's
Day: ";Day$(Y%)
1100 VDU23,1,0;0;0;0;
1110 ENDPROC
1120 :
1130 DEFPROCopenfile
1140 I=OPENUP "X.INDEX"
1150 FORX%=1TO366:INPUT#I,Index%(X%):NE
XT
1160 F=OPENUP"X.DIARY"
1170 ENDPROC
1180 :
1190 DEFPROCmenu
1200 CLS:P%=0:VDU23,1,0;0;0;0;
1210 PROCbox(4,1,6,18,38):PROCbox(4,1,1
,5,38)
1220 PRINTTAB(11,3)CHR$134;"DIARY OPT
IONS";

```

```

1230 PRINTTAB(3,8)CHR$134;"1 Select a
given date"
1240 PRINTTAB(3,10)CHR$134;"2 Select a
ll birthdays"
1250 PRINTTAB(3,12)CHR$134;"3 Select a
ll school dates"
1260 PRINTTAB(3,14)CHR$134;"4 Select d
ates to be remembered"
1270 PRINTTAB(3,16)CHR$134;"5 Entries
over a given period"
1280 PRINTTAB(3,18)CHR$134;"6 Browse"
1290 PRINTTAB(3,21)CHR$129;"0 Finish w
ith diary"
1300 Z%=GET-48:IF Z%<0 OR Z%>6 GOTO1300
1310 IFZ%=0 THEN1430 ELSE IF Z%=6 THEN1
350
1320 PRINTTAB(2,6+2*Z%)CHR$136;TAB(5,6+
2*Z%)CHR$137;
1330 PRINTTAB(3,21)CHR$129;CHR$136;"IS
PRINTOUT REQUIRED?";CHR$137;
1340 Z$=GET$:IFZ$="Y"ORZ$="y" P%=1
1350 VDU23,1,255;0;0;0;
1360 ON Z% GOTO1370,1380,1390,1400,1410
,1410
1370 PROCday:GOTO1200
1380 PROCbday:GOTO1200
1390 PROCsday:GOTO1200
1400 PROCxday:GOTO1200
1410 IFZ%=5 C%=0 ELSE C%=1
1420 PROCprd:C%=0:GOTO1200
1430 ENDPROC
1440 :
1450 DEFPROCday
1460 PROCscreen(2)
1470 PRINTTAB(2,3)CHR$134;"SELECTED DAY
";
1480 PRINTTAB(2,8)CHR$130;"Enter";CHR$1
29;"NUMERICALLY:"
1490 PRINTTAB(5,10)CHR$130;"DAY - ";;
INPUT""D%
1500 PRINTTAB(5,12)CHR$130;"MONTH - ";;
INPUT""M%
1510 PROCcheck(D%,M%):IF F%=1 GOTO1460
1520 RESTORE1600
1530 IFM%=1 THEN T%=D%:GOTO1570
1540 T%=0
1550 FORX%=1TOM%-1:READ N%:T%=T%+N%:NEX
T
1560 T%=T%+D%
1570 Day%=(Y%+T%+6-(L%=0)*(M%>2))MOD7
1580 PRINTTAB(16,3)CHR$131;Day$(Day%);"
";D%:"";M%:"";Year%
1590 PROCload(T%)
1600 DATA 31,29,31,30,31,30,31,31,30,31
,30,31
1610 ENDPROC
1620 :
1630 DEFPROCbday
1640 PROCscreen(1)
1650 PRINTTAB(2,3)CHR$130;"BIRTHDAY:";

```

```

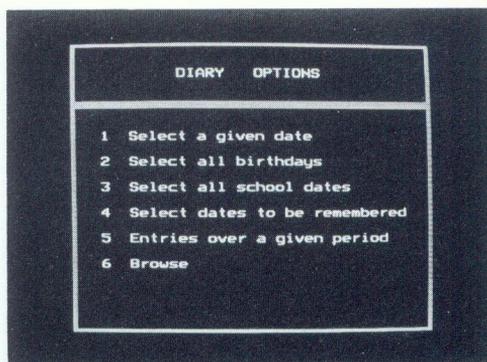
1660 FORX%=1TO366
1670 IFIndex%(X%)<>INT(Index%(X%)/2)*2
PROCdate:PROCload(X%)
1680 NEXT
1690 ENDPROC
1700 :
1710 DEFPROCsday
1720 PROCscreen(6)
1730 PRINTTAB(2,3)CHR$133;"SCHOOL DATES
";
1740 FORX%=1TO366
1750 IF(Index%(X%)>9ANDIndex%(X%)<12)OR
(Index%(X%)>13) PROCdate:PROCload(X%)
1760 NEXT
1770 ENDPROC
1780 :
1790 DEFPROCxday
1800 PROCscreen(5)
1810 PRINTTAB(2,3)CHR$129;"SPECIAL DATE
!";
1820 FORX%=1TO366
1830 IFIndex%(X%)>11 PROCdate:PROCload(
X%)
1840 NEXT
1850 ENDPROC
1860 :
1870 DEFPROCprd
1880 PROCscreen(3)
1890 IF C%=0 PRINTTAB(3,3)CHR$131;"PERI
OD FROM"; ELSE PRINTTAB(3,3)CHR$131;"BRO
WSING MODE - R : reverse";TAB(19,4)CHR$1
31;"M : exit to Menu";
1900 MM%=0
1910 PRINTTAB(3,8)CHR$131;"Enter";CHR$1
29;"NUMERICALLY:"
1920 PRINTTAB(5,10)CHR$131;:INPUT"FIRST
day:"D%
1930 PRINTTAB(5,11)CHR$131;:INPUT"Month
:"M%
1940 PROCcheck(D%,M%):IF F%=1 GOTO1880
1950 IF C%=1 GOTO1990
1960 PRINTTAB(5,13)CHR$134;:INPUT"LAST
day:"E%
1970 PRINTTAB(5,14)CHR$134;:INPUT"Month
:"O%
1980 PROCcheck(E%,0%):IF F%=1 PRINTTAB(
5,13)STRING$(14," ");TAB(5,14)STRING$(14
," ");TAB(10,16)STRING$(14," ");GOTO1960
1990 RESTORE1600
2000 IFM%=1 THEN T%=D%:GOTO2040
2010 T%=0
2020 FORX%=1TOM%-1:READ N%:T%=T%+N%:NEX
T
2030 T%=T%+D%
2040 Day%=(Y%+T%+6-(L%=0)*(M%>2))MOD7
2050 IFC%=1 S%=366:GOTO2140
2060 PRINTTAB(16,3)CHR$131;Day$(Day%);"
";D%:"";M%:"";Year%
2070 RESTORE1600
2080 IFO%=1 THEN S%=E%:GOTO2120

```

```

2090 S%=0
2100 FORX%=1TOO%-1:READ N%:S%=S%+N%:NEX
T
2110 S%=S%+E%
2120 Day2%=(Y%+S%+6-(L%=0)*(O%>2))MOD7
2130 PRINTTAB(9,4)CHR$131;"TO      ";Day
$(Day2%);" ";E%;" ";O%;" ";Year%
2140 Q%=1:X%=T%
2150 REPEAT
2160 IFIndex%(X%)<0 last%=X%:Day%=(Y%+
X%+6-(L%=0)*(M%>2))MOD7:PROCload(X%)
2170 X%=X%+Q%
2180 UNTIL X%<1 OR X%>S% OR MM%=1
2190 IF MM%=0 AND C%=1 THEN VDU7:PRINTT
AB(10,7)"FINAL ENTRY - R FOR REVERSE";:P
ROCpause(50):X%=last%:GOTO 2150
2200 ENDPROC
2210 :
2220 DEFPROCload(t%)
2230 VDU28,3,16,36,7,23;11,0;0;0:CLS
2240 PTR#F=(t%-1)*250
2250 INPUT#F,R$
2260 PRINTTAB(0,2)R$;TAB(20,9)Day$(Day%
);
2270 IFP%=1 PROCprint
2280 VDU28,2,22,36,19,12
2290 PRINTTAB(3,0)CHR$131;"Press";SPC4;
"C";SPC5;"to change"TAB(12)CHR$131;"I";
SPC5;"to index"TAB(12)CHR$131;"P";SPC5;
"to print"TAB(12)CHR$134;"SPACE";CHR$13
1;"to continue";
2300 VDU28,3,16,36,7
2310 REPEAT
2320 IF INKEY-83 VDU23;11,255;0;0;0:PRO
Cchange(t%):VDU23,1,0;0;0;0;
2330 IF INKEY-56 PROCprint
2340 IF INKEY-38 PROCindex(t%)
2350 IF INKEY-52 AND C%=1 Q%=Q%:PRINTT
AB(20,1)"REVERSE";:PROCpause(20)
2360 IF INKEY-102 MM%=1:PRINTTAB(20,1)"
EXIT to MENU";
2370 UNTIL INKEY-99
2380 VDU26
2390 ENDPROC
2400 :
2410 DEFPROCdate
2420 RESTORE1600:M%=1:D%=X%
2430 READN%
2440 IFD%<=N% GOTO2500
2450 REPEAT
2460 M%=M%+1:D%=D%-N%
2470 READN%
2480 UNTILD%<=N%
2490 IFZ%>4 THEN2530
2500 VDU26:PRINTTAB(16,3)STRING$(20, "
");
2510 Day%=(Y%+X%+6-(L%=0)*(M%>2))MOD7
2520 PRINTTAB(16,3)CHR$131;Day$(Day%);"
";D%;" ";M%;" ";Year%;
2530 ENDPROC

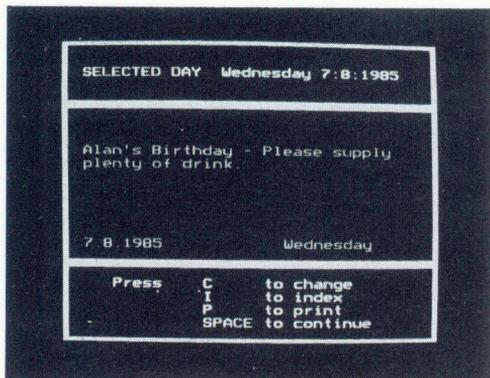
```



```

2540 :
2550 DEFPROCchange(t%)
2560 PROCpause(20)
2570 *FX15,1
2580 VDU30:PRINT"Enter new details:"
2590 R$=F$IN(0,1,238)
2600 VDU23;11,0;0;0;0;:PRINTTAB(0,0)CHR
$129;CHR$136;"CORRECT?";SPC7;CHR$137:Z$=
GET$:CLS:IFZ$<<"Y"ANDZ$<<"y" PRINTTAB(0,
2)R$:GOTO2580
2610 IFLEN(R$)<238 R$=R$+" ":GOTO2610
2620 Ref$=STR$(D%)+". "+STR$(M%)+". "+STR
$(Year%)
2630 R$=R$+Ref$
2640 IFLEN(R$)<248 R$=R$+" ":GOTO2640
2650 PTR#F=(t%-1)*250:PRINT#F,R$
2660 PTR#I=(t%-1)*5
2670 Index%(t%)=8
2680 PRINT#I,Index%(t%)
2690 PRINTTAB(0,2)R$;TAB(20,9)Day$(Day%
);
2700 ENDPROC
2710 :
2720 DEFPROCindex(t%)
2730 VDU28,2,22,36,19,12
2740 Index%(t%)=8:B%=0:R%=0:W%=0
2750 PRINTCHR$130;"Press"CHR$134;" B";
CHR$130;"= Birthday";SPC6;" ";CHR$134;"S"
;CHR$130;"= School "CHR$134;" X";CHR$13
0;"= Special date ";CHR$134;"C";CHR$130
;"= Cancel "TAB(6)CHR$134;"<RETURN>";CH
R$130;"to continue";
2760 *FX15,1
2770 REPEAT
2780 IF INKEY-101 AND B%=0 Index%(t%)=I
ndex%(t%)+1:B%=1:PRINTTAB(1,1)CHR$136;TA
B(18,1)CHR$137;
2790 IF INKEY-82 AND R%=0 Index%(t%)=In
dex%(t%)+2:R%=1:PRINTTAB(18,1)CHR$136;TA
B(31,1)CHR$137;
2800 IF INKEY-67 AND W%=0 Index%(t%)=In
dex%(t%)+4:W%=1:PRINTTAB(1,2)CHR$136;TAB
(18,2)CHR$137;

```



```

2810 IFINKEY-83 Index%(t%)=0:MM%=1:PRIN
TTAB(19,2)CHR$136;" CANCELLED";CHR$137;:
PROCpause(120)
2820 UNTIL INKEY-74 OR Index%(t%)=0
2830 PTR#I=(t%-1)*5:PRINT#I,Index%(t%)
2840 *FX15,1
2850 CLS
2860 PRINTTAB(3,0)CHR$131;"Press";SPC4;
"C";SPC5;"to change"TAB(12)CHR$131;"I";
SPC5;"to index"TAB(12)CHR$131;"P";SPC5;
"to print"TAB(12)CHR$134;"SPACE";CHR$13
1;"to continue";
2870 VDU28,3,16,36,7
2880 ENDPROC
2890 :
2900 DEFPROCcheck(d%,m%)
2910 F%=0
2920 IFd%<1 OR d%>31 OR m%<1 OR m%>12 F
%=1:GOTO2970
2930 IFd%<29 GOTO2980
2940 IFm%=2 AND d%=29 AND L%=0 F%=1:GOT
O2970
2950 IFm%=2 AND d%>29 F%=1:GOTO2970
2960 IFd%=31 AND (m%=4 OR m%=6 OR m%=9
OR m%=11) F%=1
2970 IF F%=1 PRINTTAB(10,16)CHR$133;"NO
SUCH DAY!!":PROCpause(100)
2980 ENDPROC
2990 :
3000 DEFFNin(U%,V%,W%)
3010 B%=0:R1$="" :PRINTTAB(U%,V%);
3020 K%=GET
3030 IF K%=13 =R1$
3040 IF K%=127 PROCdel:K%=0
3050 IF K%<32 K%=0
3060 IF K% PROCadd
3070 GOTO3020
3080 :
3090 DEFPROCdel:IFB%=0 VDU7:ENDPROC
3100 B%=B%-1:R1$=LEFT$(R1$,B%):VDU8,ASC
" ",8:ENDPROC
3110 :

```

```

3120 DEFPROCadd
3130 IFB%=W% VDU7:ENDPROC
3140 B%=B%+1:R1$=R1$+CHR$K%:VDUK%:ENDPR
OC
3150 :
3160 DEFPROCscreen(z%)
3170 CLS:PROCbox(z%,1,18,6,38):PROCbox(
z%,1,6,12,38):PROCbox(z%,1,1,5,38)
3180 ENDPROC
3190 :
3200 DEFPROCprinter
3210 REM Setup for Epson RX80
3220 REM Margins set for width of 68 ch
aracters
3230 VDU2,1,27,1,51,1,32,1,27,1,108,1,6
,1,27,1,81,1,74,3
3240 ENDPROC
3250 :
3260 DEFPROCprint
3270 *FX3,10
3280 VDU2,10,10,13
3290 PRINTDay$(Day%);" ";RIGHT$(R$,10)
3300 PRINTLEFT$(R$,238)
3310 VDU3
3320 *FX3,0
3330 ENDPROC
3340 :
3350 DEFPROCpause(t%)
3360 T=TIME:REPEAT UNTIL TIME>T+t%
3370 ENDPROC
3380 :
3390 DEFPROCend
3400 CLOSE#F
3410 CLOSE#I
3420 VDU2,1,27,1,64,3
3430 CLS
3440 VDU23,1,255,0;0;0;0;
3450 END
3460 :
3470 DEFPROCbox(c%,x%,y%,h%,w%)
3480 LOCAL c$,i%,y1%
3490 y1%=y%+h%-1
3500 FORi%=y%TOy1%
3510 PRINTTAB(x%-1,i%);CHR$(145+c%);
3520 IFi%=y% PRINT CHR$183; ELSE IFi%=y
1% PRINT CHR$245; ELSE PRINT CHR$181;CHR
$130;
3530 IFi%=y% PRINT STRING$(w%-2,CHR$163
);
3540 IFi%=y1% PRINT STRING$(w%-2,CHR$24
0);
3550 IF i%=y% c$=CHR$235 ELSE IF i%=y1%
c$=CHR$250 ELSE c$=CHR$234
3560 IF i%>y% AND i%<y1% PRINTTAB(x%+w%
-2,i%);CHR$(145+c%);
3570 PRINTTAB(x%+w%-1,i%);c$;
3580 NEXT
3590 ENDPROC

```



Sure Save Utility

If you have ever cursed the time you inadvertently lost a program, by saving another with the same name, then Alan Webster's latest disc utility may be a lifesaver.

This short but useful routine for disc users will cure the well-known headache, that of losing a valuable file by accidentally saving a new file with the same name over the top of it.

Once the sure save routine is installed in your machine, and you save a file to disc (using SAVE or *SAVE) that has the same name as one already present, you will be asked the question 'Replace old file (Y/N)'. In response to the question you can reply 'Y' to over write the old file, or 'N' to abandon the save (this utility is similar to the one found in Wordwise Plus). The routine accepts only 'Y' or 'N' in response, but these can be in upper or lower case.

Type the program in, save it away (making sure that you don't use a filename already present on the disc!), and run it. The program will assemble the machine code and print a line of text at the end. To save the machine code just copy this statement and press Return at the end.

To use the routine at any time, simply type *SURE <return> to load from disc and run. The routine will then remain active until the Break key is pressed. To restart the routine after Break has been pressed, just type CALL &900 <return>.

PROGRAM NOTES

- Lines 130-170 Re-direct FILEV vector
- Line 210 Test for 'save'
- Lines 220-230 Address of filename
- Lines 240-260 Read catalogue information for file
- Line 270 File not present. Save it
- Lines 280-290 Print 'Replace' message
- Line 300 Beep and wait for input
- Line 310 Test for escape and change to upper case
- Line 320 Test for 'Y' or 'N'

- Line 340 Execute command
- Line 350 Escape error message
- Line 370-380 Don't execute command

```

10 REM PROGRAM SURE SAVE
20 REM VERSION B0.4
30 REM AUTHOR Alan Webster
40 REM BEEBUG DECEMBER 85
50 REM PROGRAM SUBJECT TO COPYRIGHT
60 :
100 MODE 3
110 FOR PASS=0 TO 3 STEP 3
120 P%=&900:[OPT PASS
130 .start
140 LDA#212:STA#70:LDA#213:STA#71
150 LDA#filev MOD 256:STA#212
160 LDA#filev DIV 256:STA#213
170 RTS
180 .filev
190 STA#72:STX#73:STY#74
200 PHA:TYA:PHA:TXA:PHA
210 LDA#72:CMP#5:BCSend2
220 LDY#0:LDA(&73),Y:STABlock
230 INV:LDA(&73),Y:STABlock+1
240 LDX#block MOD 256
250 LDY#block DIV 256
260 LDA#5:JSR&FFDD
270 CLC:CMP#0:BEQend2
280 LDY#255:.Loop:INV:LDAmess,Y
290 JSR&FFEE:CMP#63:BNEloop
300 .quest:LDA#7:JSR&FFEE:JSR&FFEE0
310 BIT&FF:BMIescape:CLC:AND#223
320 CMP#78:BEQreturn:CMP#89:BNEquest
330 JSR&FFEE:.end:JSR&FFEE7:.end2
340 CLC:PLA:TAX:PLA:TAY:PLA:JMP (&70)
350 .escape:LDA#126:JSR&FFF4:BRK
360 BRK:OPT FNstring("<Escape>"):BRK
370 .return:JSR&FFEE:LDA#7:JSR&FFEE
380 PLA:TAX:PLA:TAY:PLA
390 BRK:BRK:OPT FNstring("File not sav
ed"):BRK
400 .mess:OPT FNstring("Replace old fi
le (Y/N) ?")
410 .block:OPT FNstring(STRING$(20,"*")
))
420 ]
430 NEXT
440 PRINT;"0";~P%;TAB(14);"End of code
"
450 PRINT" *SAVE SURE 900 ";STR$~P%;"
900 900"
460 END
470 :
1000 DEFFNstring(A$)
1010 IF PASS=3 PRINT;"0";~P%;TAB(14);"F
Nstring("";A$;")"
1020 FORM%=1:TOLen(A$)
1030 ?(P%+M%-1)=ASC(MID$(A$,M%,1))
1040 NEXT:P%=P%+M%-1
1050 =0

```

Title : Match Day
Supplier : Ocean Software
Price : £9.95
Reviewer : Alan Webster
Rating : ****

If you're missing televised football this year then Ocean have just the answer. Not only does Match Day give thrilling action on your screen, it allows you to take part

Title : Confuzion
Supplier : Incentive Software
Price : £6.95
Reviewer : Geoff Bains
Rating : **

After all the publicity, you would be forgiven for thinking that Confuzion is something really special. It isn't. However, it is quite a good game that

Title : DeathStar
Supplier : Superior Soft
Price : £9.95 (£11.95 on disc)
Reviewer : Alan Webster
Rating : ****

DeathStar is a superbly colourful and frantic game. It has all the right ingredients to make it a classic arcade game, including speed and excellent graphics.

Title : Tomahawk
Supplier : Kansas
Price : £5.00
Reviewer : Daniel Gaster
Rating : *

Tomahawk is the name of the plane you have to fly over land and sea, bombing enemy ships, tanks, bridges, roads and buildings.

as well. There is also the added bonus of no referee to blow his whistle for fouls or to give 'harsh' decisions.

Match Day is a 3D representation of soccer featuring excellent graphics and a nice 'Match of the Day' tune at the start of each game. Options are available for a one player versus computer game, or a two player game.

You define your own keys for up, down, left, right and kick and you are in control of your player nearest to the

has some unusual features to it. You control a spark (!) travelling along miles of fuzewire (!!) and have to blow up the Confuzion Bombs (!!!) lying around a massive factory.

So far, so ordinary. However, the game is really a series of interactive sliding block puzzles. The fuzewire is laid in patterns across the sliding blocks and you must manoeuvre these so that the spark runs into the confuzion bombs along the outside. Needless to say you have only a

DeathStar is one of those annoying arcade games. You can't leave it alone, but the noise drives you insane! Why don't software companies put sound on/off options into all of their games?

The object in DeathStar is to roam the galaxy, collecting starbombs and shooting the warriors and workers. Once you have collected enough starbombs you can set about the more difficult task of destroying the DeathStar. This is gradually made up by the workers during

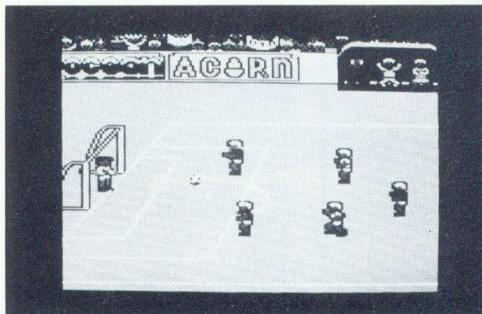
The screen depicts the plane from above with the landscape scrolling below it. The objects on this are randomly scattered and vary from land to sea. Many fire missiles that home in towards the plane and must be spotted early to be avoided, though they are very difficult to see against the background at times. There are also runways at various places where extra points can be gained by flying low along them.

Your plane can be moved up and down (as well as left and right) and this is

ball. There are seven players on each team including the goalkeeper who bends his legs when he is ready for action.

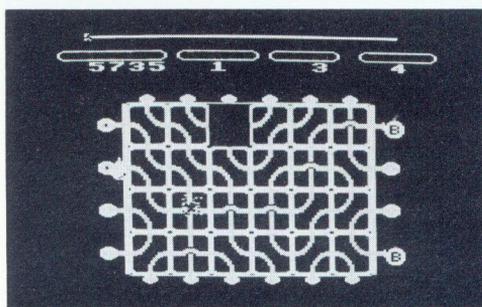
The computer controls the players very well and uses good tactics for playing. The heaviest defeat I have inflicted over the computer is 4-1, and likewise the computer has beaten me 3-0.

Overall, if you have £10 to spare this Christmas, Match Day would make an ideal present for football fanatics everywhere.



limited time to do this and (initially) only five sparks to destroy all the bombs in the factory. The evil owners of the Bomb factory have installed a sprinkler system on some of the factory levels. Your spark is extinguished on contact with water and so this is a hazard to avoid.

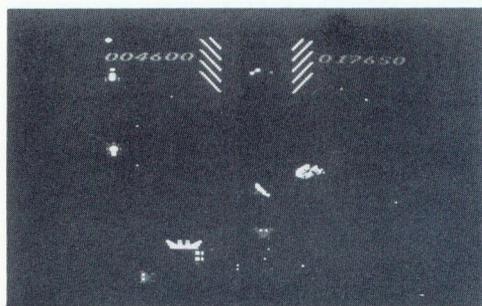
Confuzion is both fairly addictive and taxing on the grey cells. Planning your route across the puzzles and avoiding the water droplets is not easy. It isn't a classic by any means, just quite good fun.



the game, and when complete it will come straight for you.

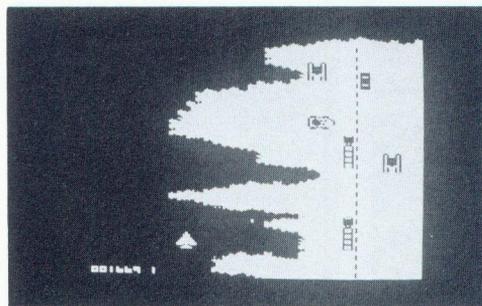
To kill the DeathStar you must destroy all 20 bits without being sucked into its depths by a powerful force field. After destroying the DeathStar you progress onto a bonus screen, then to the next, more difficult screen.

DeathStar is a genuine arcade game, and should sell well with the shoot-em-up addicts. If you do buy it, a set of earplugs would not go amiss.



snown on the screen by the size of the plane changing. The gradual descent of a dropped bomb means that the timing has to be just right to hit an object. When flying at the higher two of the three flight levels the time for a bomb to descend means that luck is the major factor in scoring.

After a few attempts Tomahawk holds little challenge as it seems to go on without any specific goal in mind. Despite its relative cheapness, I cannot recommend this game.



Title : Chicane
Supplier : Kempston
Price : £9.95
Reviewer : Geoff Bains
Rating : ***

Chicane was released at much the same time as Acornsoft's Revs and as such came in for a lot of flack when compared against that epic. Chicane is nowhere near

the standard of Revs, either in programming or realism. However, it does provide a good racing game for the Beeb.

The format is standard enough. You first have to qualify and then you pit your skills against the other drivers in the race proper. The controls are simple to master. There's only five of them. This means that you don't need a degree in keyboard manipulation to have fun with this game, unlike Revs.

The graphics are good. You view the

Title : Sabre Wolf
Supplier : Ultimate Play the Game
Price : £9.95
Reviewer : Mitch
Rating : ****

Graphic adventures seem to be the vogue at present. The title music and graphics of Sabre Wolf are very impressive and the leaflet, listing the many monsters and

objects, intriguing. When the game is loaded, the screen clears to show our gallant adventurer standing in a jungle clearing surrounded by minutely detailed foliage. Then, faster than a dragon up a drainpipe, I was set upon by a host of creatures who made me very dead!

The object of this game is to tramp around a jungle maze picking up objects and avoiding the various monsters which materialize around you. You are provided with a trusty sabre with which you must constantly hack at everything that moves.

Title : Alien 8
Supplier : Ultimate
Price : £9.95
Reviewer : Daniel Gaster
Rating : ****

Alien 8 is one of a recent series of conversions by Ultimate of old Spectrum classics. The game is an arcade/graphic adventure. Included in the package

is a glossy instruction leaflet which sets the scene of the game with a story but does little to tell you the purpose of the game. Briefly, this is to pick up a variety of objects (only three at a time) and restore them to their correct rooms.

The rooms are presented on the screen in two colour, 3D pictures and can contain platforms and objects on many different levels along with sliding platforms, disintegrating platforms, monsters (such as clockwork and dalek mice), and more. The size of the Alien8 arena is quite

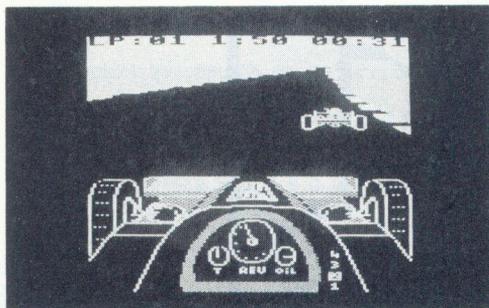
Title : Estra
Supplier : Firebird
Price : £2.50
Reviewer : Daniel Gaster
Rating : **

Estra is one of the new cheap range of games brought out by British Telecom. You have to collect the 24 pieces of the sacred statue of Estra, scattered around

the edge of the screen and deposit them within the central force field.

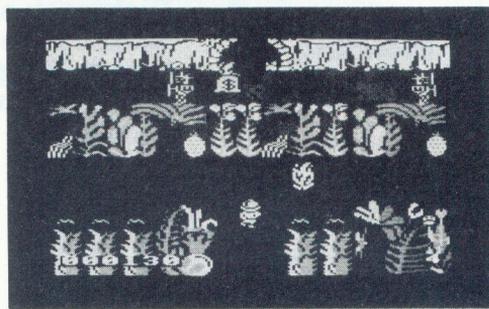
Various nasties try to prevent your achieving this. Between each fragment of statue is a small blob which occasionally moves towards the central forcefield. Other creatures bounce around the edge of the screen, and a crab-like monster follows you everywhere. When your energy runs low an extra supply pack appears within a deadly floating forcefield, though the positioning of these can make it impossible to get the pack!

track over the car's dashboard and have a good view of the speed and gear indicators, the rear view mirrors, and the track ahead. Unlike more realistic simulations, when you crash into the roadside signs or other cars you just start up again and carry on. It's funny, but in a crash the other guy always seems to just go speeding off, leaving you to shift down gears and enter the race again. However, Chicane is generally within everyone's grasp - financially and skilfully - and great fun with it.



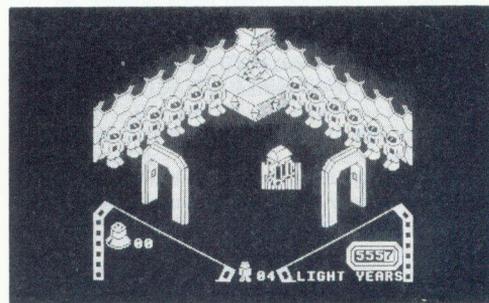
Success depends on lightning reactions and the ability to map a maze whilst being pursued by three charging hippos.

To assist you are the multi-coloured orchids which magically bloom for short periods in each frame. Running through the different colours gives you various powers which last for a few nail-biting seconds. At least it would be nail-biting if I could release the keys for an instant. The sound and graphics are of a very high standard and this game deserves all the praise it has received.



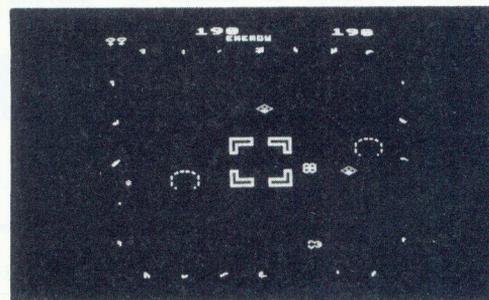
amazing. Almost every room presents a challenge and requires ingenuity and quick reflexes to win through.

The robot you control can move behind objects and can completely vanish from view! Unfortunately, the game tends to slow down when this happens and it speeds up when passing through a simple room. Despite this, I would recommend Alien 8 to anyone wanting a challenge. A truly top quality game.



The game is fast moving and smooth and the plot well thought out. The sound effects are above average and pretty well all the colours are used to good effect. Although the aim seems easy, the screens quickly become very challenging as more monsters appear and the pace speeds up.

Instructions are only included on one side of the cassette so if they are not required the game can be loaded much quicker. I enjoyed playing this game and it is good value for money.



1st course

Colour Blending

If you think the Beeb is only capable of displaying 8 colours then think again. Alan Dickinson shows how to blend these colours together to produce many intermediate shades.

Using graphics on the BBC micro, and colour graphics in particular, can offer a wealth of programming opportunities with a fascination of its own. However, the range of colours may at first sight seem limited compared with some other machines. It is not difficult, though, to mix the standard colours to produce even more shades as you will see.

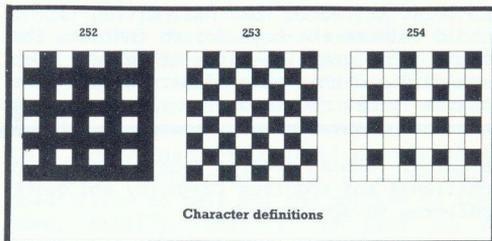
In the non-teletext modes, modes 0-6, the Beeb is capable of displaying 8 different colours and 8 flashing combinations of colour, although only one of those modes, mode 2, allows all 16 colour options to be displayed simultaneously. The other modes have fewer colour choices, but higher resolution.

Each point on the Beeb screen can be coloured independently of adjacent points, and this feature enables us to create a range of shades by weaving together two or more colours. On a high resolution monochrome monitor, the result of such weaving would be no more than a textured area of the screen, but on a lower resolution monitor, or a television set, the colours blend to form new shades and hues. The effectiveness of the result will depend on the colours used, and the quality of your video.

The program set out here illustrates the simplest way of weaving together two colours. To try it out just enter the program, save to cassette or disc if you

wish to keep a copy, and type RUN. It is set up to use mode 1, but you can try it in other modes simply by changing the first line. Of course, it won't work in mode 7, which uses entirely different screen handling techniques.

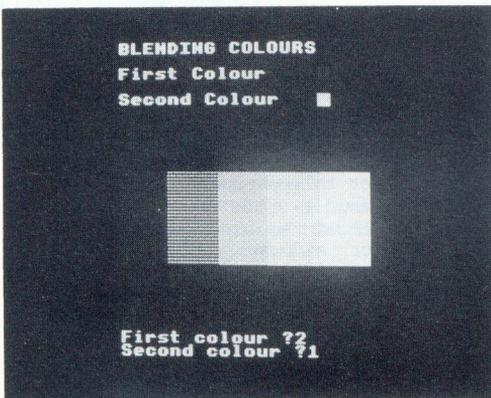
The first few lines of VDU23 commands define a set of five special characters, assigned ASCII values 251 to 255, which are used to form the mixing patterns. They vary from a complete block of foreground colour (255) to a complete block of background colour (251), with a graduation of lattices in between (252, 253 and 254 as shown).



The use of these special characters is demonstrated by the program. The pattern is first drawn so that the foreground is in logical colour 1 (red), and the background is in logical colour 2 (yellow). Five adjacent areas of the screen, each 7 characters high by 4 wide, are filled in using each character in turn (lines 330 to 380). Once the basic display is complete, a text window is defined to prevent the pattern being scrolled off the screen, and the final section of code (lines 490 to 550) switches the logical colours 1 and 2 to different physical colours of your choice using the VDU19 command (see 'Beginners' article in Vol.3 No.4 for more information on this). Some of the intermediate colours formed are quite tasteful, whilst others are startling and ghastly. White, (colour 7), mixes well with most colours as does black, (colour 0). Remember that although you are using a four colour mode, you can

input the codes for any of the 16 physical colours, numbered 0-15.

This program provides a useful testbed for working out the best colour mixes for any particular purpose. In your own programs, the same character definitions could be used to colour in any area that can be defined in character terms. The same principles for mixing colours can also be used with the MOVE, DRAW and PLOT instructions. For example, you could draw a grid of vertical and horizontal lines in one colour over a background of a different colour. As with many of the visual effects that can be produced on the Beeb, experimenting with different ideas and techniques can be well worth while.



```

10 REM Program BLENDER
20 REM Version B1.0
30 REM Author Alan Dickinson
40 REM BEEBUG December 1985
50 REM Program subject to copyright
60 :
100 MODE1
110 :
120 REM DEFINE CHARACTERS USED
130 REM TO MIX COLOURS.

```

```

140 :
150 VDU23,251,255,255,255,255,255,255,
255,255
160 VDU23,252,85,255,85,255,85,255,85,
255
170 VDU23,253,85,170,85,170,85,170,85,
170
180 VDU23,254,85,0,85,0,85,0,85,0
190 VDU23,255,0,0,0,0,0,0,0,0
200 :
210 REM DRAW SCREEN WITH GRADUATION
220 REM OF COLOUR FROM COLOUR1 TO 2
230 :
240 CLS
250 PRINTTAB(0,2)"BLENDING COLOURS"
260 PRINTTAB(0,4)"First Colour"
270 PRINTTAB(0,6)"Second Colour"
280 COLOUR130
290 COLOUR1
300 PRINTTAB(16,4)CHR$251;
310 PRINTTAB(16,6)CHR$255;
320 :
330 FOR Y%=12 TO 18
340 PRINTTAB(0,Y%);
350 FOR X%=0 TO 4
360 PRINTSTRING$(4,CHR$(251+X%));
370 NEXT X%
380 NEXT Y%
390 :
400 COLOUR 3
410 COLOUR 128
420 :
430 REM SET WINDOW TO PROTECT PATTERN
440 REM AND SWITCH COLOURS IN PATTERN
450 REM BY ALTERING PALETTE (VDU19'S)
460 :
470 VDU28,0,31,19,24
480 :
490 REPEAT
500 CLS
510 INPUT"First colour ",F%
520 INPUT"Second colour ",B%
530 VDU19,1,F%,0,0,0
540 VDU19,2,B%,0,0,0
550 UNTIL FALSE
560 END

```



HINTS HINTS HINTS HINTS HINTS HINTS HINTS

STIPPLED LINES - C.J. Collins

To plot an area with alternate pixels in alternate colours, the quickest method is to use the dotted line PLOT codes (16 to 23) and shift every other line one pixel horizontally.

DNFS ADDRESSES - James Medcalf

Whereas the Acorn DFS stores the load address and length of a program just loaded at &BE and &C2, respectively, the DNFS stores these at locations two bytes lower - at locations &BC and &C0.



Upgrade your B+ to 128K

Acorn has not only released the 128K version of the B+, but all existing B+ owners can upgrade their machine to this memory size with Acorn's upgrade kit. Geoff Bains investigates.

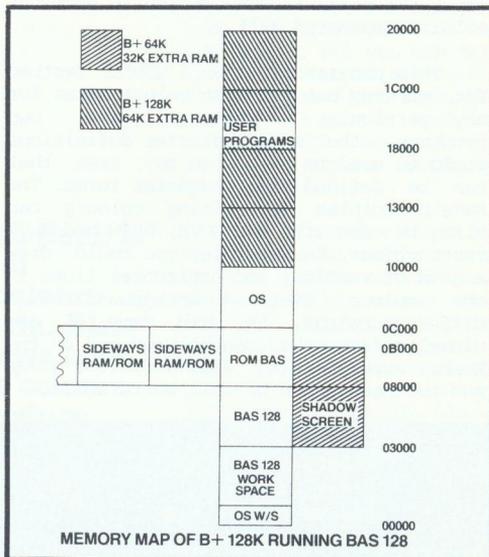
For once, Acorn has not been left behind. Amidst the current surge of forthcoming 128K home computers, Acorn has announced a 128K B+ and an upgrade for the 64K B+. The B+ 64K upgrade costs £39.95 including fitting. It must be fitted by a dealer (such as the new BEEBUG shop).

The upgrade circuit board contains four chips (two 32K RAM chips and a couple of control chips) and sits vertically on the extreme right hand edge of the B+'s main board. You also get a disc, a slim manual, and a new 1770 DFS chip with utilities to make use of the extra 64K. However, Acorn was not able to tell us where these utilities will come from if you purchase an ADFS upgrade (see page 19 of this issue) to replace your 1770 DFS.

Acorn claims that the expansion socket the upgrade board uses was never meant for this. It was meant to take an alternative type of 64K RAM if supplies of the chips on-board ran out. However, some bright spark thought that using it to add another 64K would be a good idea. Judging by the mess of ten flying cables that the upgrade requires, this story seems to be true.

The extra 64K of RAM is basically four sideways RAM blocks and can be used in one of two ways. Firstly the RAM can indeed be used as four sideways RAM areas, in sideways blocks, &C, &D, 0, and 1, with sideways ROM images loaded in from disc to be used in the normal way. For this purpose the DFS has a few extra utilities. *SRLOAD and *SRSAVE will load and save ROM images to and from disc. *SRREAD and *SRWRITE will copy sideways RAM to RAM and vice versa. However, *SRSAVE and *SRREAD will not operate with sideways ROM as a protection against piracy.

The extra 64K can also be used as a contiguous block of memory for data



storage. For this purpose there are two more commands - *SRDATA and *SRROM to specify each of the 16K blocks as general RAM or for sideways software. However, using it in this way is pretty tricky.

Much easier is to use Basic. Supplied with the upgrade is a disc containing a new BBC Basic called BAS128. This loads into the area of memory normally occupied by your programs. Your programs instead go in the 64K extra RAM. Having a full 64K free for Basic programs makes the B+128K unique amongst the 128K, 8 bit home computers. BAS128 is identical to normal ROM Basic with a couple of exceptions. All RAM addresses range from &10000 to &1FFFF and DIM and the indirection operators work accordingly. Using the (normal) range of &0000 to &FFFF causes problems if you start to poke into regions of the RAM containing BAS128. The assembler operates with these 17 bit addresses too, but you cannot assemble any code greater than 16K in length as BAS128 cannot cope with a jump across the 16K RAM blocks.

Although the B+128K is so obviously an afterthought, it is put together extremely well. The four sideways RAM slots are nothing very new or exciting, but 64K free for Basic is useful and should keep the BBC micro in the public's eye and on their shopping lists for a bit longer.

Programming with

Wordwise

(Part 3)

Stephen Ibbs concludes his series on programming in Wordwise Plus with more useful routines and hints to make the most of this popular wordprocessor.

Plus Plus

The problem set last month concerned the conversion of the chapter numbering routines from BEEBUG Vol.4 No.2 into a single routine with procedures that can be selected from a menu. There will need to be three sections to such a program:

The design of the menu.

The ability to reset all variables.

The actual numbering routines.

The menu can be made more 'user-friendly' by the use of VDU commands (corresponding to their Basic counterparts) as described in the 'Hints for Wordwise Plus Users' in BEEBUG Vol.4 No.5. Here is the menu section of the program:

```
SELECT TEXT
CLS
PRINT
DOTHIS
VDU131,141
PRINT "      Chapter numbering"
TIMES 2
VDU31,0,6
PRINT " Press 1 for a new chapter"
PRINT " Press 2 for a new section"
PRINT " Press 3 for a new sub-section"
PRINT
PRINT
VDU134
PRINT "Press Return to reset numbers"
D%=GET
```

VDU141 gives the double height lettering of the title and VDU131 the yellow letters. Note we use the DOTHIS-TIMES commands to print the two halves of the double height title. The VDU31 command, followed by 2 parameters,

places the cursor anywhere on the screen.

The ability to display a menu from which selections may be made, makes the operation of routines very easy, but be careful not to use up precious memory space just on 'decorating' a routine.

GET assigns the ASCII code for the key pressed to D%. We need to check if Return has been pressed, reset the variables if it has, then return to the text by jumping to the end of the routine. However, if 1, 2, or 3 have been pressed then the appropriate procedure must be called. So the following lines are added:

```
IF D%=13 THEN PROC RESET
IF D%=13 THEN GOTO end
PRINT
VDU134
PRINT "Type the title then press RETURN"
VDU31,1,16
IF D%=49 THEN PROC CHAP
IF D%=50 THEN PROC SECT
IF D%=51 THEN PROC SUB
.end
A$=""
DISPLAY
END
```

After the end has been marked we can insert the procedures as follows.

```
.CHAP
A%=A%+1
B%=0
C%=0
A$=GLK$
TYPE STR$A%+" "+A$+"|R"
ENDPROC
.SECT
B%=B%+1
C%=0
A$=GLK$
TYPE STR$A%+"."+STR$B%+" "+A$+"|R"
ENDPROC
.SUB
C%=C%+1
A$=GLK$
TYPE STR$A%+"."+STR$B%+"."+STR$C%+" ";
TYPE A$+"|R"
ENDPROC
.RESET
A%=0
B%=0
C%=0
A$=""
ENDPROC
```

You will notice that the TYPE lines have been simplified by using the STR\$ function. This is much easier than having to convert each integer variable into its string equivalent. However, note that in the SUB procedure the TYPE command is split between two lines. No segment program line should be longer than 40 characters. Otherwise, under rare circumstances, the routine may not run. Note also that only A\$ is now used so it is easily cleared at the end.

HINT: No segment line should be longer than one screen line. Use abbreviations and omit spaces to reduce the line length.

Note that inserting words into your text, using TYPE, will result in the word count, displayed at the top of the screen, being incorrect. This can be corrected using the command RECOUNT, either from within your program or in immediate mode.

HINT: When you have used the TYPE command to insert text into the main area, you will need to RECOUNT to maintain an accurate word count.

TEXT PRINTING

If you tried the PRINT routines published last month you may have found that it would be useful to know if your text files finish with a BP embedded command as this would spoil the continuity of the printout. It is quite easy using the knowledge gained so far and the TRUE/FALSE functions to check if a BP exists and to deal with it.

One problem is that the embedded command could be in upper or lower case. This problem occurs time and time again. Rather than test for both BP and bp (or even Bp and bP), we can AND the ASCII code with 223 to always obtain the upper case code.

HINT: When a letter is to be entered, trap lower case by using GET AND 223.

Thus to check for a BP, the lines:

```
SELECT TEXT
A%=FALSE
CURSOR BOTTOM
CURSOR LEFT 5
FIND CHR$2
IF EOT THEN GOTO fail
```

start the routine by setting A% to return a FALSE value (zero), move back a few characters from the end of the text and look for an fl (found with CHR\$2). There is a trap for EOT which will eventually jump to the end of our routine. Once we have found the fl, the cursor must move to the right, then check the next 2 characters:

```
CURSOR RIGHT
IF (ASC(GCT$) AND 223)<>66 THEN G.fail
IF (ASC(GCT$) AND 223)=80 THEN A%=TRUE
.fail
IF A% THEN CURSOR LEFT 3
DISPLAY
```

66 and 80 are the ASCII codes for B and P respectively. Only if both are found will A% become TRUE and the cursor be moved back to the start of the embedded command. Note that the penultimate line does not have to be: IF A%=TRUE THEN... because IF A% THEN... means the same.

This 'subroutine' would normally be part of a larger program, for example a continuous files printer, where you may need to check how each file finishes. Once you have A% set as TRUE you can then DELETE or TYPE the BP command as needed. Deleting would be very easy, by inserting the following just before the DISPLAY line, or making it a procedure to be called by, say, 'IF A% THEN PROC delete'.

```
IF NOT A% THEN GOTO skip
REPEAT
DELETE AT
UNTIL EOT
.skip
```

USING TEXT FILES

As well as using text in memory, segment routines can be written to access Wordwise text files on disc (and, less usefully, on tape) or create new ones. There are two important points to remember when accessing files in this way:

- 1) Check often to see if you are at the end of the file, using EOF#.
- 2) Remember to close the file when you have finished.

Initially, to open a file, you have to assign its 'channel number' to an integer variable. Thus to open a file called TESTER, we would use:

```
A%=OPENIN "TESTER"
```

Once opened, any reference to this file is done via A%. As a very simple example, let us take the first five lines from a file and insert them into segment 6. The routine could consist of the following:

```
A%=OPENIN "TESTER"
X%=0
REPEAT
A%=GLF$#A%
SELECT SEGMENT 6
TYPE A$
IF LEN(A$)<255 THEN TYPE "|R"
X%=X%+1
UNTIL X%=5
CLOSE#A%
A$=""
DISPLAY
```

The GLF\$# ('Get Line from File') behaves in a similar way to GLT\$ in that it can accept up to 255 characters, but will stop before this figure is reached if it finds a Return. When collecting single characters, you can either use GCF\$# which returns the character, or BGET# which returns the ASCII equivalent.

As was mentioned in the first part of this series, a "|R" will have to be included in the TYPE lines, otherwise the recovered text will have no Returns in it. If you are using discs, you will see how fast the routine runs. The OPENOUT function can be used when writing files. To develop the above example further we could take the first 100 characters from file TESTER and insert them into a second file TESTING with the following:

```
A%=OPENIN "TESTER"
B%=OPENOUT "TESTING"
X%=0
REPEAT
C%=BGET#A%
BPUT#B%,C%
X%=X%+1
UNTIL X%=100
CLOSE#A%
CLOSE#B%
DISPLAY
```

However, note that if a file called TESTING already exists on the disc, this will be deleted, so check before running any routine like this. Add some lines to the start of the routine:

```
B%=0
B%=OPENIN"TESTING"
IF B%<>0 THEN CLOSE#B%
IF B%<>0 THEN GOTO warn
```

A routine, 'warn', should be inserted somewhere in the program to warn that a file already exists with that name. B% will only be assigned a channel number, and be greater than 0, if a file called TESTING is found. BGET# is used because the writing to a file can only be performed one character at a time, using BPUT#.

You can search for a particular character or line in a file using the GCF\$# and GLF\$# functions, but you must remember to move backwards after finding it so that the pointer is at the correct place, using the PTR# function. For example if we were looking for the line "This is a test line" followed by a RETURN you would then write:

```
A$="This is a test line"
A%=OPENIN "TESTING"
REPEAT
B%=GLF$#A%
UNTIL B$=A$ OR EOF#A%
PTR#A%=PTR#A%-(LEN(B$)+1)
DISPLAY
```

The same routine could be used if you were looking for one particular character by using GCF\$# instead, in which case you would need only to move the pointer back one position. In case the line doesn't exist the REPEAT-UNTIL condition include a test for the end of the file. Also, if the line being searched for is exactly 255 characters long, a Return won't be encountered so only move the pointer back by the length of B\$.

HINT: When you've used GCF\$# or GLF\$# to find a particular point in the file remember you may need to move the pointer back either 1 position with GCF\$, or the length of GLF\$# (plus 1 if the length is less than 255).

This short series is intended to provide an introduction to programming in Wordwise Plus. Much more could be written about the commands we have covered and, indeed, those that we have not touched. Readers are strongly encouraged to delve further into this language and see just how flexible and useful it can be.

Earlier BEEBUG Workshops on sorting techniques provoked considerable interest. This time Surac considers how best to choose a sort routine to match your requirements. The fastest sort in theory is not always the fastest in practice.

Regular readers may remember that, in Beebug Vol.3 No.10 & Vol.4 No.1, I looked at various ways of sorting data into order. Those articles prompted quite a lot of correspondence and so, this month, I'll follow up some of the points which you raised.

One result of the fixed length of BEEBUG Workshops is that sometimes, less information is included than one might wish. One such area in those articles was how to add an item into the right place in an already-sorted list. For instance, if you are keeping a list of club members, you will probably want to add new ones in alphabetical order.

One way is to re-sort the list as each item is added. To do this, though, you must choose the sorting algorithm very carefully so that the sort after each item is entered is very fast.

Here's where it comes in useful to know how the different sorts react to different data arrangements. Some are very much affected by the way that the data is already ordered, while others have pretty consistent run times. For example, the Shell sort is a good general-purpose sort, and normally much faster than the Bubble sort. In one particular case, however, the Bubble is noticeably faster. If the list is

already in order, apart from only one item, then the correction can be made very quickly. This is because the sort only needs to make one pass through the list.

This is exactly the case we have when we add an item to a list. However, we do need a slightly more specialised program. My original Bubble sort started at the bottom (i.e. the low index numbers) of the array of data and bubbled items up to the top. If, however, we are adding data to the list, it is easier to add it at the top (e.g. by adding 1 to an array index); we therefore need to adapt the system to bubble (or should it be sink?) an item DOWN to the correct place.

Here is a revised version of the original PROCBUBBLE to do the job. You will see that I have also changed it to access the data in array() via the pointers in ptr%() - this is the most likely need in database programs, etc.

```

SINKSORT
10000 DEF PROCsink(st%,fin%)
10010 IF st%>=fin% THEN ENDPROC
10020 LOCAL F%,I%
10030 FOR I%=st% TO fin%
10040 ptr%(I%)=I%
10050 NEXT
10060 REPEAT
10070 F%=FALSE
10080 FOR I%=fin% TO st%+1 STEP -1
10090 IF array(ptr%(I%-1))>
array(ptr%(I%))
THEN PROCswap
10100 NEXT
10110 st%=st%+1
10120 UNTIL NOT F%
10130 ENDPROC

10500 DEF PROCswap
10510 LOCAL temp%
10520 temp%=ptr%(I%-1)
10530 ptr%(I%-1)=ptr%(I%)
10540 ptr%(I%)=temp%
10550 F%=TRUE
10560 ENDPROC
    
```

QUICKSORT

Several Beebug members referred to the 'Quicksort'. This is a very fast way of

```

QUICKSORT
11000 DEF PROCquicksort(st%,fin%)
11010 IF st%>fin% THEN ENDPROC
11020 LOCAL hiptr%,loptr%,midval
11030 loptr%=st%:hiptr%=fin%
11040 midval=array((st%+fin%)/2)
11050 REPEAT
11060   loptr%=loptr%-1
11070   REPEAT
11080     loptr%=loptr%+1
11090     UNTIL array(loptr%)>midval
11100     hiptr%=hiptr%+1
11110     REPEAT
11120       hiptr%=hiptr%-1
11130       UNTIL array(hiptr%)<=midval
11140       IF loptr%<=hiptr% THEN
         PROCqswap(loptr%,hiptr%)
11150       UNTIL loptr%>hiptr%
11160       IF st%<hiptr% THEN
         PROCquicksort(st%,hiptr%)
11170       IF loptr%<fin% THEN
         PROCquicksort(loptr%,fin%)
11180     ENDPROC

12000 DEF PROCqswap(sptr1%,sptr2%)
12010 LOCAL temp
12020 temp=array(sptr1%)
12030 array(sptr1%)=array(sptr2%)
12040 array(sptr2%)=temp
12050 loptr%=loptr%+1
12060 hiptr%=hiptr%-1
12070 ENDPROC

```

sorting long lists, although it's not so good for short ones. Unfortunately, it is a little tricky to explain.

Briefly, the sort finds the value in the middle of the array and then works in from each end to the middle. As it works in, the sort moves the individual elements around so that all in one half are greater than or equal to the middle value, while those in the other half are less than it. At that stage, it has done a very rough ordering.

Having done that, it splits the whole array into 2 halves, re-sorting each half into a high side and a low side. Each of the 2 halves is then split into 2 more halves and the procedure repeated. Then the quarters are halved... At the end of repeated halving, which goes as far as adjacent elements, the whole thing is found to be sorted. The technique is sometimes called 'Sorting from Both Ends'. What I hope that description shows is that the sort uses the same basic split and

divide process on successively smaller parts of the array until it is finished. Such an approach is a perfect application for a recursive procedure.

All that recursion means is that a function is called by itself, which is called by itself..., doing a bit each time until a call creates an exit condition and the whole chain unravels. Here's some pseudo-code for an important recursive procedure:

```

1 DEF PROCdrink_Beer
2 Take a Swig
3 IF Glass NOT Empty PROCdrink_Beer
4 ENDPROC

```

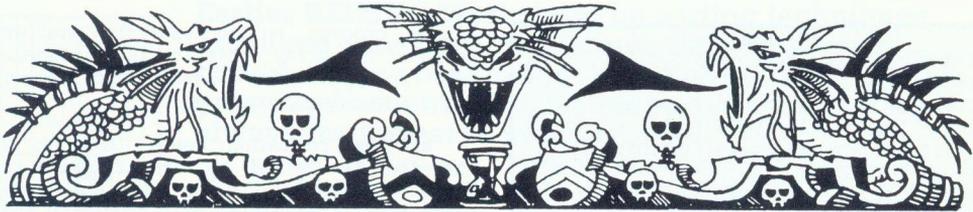
Most Basics make it almost impossible to use recursion, since all the variables used at each recursion level must be preserved while deeper calls are active. BBC Basic, though, supports recursion very well, with its FNs, PROCs and LOCAL variables. Here is a recursive Quicksort.

Line 11040 finds the middle value of array(), or the part of it sent to the procedure, and then the two REPEAT-UNTIL loops at lines 11060-11090 and 11100-11130 find misplaced elements either side of the centre point. Assuming that the 2 pointers have not crossed (why?) the elements are swapped.

This continues until the pointers HAVE crossed when, and only if there are elements to sort, the recursive calls to further sort the high and low halves are made. These, of course, repeat the process until only one element in array() has to be 'sorted', when that recursive chain unravels. Note that the procedure's parameters are LOCAL, as are the variables it uses. This means, for example, that loptr% in one level of the chain is different from the loptr% in the next level up or down - the system saves them all for us.

On the magazine cassette/disc, you will find a Quicksort demo program which shows very clearly how the method sorts in from each end. It also sorts the same items via a Shell sort, so you can see the speed difference of the two approaches.

Finally, my thanks to all those Beebug members who contributed ideas to this month's column, in particular Mr G.L.Howitt and Mr A.G.Rimmer.



by Mitch

ENTURE GAMES ADVENTURE GAMES ADVENT

Wizards who have been cursed with the affliction of a Spectrum in the past may have come across this month's first adventure before.

Title : Castle Blackstar
Supplier : CDS Micro Systems
 Silver House, Silver St.,
 Doncaster, South Yorks.
Price : £6.95 (cass. only)

The original version of this text-only game was voted one of the top eight adventures of 1984 - and it deserved it. The game conceals over 200 well described locations about its person, crammed full of reasonable puzzles. The object of the quest is to search the rooms and underground passages of Castle Blackstar and thus find and restore the magic orb to the goddess Artemis.

Luckily the goddess keeps a weather eye on your progress and should you go 'A over T' down some black hole, she will reincarnate you with a quick puff of pixie dust. The game permits easy access to a large number of initial locations which contain a wealth of interesting objects to manipulate - a bottle of heat resisting liquid, a coil of elfin rope and a bag of flour, for example.

There are many cryptic messages to read, odd shaped buttons to push and wheels to turn, all of which have intriguing results off-screen. As with many Middle Earth scenarios there is a forest maze, in which I soon felt as dense as the trees so I returned as quickly as I could to the relative sanity of the castle.

The game accepts sentence commands but should it fail to recognise something, it unfortunately gives no indication as to

which word it balks at - budding authors please note! No hints and answers sheet is supplied, so befuddled players must write to CDS for any help. I must confess that I prefer this option. Supplying a sealed envelope to an adventurer is a bit like placing a bag of buns within talon reach of the BEEBUG dragon.

There is a Troll Bridge and a knife-throwing elf but there is little else in common with 'Colossal' other than the sheer size of this game. Furthermore, even more complex adventures are promised by this company. That's certainly good news.

On the subject of Answer Sheets, I must confess to having finally peeked at the solutions to Acornsoft's Acheton - I was not impressed! Personally I feel some of the solutions are less than fair which is a great pity as I had high hopes for the game - any comment?

For those intrepid adventurers who possess a fairly full purse of gold, a lust for adventure beyond the stars, (and a disc drive) comes Enthar Seven.

Title : Enthar Seven
Supplier : Robico Software
 3 Fairland Close,
 Llantrisant, Mid Glamorgan
Price : £17.95 (two 40 track discs)
 £16.95 (80 track disc)

Fresh from its triumphs with 'Assassin' and 'Isle of Xaan' (see the reviews in BEEBUG Vol.4 No.4), Robico has unveiled the answer to Acornsoft's 'Acheton'. With over 450 locations to explore, 80 objects to manipulate and 1200 messages, Enthar is a whole new dimension for Beeb disc owners. Happily, the problems contained within, are not as



ADVENTURE GAMES ADVENTURE GAMES AD

large as the game itself and so it was more the eventual dawning of the new day which finally made me release the keyboard and stagger back up from the dungeon to bed.

It is the distant future. You are on board a fragile 'Interplanetary Space Hopper' which orbits a small Earth-like planet known as Enthar Seven. Your task is to reach the planet's surface before the Hopper's orbit decays and the spaceship spirals into oblivion! Once on Enthar Seven you must explore the planet and find and board an escape vessel.

The Enthar Seven planet is a long abandoned Earth colony which is now kept ticking over by a small crew of robots who continue to clean and guard the various electronic whizz boxes. Their other main task is to blast the living daylights out of foolish adventurers who are to be found wandering around looking for such things as computer registration cards and sticking plasters!

My spell book defines a good adventure as one which allows me to wander throughout large sections of the terrain before finally having to face the nitty-gritty. I also prefer my nitty to be mildly gritty rather than a few concrete blocks as in Acheton! Enthar has achieved this recipe rather well, in that the central feature of the game is the transporter room. Using the transporter dial you may select, at random, any of the seven main areas on the surface of the planet Enthar. This ensures that you won't sit crying over an expensive 'dragon in a poke' having been stumped by the first mind boggling dead end.

The locations are generally given full screen descriptions which give fairly obvious nudges when you are paying

attention. The main areas feature forests, laboratories, space shuttle interiors, and various computer control centres. The obligatory dimming torch and maze is to be found but their solution is designed to slow you down rather than to cause apoplexy.

The puzzles posed by the game are intriguing and the solutions are moderately easy. In the main I knew a probable solution to each puzzle and a likely object to overcome it. However, the difficulty came when attempting to find the object! The typical command examples given with the game are an invaluable source of help when phrasing your inputs, thus avoiding the usual frustrating search for the winning phrase required to satisfy a stubborn command analyser.

The game is text only, but this is always plentiful, witty, and in colour. As the game is supplied on disc (as 200K of data), there is some spare room so I'm surprised that Robico did not guild the lily with a few pretty pictures.

The game's high price tag is unfortunate as it comes at a time when price decreases seem to be the norm. However, this can be partially offset by joining the Robico Club which entitles members to a discount of approx £1.50. In addition to the game on the program disc, there is a section of commercials. Screen shots from other adventures and graphic games are shown along with reviewers' comments. This seems a painless way to view before you buy.

The true measure of this game is that I know I will continue to stumble through its corridors until I finish it, which is more than can be said for many other offerings I have peered into of late.



Programming sideways



Basic Programs

J. P. Jakubovics extends last month's sideways software utility to download your Basic programs from sideways ROM and RAM.

As mentioned in last month's article, only machine code routines can be executed in ROM. Basic programs can however be stored in ROM, provided they are moved down into main memory before being run. The move down can be done with a very simple machine code routine as in Program 3. To use it, proceed as follows:

1. Load into memory the Basic program you want to store in ROM, and make sure that it works correctly. While it is in the computer, find the length of your Basic program by typing:

```
L%=TOP-PAGE
PRINT L%
```

Make a note of the result, because if you need to switch off the computer before you have finished all the steps below, you will need to reset L% to this value.

2. The Basic program to be transferred into ROM should be saved to disc or tape.
3. Type or load in Program 1 published in last month's article and add Program 3 to it (this can be done by merging the two programs together as explained on page 402 of the User Guide). Save the combined program, set PAGE to &4000 (PAGE=&4000 <Return>) and load the program back in.
4. Now run the program and note the value of 'basic%' which is printed out.

5. *LOAD the Basic program to the address given by 'basic%'. For example, if basic% was printed as &2287 then type '*LOAD name 2287'.
6. *SAVE the entire machine code from address &2000 up to &4000 i.e. type *SAVE romname 2000 4000.
7. To test the program, you must either *LOAD it in at &8000 if you have sideways RAM, or blow it into an EPROM. Unfortunately, Program 2 from last month does not cope with Basic, because the Basic program to be copied from one part of memory to another would overwrite Program 2 before it had finished running.

The two lines 2620 and 2710 use the letter B to access the Basic program in ROM. This letter can be changed to any other capital letter from A to Z, but the two occurrences have to be the same.

When you have completed the above steps and wish to test that it all works, type *B (or whichever letter you have chosen), and your Basic program will be moved down into main RAM (starting at PAGE) and run.

Program 3

```
2620 .help(ASC("B")-65) OPT FNegus(CHR
$13+CHR$13+"Runs the BASIC program in RO
M.")
2630 BRK
2640 .basic%
2650 ]
2660 FOR J%=1 TO L%
2670 P%?J%=0
2680 NEXT
2690 P%=P%+L%
2700 [OPT I%
2710 .com%(ASC("B")-65) LDX # (L% DIV 25
6)
2720 TXA
2730 CLC
2740 ADC &18
2750 STA &6F
2760 LDA #0
2770 STA &6E
2780 LDA #((basic%+os%+L%- (L% MOD 256))
DIV 256)
2790 STA &6D
2800 LDA #((basic%+os%+L%- (L% MOD 256))
MOD 256)
2810 STA &6C
2820 LDY # (L% MOD 256)
```

```

2830 .downloop% DEY
2840 LDA (&6C),Y
2850 STA (&6E),Y
2860 CPY #0
2870 BNE downloop%
2880 DEC &6F
2890 DEC &6D
2900 DEX
2910 BPL downloop%
2920 BMI fx15%
2930 .run% OPT FNegus("O."+CHR$13+"RUN"
+CHR$13)
2940 BRK
2950 .fx15% LDA #15
2960 LDX #1
2970 JSR &FFF4
2980 LDX #0
2990 STX &6B
3000 LDA #138
3010 .fx138% LDX &6B
3020 LDY run%+os%,X
3030 BEQ endbasic%
3040 LDX #0
3050 JSR &FFF4
3060 INC &6B
3070 BNE fx138%
3080 .endbasic% RTS
9815 PRINT"basic%&";~basic%

```

As an example of transferring a Basic program to sideways ROM or RAM, you can use the program listed below. This is a simple Basic program to print out the titles of your sideways ROMs in mode 7.

```

10 MODE 7
20 FORA%=1 TO 23
30 VDU31,0,A%,146,234
40 IF A%>1 AND A%<23 VDU131
50 VDU31,38,A%,146,181
60 NEXT
70 FORA%=2 TO 38
80 VDU31,A%,1,252
90 VDU31,A%,23,175
100 NEXT
110 VDU28,3,21,36,3,12
120 PRINTTAB(10)"ROM SOFTWARE"
130 PRINTTAB(9)CHR$145;STRINGS(12,CHR$
163)
140 PRINT:FORB%=0TO14
150 VDU133:IFB%<10 VDU32
160 PRINT;B%;".":VDU134

```

```

170 N%=&8009:REPEAT
180 Y%=B%:?&F6=N%:?&F7=N% DIV 256
190 A%=USR(&FFB9)AND&FF0000 DIV &10000
200 IFA%>0 AND A%<32 A%=32
210 VDU4%:N%=N%+1:UNTILA%=0 OR N%>&802
4
220 PRINT:NEXT

```

Type the above example in and save it to cassette or disc. Then find the length of the program by typing L%=TOP-PAGE and PRINT L%. The value of L% should be around 414.



Now combine Program 1 from last month and Program 3 from this month to form a new program. Save the new program and set PAGE to &4000. Re-load the program and run it.

This should now set up the machine code move-down routine and print out a value for basic%. The value of basic% should be around &22B6. Then *LOAD the Basic program to this address (*LOAD test 22B6) and finally *SAVE the whole code from &2000 to &4000.

After completing the above, you can then *LOAD the code into sideways RAM (using *LOAD code 8000) or program the code into an EPROM. If you do use sideways RAM remember to press Break after loading the code and before typing *B to access the ROM.

HINTS HINTS HINTS HINTS HINTS HINTS HINTS HINTS

CLEAR DESCRIPTION - John Thomas

The Basic command, CLEAR, is described in the User Guide as resetting all variables except A%-Z% and @%, to zero. However, it also clears the stack and so cannot be used within a loop or in a subroutine or procedure.

BOOK REVIEWS

Wordwise Plus by Bruce Smith published by Collins at £9.95
Reviewer: Mike Williams

This book contains 230 pages covering both Wordwise and Wordwise Plus. Newcomers to both wordprocessing and Wordwise should find this book invaluable from the start. It contains, for example, about the clearest instructions I've seen for fitting a ROM into your machine.

The early chapters introduce, with many examples, all the main features of Wordwise as a wordprocessor. There are also frequent illustrations of the Beeb's keyboard to highlight the use of a particular key or group of keys. Some may feel that this is overdone - who needs a third of a page diagram just to identify the Tab or Break keys? Newcomers, though, will find this attention to detail reassuring.

The second part of the book is largely devoted to the use of Wordwise Plus as a programming language. Most of the features of writing Wordwise Plus segment programs are clearly described, and this is all put to good use in part three which contains 16 complete segment programs. BEEBUG members may well feel that this whole subject has already been quite comprehensively covered in the magazine.

To sum up, this is a very well written and illustrated book and should sell well particularly to those with little or no experience of Wordwise. More experienced

users of both Wordwise and the Beeb may feel that for them too much space has been wasted and that the price of £9.95 is high for what the book has to offer.

Within the BBC Microcomputer by Roger Cullis published by Losco Ltd at £11.95.
Reviewer: Ian Tresman

'Within The BBC Microcomputer' holds pages of numbers and addresses, and a treasure of information for the curious trying to find what makes the BBC micro tick. This book describes all the major software systems by referencing the main subroutine entry point addresses. And unlike recent Basic reference books that have appeared, Roger Cullis not only goes into more detail, but covers a wider range of software too. You will find comprehensive descriptions of these systems: OS 1.2, Basic I, Basic II, HiBasic, DFS 0.9, NFS 3.34, 6502 Second Processor OS, Tube Communications Routines.

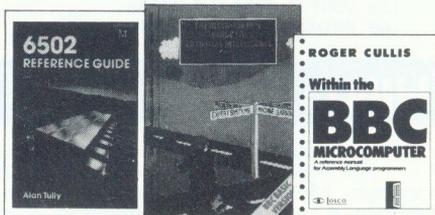
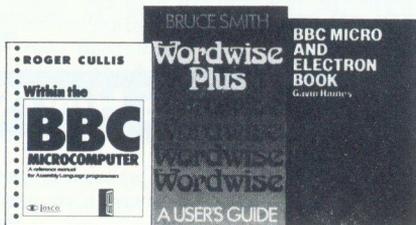
The layout of each system description is similar. Following a general memory map, zero page usage, and summary, the gazetteer details four to eight hundred routine entry points. For example, the OS 1.2 sections shows you the start of every *FX and OSBYTE call, or even where the cassette filing system generates a 0.2 second interblock gap.

Although this book is printed on unusually thick, rough paper held together in the standard spiral bound format, do not judge it by its proverbial cover. Buy it!

6502 Reference Guide by Alan Tully, published by Melbourne House at £9.95.
Reviewer: Mike Williams

As the title suggests, this is essentially a reference guide to programming the 6502 CPU. The book starts with the usual chapters describing the

6502 REFERENCE GUIDE Alan Tully



registers and the different addressing modes, but the bulk of the book is devoted to a comprehensive description of the complete 6502 instruction set. This also includes some useful information on timing. I would have liked longer examples showing the use of each instruction in a relevant context, rather than in isolation, and diagrams showing how the bits are shuffled around between and within the registers.

Personally, I still prefer my older '6502 Assembly Language Programming' by Leventhal (published by McGraw-Hill), a larger, more comprehensive and slightly more expensive book. The 6502 Reference Guide by Alan Tully is a good (and cheaper) runner-up.

BBC Micro and Electron Book by Gavin Haines, published by McGraw-Hill at £11.95.

Reviewer: Ian Tresman

If the title sounds like just another run-of-the-mill computer book then read on. Gavin Haines presents a very readable introduction to the BBC Micro but does not stop there, and he does not make the mistake of resorting to computer jargon.

There are sections aimed at the absolute beginner such as 'Typing in programs', and 'How to write a Basic program'. Of course, there is the usual description of Basic keywords liberally sprinkled with example programs.

I suspect that the author's own experiences have caused to him to pen the more practical sections in his book. 'After-sales queries', 'Buying equipment', 'Types of disk interface', and 'Transferring your favourite cassettes to disk', are some of the areas covered.

Once these topics have earned the novice his stripes, the book continues with not only an 'Introduction to the

6502', and 'Machine code hints', but 'Writing advanced software', and 'Selling your software'.

Whether you are just starting out with your newly acquired BBC Micro or Electron, or really exploiting its potential, then this book is for you.

The Hitch-Hiker's Guide to Artificial Intelligence by Richard Forsyth and Chris Naylor, published by Chapman and Hall at £8.95.

Reviewer: Mike Williams

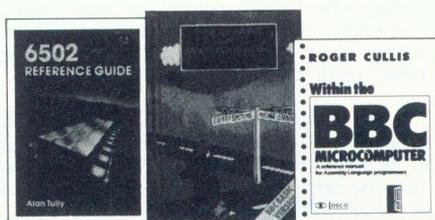
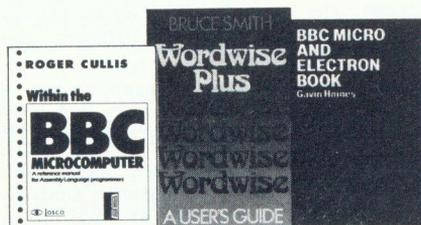
It is worth mentioning at the outset that this book is available in two versions, the one reviewed here being in BBC Basic (the other is in Applesoft Basic).

Artificial Intelligence is a subject that most computer enthusiasts have heard about, and maybe occasionally have read about in the popular press. Most would probably admit that in practice they were little more than novices in the field. This is where the authors of this book have stepped in with the aim of showing that artificial intelligence is exciting and well within the capabilities of most home users. How far the authors have succeeded you will have to judge for yourself.

Certainly this is not a book for the faint-hearted. There is a fair sprinkling of theory throughout the book but all is presented in a very readable way. Typical topics covered by this book include expert systems, natural language, image processing, machine learning, problem solving and game playing strategies.

If you are tiring of some of the more conventional applications of computers then this book offers much food for thought. It is well written, attractively produced and well worth looking at. 

6502 REFERENCE GUIDE Alan Tully



CHINESE CHEQUERS

With the plum pudding eaten and the Queen's Christmas message over for another year, what better than to relax with that traditional game of Chinese Chequers. Peter Clamp explains all.

I have a maiden aunt - a retired mathematics teacher - for whom Christmas is not Christmas unless she can play several games of Chinese Chequers following the Plum Pudding and the Queen's speech. Appealing though this game may be, even the most festive family gathering may lack the stamina for a third or fourth game, particularly after a heavy lunch. So, as part of Christmas preparations last year, I recast this board game in a form for two players, of which one is that most tolerant member of the family, the Beeb.

The rules of Chequers are straightforward. Each player has ten counters which may be moved only by that player and remain on the board throughout. Players take turns to move a counter, either sliding to an adjacent unoccupied position, or jumping over an adjacent counter (of either player) to an unoccupied position immediately beyond. Any number of such jumps may be chained together in a single turn. The winner is the first player to occupy the positions vacated by his opponent on the opposite side of the board. The human player must manoeuvre the white counters to the top of the board, while the computer attempts to occupy the lowest positions with the red counters. The six possible directions are numbered as indicated by the 'direction rose' next to the playing area.

To play well requires the exercise of considerable strategic skills. Frequently a player must make the choice between a jumping move which will take a counter, already advanced, into the home area, and

an undramatic shift of a lagging counter in danger of becoming stranded.

To run this program on a machine fitted with a DFS, you must use a move-down routine such as that in BEEBUGSOFT'S Toolkit or published in BEEBUG Vol.3 No.5.

PROGRAM NOTES

PROCturn is used to input the human player's move, and check its validity. It also (line 1310) randomly selects which player makes the first move.

PROCturn is more interesting as it embodies the computer's game strategy. Essentially, this is to assess all possible sliding and jumping moves for all ten counters, and to perform the move that gets the best assessment. Sliding moves are easily identified: there is a maximum of sixty, of which some will be off the board or blocked by other counters, and are therefore not assessed. Jumping moves are more difficult to explore systematically owing to the possibility of multiple jumps. The problem is solved by means of recursion.

The recursive procedure PROCsearch (X%,Y%,len%-1,d%) investigates the six potential jumps by a counter at position X%,Y%, eliminating impossible jumps and also any jump that would reverse the direction previously travelled (held in d%). For each of the possible jumps, the position XE%,YE% at the end of the jump is first assessed, and then investigated by a further call to PROCsearch. Every recursive procedure needs a 'let-out' condition to prevent it from calling itself ad infinitum. PROCsearch ceases to call itself when len% (the number of jumps in the move under consideration) reaches the value 6 given in line 2500. Without such a condition the computer might happily explore a fruitless triangle of three jumps until it exhausted its memory. The limit of six is arbitrary. You are welcome to change it, though a lower limit may cramp the computer's playing style, while a higher limit may exhaust the patience of its opponent!

To while away the time during all this recursion, the computer doodles some ceremonial dragons - one per counter - in the vacant area of the screen. These, and the (deeply researched) pentatonic fanfare that follows a winning move, are the only

ornamental concessions to the overall serious character of this game. In fact, the length and colouring of the dragon gives an indication of the number and length of jumping moves available to the associated counter (see lines 2460-2480).

The assessment of allowable moves is made in the procedure PROCassess. The scoring of each move is based on a number of factors, including the starting row of the counter, the number of rows advanced by the move, the nearness of the final position to the centre line of the board, and the number of jumps in the move (assigned to zero for a slide). The highest priority would be given to a move which advances a straggling counter through the maximum distance to a position near the centre line (where it is less likely to become stranded). Since the same position may sometimes be reached by unnecessarily devious jumping routes (which rather spoils the effect when such routes are really necessary), shorter routes are favoured when all other factors are equal. Just in case the computer's moves should become too predictable, a small random factor is added.

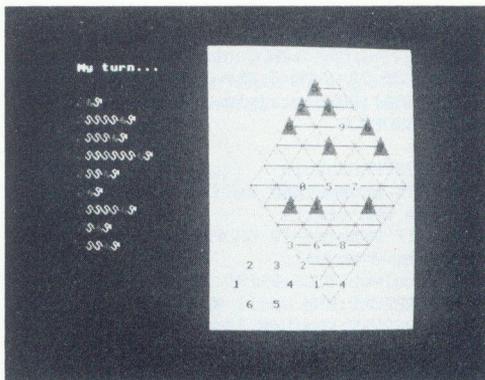
To complicate matters further, a slightly modified assessment is applied when more than seven counters have reached the home area. At this stage in the game, it is perhaps more important to approach a vacant home position than to advance for its own sake, and a factor of nearness to such a position is included in line 2580.

A historical footnote: compared with the board game Go, which is genuinely over 3000 years old, and originated in China under the name Wei-ch'i, Chinese Chequers is in reality a modern game derived from Halma, invented in England at the end of the last century. It is, in other words, about as Chinese as the BBC Micro.

```

10 REM PROGRAM CHEQUERS
20 REM VERION B0.1
30 REM AUTHOR P.CLAMP
40 REM BEEBUG DECEMBER 1985
50 REM PROGRAM SUBJECT TO COPYRIGHT
60 :
100 MODE7:PROCinstr:MODE1
110 PROCinit
120 REPEAT
130 PROCstartgame
140 REPEAT

```



```

150 PROCturn
160 PROCcturn
170 UNTIL FNgameover
180 PROCreport
190 UNTIL FNhadenough
200 MODE7:*FX12,0
210 END
220 :
1000 DEFPROCinit
1010 *FX11,0
1020 dum=RND(-TIME)
1030 DIM b%(6,6),Xinc%(6),Yinc%(6),this
mv%(7),bestmv%(7)
1040 DATA -1,1,0,1,1,0,1,-1,0,-1,-1,0
1050 FOR D%=1 TO 6:READ Xinc%(D%):READ
Yinc%(D%):NEXT
1060 ENVELOPE 1,1,48,-48,0,4,4,12,127,1
,0,-7,126,110
1070 VDU24,500;0;1279;1023;:GCOL0,130:C
LG
1080 VDU 28,0,29,14,2
1090 blk=0:red=1:ylw=2:wht=3
1100 VDU23,240,&18,&24,&22,&12,&09,&E8,
&C8,&B0
1110 VDU23,241,&18,&24,&22,&12,&09,&88,
&48,&30
1120 VDU23,242,&10,&18,&3C,&3A,&7E,&37,
&1F,&06
1130 VDU23,243,&18,&25,&23,&17,&09,&88,
&48,&30
1140 ENDPROC
1150 :
1160 DEFPROCstartgame
1170 hmvs=0:cmvs=0
1180 REM Draw board, reset b%
1190 nh=0:nc=10
1200 FOR X%=0 TO 6:FOR Y%=0 TO 6
1210 PROCgrid:b%(X%,Y%)=-1
1220 IF X%+Y%<4 b%(X%,Y%)=nh:PROCcntr(w
ht,nh):nh=nh+1
1230 IF X%+Y%>8 b%(X%,Y%)=nc:PROCcntr(r
ed,nc):nc=nc+1

```

```

1240 NEXT:NEXT
1250 REM Draw direction rose
1260 X%=-2:Y%=3:PROCcntr(wht,-1)
1270 FOR D%=1 TO 6:X%=-2+Xinc%(D%):Y%=3
+Yinc%(D%):PROCcntr(ylw,D%):NEXT
1280 ENDPROC
1290 :
1300 DEFPROCturn
1310 IF cmvs=0 AND RND(2)=2 ENDPROC:REM
Decide who starts
1320 IF FNhhome=10 ENDPROC
1330 hmvs=hmvs+1
1340 col=wht:CLS:PRINT "Your turn..."
1350 REPEAT:REM Until OK'd
1360 REPEAT:REM Until valid
1370 PRINT "Which counter""to move?""
(0-9)":cr=EVAL(FNget("0123456789"))
1380 PROCfind:PROCflash
1390 PRINT "Slide or Jump?""(S/J/R)":M
S=FNget("SJR")
1400 IF M$="S" PROCslide
1410 IF M$="J" PROCjump
1420 IF M$="R" valid=FALSE:PROCreplace
1430 UNTIL valid
1440 PRINT"Move OK?""(Y/N)":a$=FNget("
YN")
1450 IF a$="N" PROCreplace
1460 UNTIL a$="Y"
1470 IF FNhhome=10 AND FNchome<10 PROCw
in("YOU")
1480 ENDPROC
1490 :
1500 DEFPROCcturn
1510 IF FNchome=10 ENDPROC
1520 cmvs=cmvs+1
1530 col=red:CLS:PRINT "My turn..."
1540 bestasst%=-1000:ch%=FNchome
1550 FOR cr=10 TO 19
1560 PROCfind
1570 REM Try all poss slides
1580 thismv%(0)=-cr:len%=0
1590 FOR D%=1 TO 6
1600 IF FNcanslide thismv%(1)=D%:thismv
%(2)=0:PROCassess
1610 NEXT
1620 REM Try all poss jumps
1630 thismv%(0)=cr
1640 PROCsearch(XB%,YB%,1,-3)
1650 NEXT cr
1660 PROCcmove
1670 IF FNchome=10 AND FNhhome<10 PROCw
in("I")
1680 ENDPROC
1690 :
1700 DEFNFgameover =FNhhome=10 AND FNch
ome=10
1710 :
1720 DEFPROCreport
1730 CLS
1740 PRINT""Total moves""this game"

```

```

1750 PRINT"You... ";hmvs
1760 PRINT"Me... ";cmvs'
1770 IF cmvs=hmvs PRINT"Actually...""i
t's a draw!":ENDPROC
1780 IF cmvs>hmvs PRINT"You won by"
1790 IF cmvs<hmvs PRINT"I won by"
1800 diff%=ABS(cmvs-hmvs)
1810 IF diff%=1 PRINT"just one move" EL
SE PRINT;diff%;" moves"
1820 ENDPROC
1830 :
1840 DEFNFhadenough
1850 PRINT"Another game?""(Y/N) "
1860 =FNget("YN")="N"
1870 :
1880 DEFPROCcntr(col,cr)
1890 MOVE FNxs=30,FNys=30:PLOT 0,60,0:G
COL0,col:PLOT 81,-30,70
1900 IF cr>-1 VDU5:PLOT 0,-15,-30:GCOL
0,blk:PRINT;crMOD10:VDU4
1910 ENDPROC
1920 :
1930 DEFPROCgrid
1940 PROCcntr(ylw,-1):GCOL0,blk
1950 FOR D%=1 TO 6
1960 IF NOT FNoffboard MOVE FNxs,FNys:P
LOT 5,(FNxs+Xinc%(D%)*25-Yinc%(D%)*25),(
FNys+Xinc%(D%)*35+Yinc%(D%)*35)
1970 NEXT
1980 ENDPROC
1990 :
2000 DEFPROCfind
2010 X%=-1:REPEAT X%=X%+1
2020 Y%=-1:REPEAT Y%=Y%+1
2030 UNTIL b%(X%,Y%)=cr OR Y%=6
2040 UNTIL b%(X%,Y%)=cr
2050 XB%=X%:YB%=Y%
2060 ENDPROC
2070 :
2080 DEFPROCflash
2090 PROCcntr(col,cr):PROCblip:PROCcntr
(2,cr):PROCblip:PROCcntr(col,cr):PROCbli
p
2100 ENDPROC
2110 :
2120 DEFPROCblip
2130 SOUND 1,-10,101+4*cr,8:TIME=0:REPE
AT UNTIL TIME>40
2140 ENDPROC
2150 :
2160 DEFPROCslide
2170 PRINT"which""direction?""(1-6/R)
"
2180 a$=FNget("123456R")
2190 IF a$="R" valid=FALSE:PROCreplace:
ENDPROC
2200 D%=EVAL(a$):IF FNcanslide PROCstep
:valid=TRUE ELSE PROCsorry:valid=FALSE
2210 ENDPROC
2220 :

```

```

2230 DEFPROCjump
2240 PRINT "which""direction(s)?" ""(1-6
/F/R)"
2250 a$=FNget("123456FR")
2260 IF a$="F" valid=TRUE:ENDPROC
2270 IF a$="R" valid=FALSE:PROCreplace:
ENDPROC
2280 D%=EVAL(a$):IF FNcanjump PROCstep:
GOTO 2250 ELSE PROCsorry:valid=FALSE
2290 ENDPROC
2300 :
2310 DEFPROCstep
2320 b%(X%,Y%)=-1:XES%=XE%:YES%=YE%:PRO
Cgrid:X%=XES%:Y%=YES%:b%(X%,Y%)=cr:PROCf
lash
2330 ENDPROC
2340 :
2350 DEFPROCsorry
2360 PRINT "Sorry"" -impossible!":SOUN
D 0,-15,2,10:TIME=0:REPEAT UNTIL TIME>15
0:PROCreplace:*FX15,0
2370 ENDPROC
2380 :
2390 DEFPROCreplace
2400 IF NOT (XB%=X% AND YB%=Y%) XE%=XB%:
YE%=YB%:PROCstep
2410 CLS:PRINT "Start again..."
2420 ENDPROC
2430 :
2440 DEFPROCsearch(X%,Y%,len%,d%)
2450 LOCAL D%
2460 IF len%=1 VDU17,red,240:REM tail
2470 IF len%=2 VDU17,ylw,241:REM body
2480 IF len%=3 VDU17,wht,241:REM body
2490 FOR D%=1 TO 6
2500 IF ABS(D%-d%)<3 IF FNcanjump this
mv%(len%)=D%:thismv%(len%+1)=0:PROCases
s:IF len%<6 PROCsearch(XE%,YE%,len%+1,D%
)
2510 NEXT
2520 IF len%=1 VDU17,red,242,17,wht,243
,10,10,13:REM head
2530 ENDPROC
2540 :
2550 DEFPROCassess
2560 brow%=XB%+YB%:erow%=XE%+YE%:adv%=b
row%-erow%:dev%=ABS(XE%-YE%)
2570 IF ch%<=7 asst%=adv%*10+brow%*3-de
v%*3-len%*2+RND(3)
2580 IF ch%>7 asst%=adv%*10+brow%*10-AB
S((XE%-YE%)-(Xvac%-Yvac%))*10-len%*2
2590 IF asst%<bestasst% ENDPROC
2600 bestasst%=asst%
2610 FOR M%=0 TO 6:bestmv%(M%)=thismv%(
M%):NEXT
2620 ENDPROC
2630 :
2640 DEFPROCcmove
2650 cr=ABS(bestmv%(0)):PROCfind:PROCfl
ash

```

```

2660 IF bestmv%(0)<0 D%=bestmv%(1):dum=
FNcanslide:PROCstep:REM Sliding move
2670 IF bestmv%(0)>0 M%=1:REPEAT D%=bes
tmv%(M%):dum=FNcanjump:PROCstep:M%=M%+1:
UNTIL bestmv%(M%)=0:REM Jumping move
2680 ENDPROC
2690 :
2700 DEFPROCwin(w$)
2710 CLS:FOR W%=1 TO 20:PRINT w$+" WON!
!":NEXT:PROCmusic
2720 ENDPROC
2730 :
2740 DEFPROCmusic
2750 RESTORE 2760
2760 DATA 149,8,137,8,129,8,137,8,149,8
,137,8,129,4,137,4,117,4,0,0
2770 TIME=0
2780 REPEAT READ ptch,durn
2790 SOUND&101,1,ptch,durn
2800 SOUND&102,1,ptch-28,durn
2810 UNTIL ptch=0
2820 REPEAT UNTIL TIME>300
2830 ENDPROC
2840 :
2850 DEFFNget(r$)
2860 REPEAT:a$=GET$:UNTIL INSTR(r$,a$)>
0
2870 COLOURylw:PRINTa$:COLOURwht
2880 =a$
2890 DEFFNxs =950+(50*(X%-Y%))
2900 DEFFNyys =100+(70*(X%+Y%))
2910 DEFFNhhome
2920 count%=0:FOR X%=3 TO 6:FOR Y%=3 TO
6
2930 IF X%+Y%>8 AND b%(X%,Y%)>=0 AND b%
(X%,Y%)<=9 count%=count%+1
2940 NEXT:NEXT
2950 =count%
2960 DEFFNchome
2970 LOCAL X%,Y%
2980 count%=0:FOR X%=0 TO 3:FOR Y%=0 TO
3
2990 IF X%+Y%<4 AND b%(X%,Y%)<10 Xvac%=
X%:Yvac%=Y%
3000 IF X%+Y%<4 AND b%(X%,Y%)>=10 count
%=count%+1
3010 NEXT:NEXT
3020 =count%
3030 DEFFNoffboard
3040 XE%=X%+Xinc%(D%):YE%=Y%+Yinc%(D%)
3050 =XE%<0 OR XE%>6 OR YE%<0 OR YE%>6
3060 DEFFNcanslide
3070 IF FNoffboard THEN =FALSE ELSE =b%
(XE%,YE%)=-1
3080 DEFFNcanjump
3090 X1%=X%:Y1%=Y%
3100 t1=NOT FNcanslide:X%=XE%:Y%=YE%
3110 t2=FNcanslide:X%=X1%:Y%=Y1%
3120 =t1 AND t2

```

BEEBUG MAGAZINE is produced by BEEBUG Publications Ltd.
 Editor: Mike Williams
 Assistant Editor: Geoff Bains
 Production Editor: Phyllida Vanstone
 Assistant Production Editor: Yolanda Turuelo

Technical Assistant: Alan Webster
 Secretary: Debbie Sinfield
 Managing Editor: Lee Calcraft
 Additional thanks are due to Sheridan Williams, Adrian Calcraft, John Yale and Tim Powys-Lybbe.

All rights reserved. No part of this publication may be reproduced without prior written permission of the Publisher. The Publisher cannot accept any responsibility, whatsoever for errors in articles, programs, or advertisements published. The opinions expressed on the pages of this journal are those of the authors and do not necessarily represent those of the Publisher, BEEBUG Publications Limited.

BEEBUG Publications Ltd (c) 1985
 Editorial Address

BEEBUG
PO BOX 50,
Holywell Hill,
St. Albans AL1 3YS

**CONTRIBUTING TO BEEBUG
 PROGRAMS AND ARTICLES**

We are always seeking good quality articles and programs for publication in BEEBUG. All contributions used are paid for at up to £40 per page, but please give us warning of anything substantial that you intend to write. A leaflet, 'Notes of Guidance for Contributors' is available on receipt of an A5 (or larger) SAE.

In the case of material longer than a page, we would prefer this to be submitted on cassette or disc in machine readable form using "Wordwise", "View", or other means, but please ensure an adequate written description of your contribution is also included. If you use cassette, please include a backup copy at 300 baud.

HINTS

There are prizes of £5 and £10 for the best hints each month, plus one of £15 for a hint or tip deemed to be exceptionally good.

Please send all editorial material to the editorial address above. If you require a reply it is essential to quote your membership number and enclose an SAE.

SUBSCRIPTIONS

Send all applications for membership, subscription renewals, subscription queries and orders for back issues to the subscriptions address.

MEMBERSHIP SUBSCRIPTION RATES

£ 6.40 6 months (5 issues) UK ONLY
 £11.90 UK - 1 year (10 issues)
 £18 Europe, £21 Middle East
 £23 Americas & Africa, £25 Elsewhere

BACK ISSUES
 (Members only)

Vol	Single issues	Volume sets (10 issues)
1	90p	£8
2	£1	£9
3	£1.20	£11
4	£1.20	—

Please add the cost of post and packing as shown:

DESTINATION	First issue	Each subsequent issue
UK	30p	10p
Europe	70p	20p
Elsewhere	£1.50	50p

All overseas items are sent airmail (please send a sterling cheque). We will accept official UK orders but please note that there will be a £1 handling charge for orders under £10 that require an invoice. Note that there is no VAT on magazines.

Back issues are for members only, so it is ESSENTIAL to quote your membership number with your order. Please note that the BEEBUG Reference Card and BEEBUG supplements are not supplied with back issues.

Subscriptions, Back Issues &
 Software Address

BEEBUG
PO BOX 109
St. Johns Road
High Wycombe HP10 8NP

Hotline for queries and software orders

St. Albans (0727) 40303
 Manned Mon-Fri 9am-4.30pm

24hr Answerphone Service for Access and
 Barclaycard orders, and subscriptions
 Penn (049481) 6666

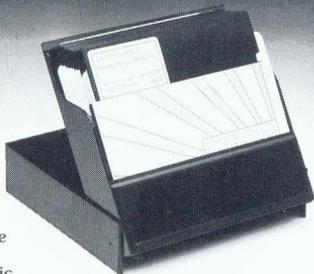
If you require members' discount on software it is essential to quote your membership number and claim the discount when ordering.

DYNAMIC DISCS

TESTED BY BEEBUG

BEEBUG, the largest independent computer user group in the UK, offer 100% tested discs supplied by one of Britain's leading disc manufacturers.

10 FREE LIBRARY CASE



Orders for 10 discs are sent in black plastic library cases.

25 FREE STORAGE BOX



Orders for 25 are delivered in strong plastic Storage box with 4 dividers.

50 FREE STORAGE BOX



Orders for 50 are delivered in strong plastic Storage box with 4 dividers.

COMPARE PRICES

4 Types of Disc To Meet Your Exact Requirement

48 TPI		DOUBLE DENSITY	
10	S/S D/D	£14.90	10 D/S D/D £20.50
25	S/S D/D	£34.90	25 D/S D/D £46.20
50	S/S D/D	£59.30	50 D/S D/D £82.40
96 TPI		DOUBLE DENSITY	
10	S/S D/D	£20.50	10 D/S D/D £21.90
25	S/S D/D	£46.20	25 D/S D/D £49.90
50	S/S D/D	£82.40	50 D/S D/D £93.50

All prices include Storage Box, VAT and delivery to your door (UK)

Suitable for BBC Micro and all other computers using 5 1/4 inch discs including Atari and Commodore.

Fully Guaranteed - Not only by Beebug but by one of the UK's top disc manufacturers.

ORDER FORM

Official orders welcome
Access/Barclaycard
24 hrs line 0494 816666
Further information
on helpline 0727 40303

Please supply me

_____ @ £ _____
 _____ @ £ _____
 Total cheque enc. £ _____

Name _____

Address _____

To Beebugsoft, PO Box 109, High Wycombe, Bucks. HP10 8NP

**BEEBUG
SOFT**

Magazine Cassette/Disc

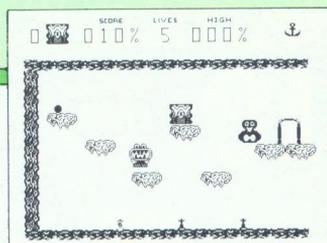
DECEMBER 1985 CASSETTE DISC CONTENTS

BEEBUG FILER – combined parts 1 and 2 of the BEEBUG database manager
FULL SCREEN BASIC EDITOR – full screen editing of Basic programs
WORDWISE PLUS EXAMPLES – final segment programs from the series
PERSONAL DIARY MANAGER – get 1986 off to a good start
SURE SAVE UTILITY – no more lost programs
PROGRAMMING SIDWAYS ROM AND RAM – storing and loading Basic programs
WORKSHOP PROCEDURES – Bubble Sort and Quick Sort
FIRST COURSE – colour blending demonstration
CHINESE CHEQUERS – a traditional game for Christmas

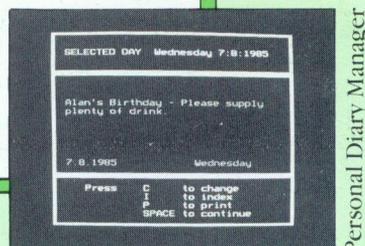
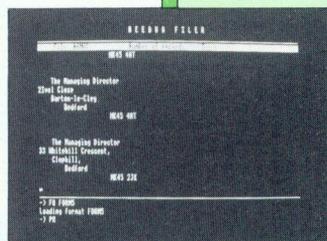
EXTRA FEATURES THIS MONTH

MORFIX – a delightful game to entertain and challenge you this Christmas, and with associated competition (see supplement for details)
MAGSCAN – data for this issue of BEEBUG (Vol. 4 No. 7)

Morfix



BEEBUG Filer



Personal Diary Manager

All this for £3.00 (cass) £4.75 (disc) +50p p&p.

Back issues (disc since Vol. 3 No. 1, cass since Vol. 1 No. 10) available at the same prices.

Subscription rates	DISC	CASS	DISC	CASS
	UK	UK	O'seas	O'seas
6 months (5 issues)	£25	£17	£30	£20
12 months (10 issues)	£50	£33	£56	£39

Prices are inclusive of VAT and postage as applicable. Sterling only please.

Cassette subscriptions can be commuted to disc subscription on receipt of £1.70 per issue of the subscription left to run.

All subscription and individual orders to
BEEBUG, PO BOX 109, St. Johns Road, High Wycombe HP10 8NP