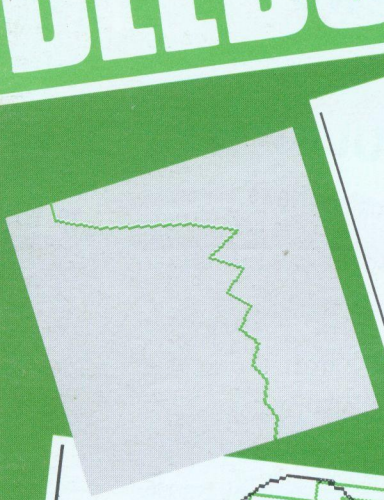
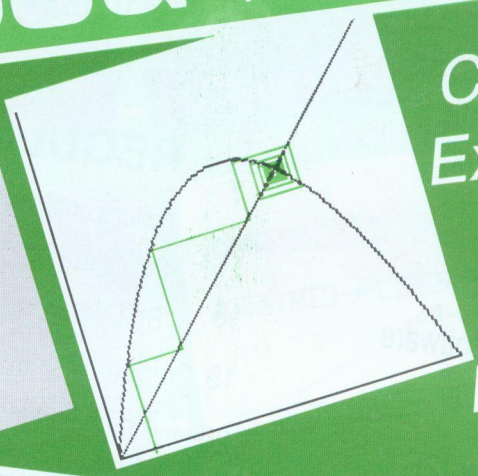
Vol.12 No.2 June 1993

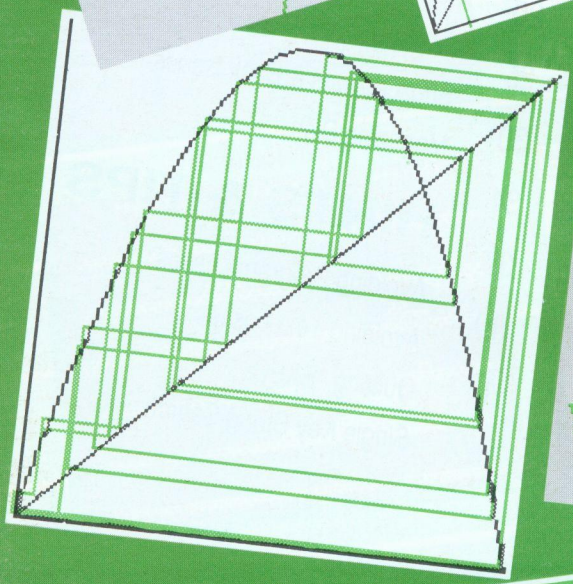
BEEBUG

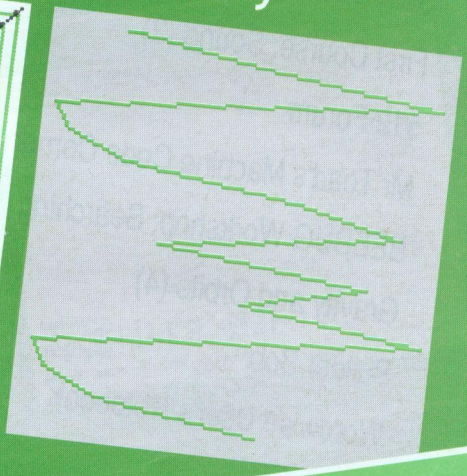
FOR THE BBC MICRO & MASTER SERIES





CHAOS: Exploring the Logistic Equation y=x(1-rx)





- ASTRO
- BUILDER BOB
- BIG NUMBERS
- SEARCHING

BEEBUG Vol.12 No.2 June 1993

FEATURES

REVIEWS

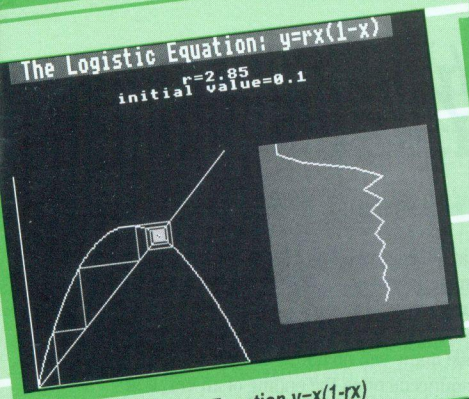
REGULAR ITEMS

HINTS & TIPS

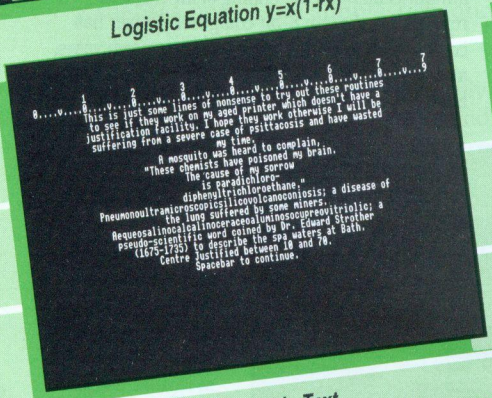
PROGRAM INFORMATION

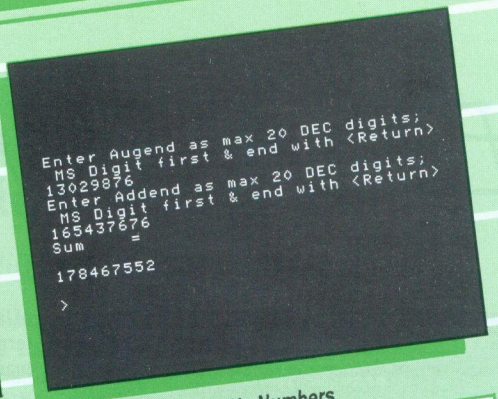
All listings published in BEEBUG magazine are produced directly from working programs. They are formatted using LISTO 1 and WIDTH 40. The space following the line number is to aid readability only, and may be omitted when the program is typed in. However, the rest of each line should be entered exactly as printed, and checked carefully. When entering a listing, pay special attention to the difference between the digit one and a lower case l (L). Also note that the vertical bar character (Shift \) is reproduced in listings as |.

All programs in BEEBUG magazine will run on any BBC micro with Basic II or later, unless otherwise indicated. Members with Basic I are referred to the article on page 44 of BEEBUG Vol.7 No.2 (reprints

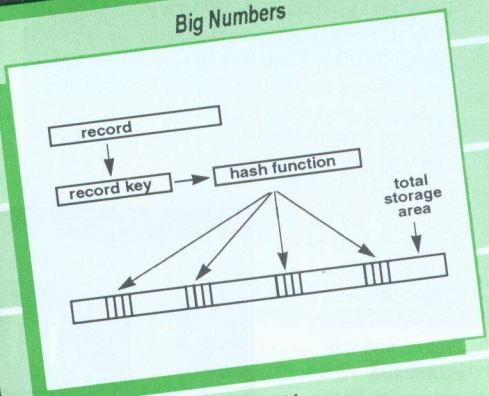


Logistic Equation y=x(1-rx)

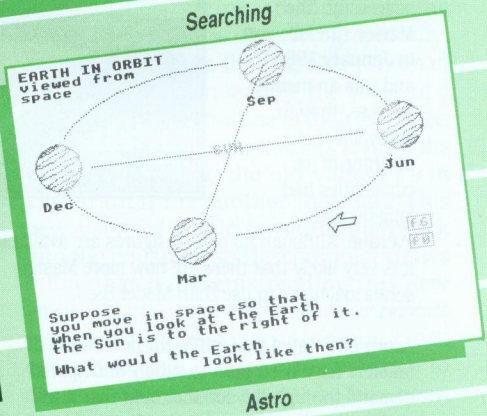




Big Numbers

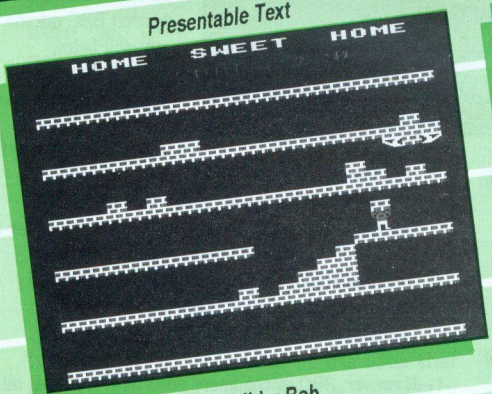


Searching



Astro

Presentable Text



Builder Bob

available on receipt of an A5 SAE), and are strongly advised to upgrade to Basic II. Any second processor fitted to the computer should be turned off before the programs are run.

Where a program requires a certain configuration, this is indicated by symbols at the beginning of the article (as shown opposite). Any other requirements are referred to explicitly in the text of the article.

Program needs at least one bank of sideways RAM.

Program is for Master 128 and Compact only.

Editor's Jottings/News

Master 128 Heads for Retirement

Acorn Computers has announced that the BBC Master 128 ceased production as at the beginning of May this year. The Master was based on the original BBC Model B design, but as readers will know, featured much more memory in the form of shadow, sideways and private RAM, plus additional software like View and ViewSheet in ROM, and many other features.

The Model B was first announced in 1981, chosen to support the BBC's Computer Literacy Project, and was enormously successful. The Master 128 followed in January 1986 and was an instant success, though many felt that a machine of its capabilities had long become overdue. Although no precise figures are available it is very likely that there are now more Master series machines in use than Model Bs.

Acorn has stated that sufficient new stocks of Masters have been built to satisfy anticipated demand through the summer, and that it will continue to provide spare parts for a minimum of five more years, as well as taking orders for peripheral software and hardware products while stocks last.

A spokesman for Acorn said that, "Sales of the older 8-bit computers have been declining rapidly over the past couple of years to the point where the cost of manufacturing these models is rendered uneconomic. In addition, there has also of late been great difficulty in securing the longer term supply of some of the components for their manufacture."

While many will no doubt mourn Acorn's announcement as the ending of an era, it is remarkable that an 8-bit computer system has continued to sell unchanged for over seven years, and that the basic concept for the machine has lasted nearly twelve. There are few, if any, other micros which could claim such longevity and such a faithful following.

In spite of this, it is by no means the end of the BBC micro. There are many systems still in regular use, and indeed BBC micros are still being bought by those entering the world of computing for the first time. There will no doubt continue to be a thriving secondhand market for some time to come, and those (particularly in education perhaps) who still seek a BBC micro or Master will be able to purchase systems from companies like Beebug who test each machine before resale and provide a warranty.

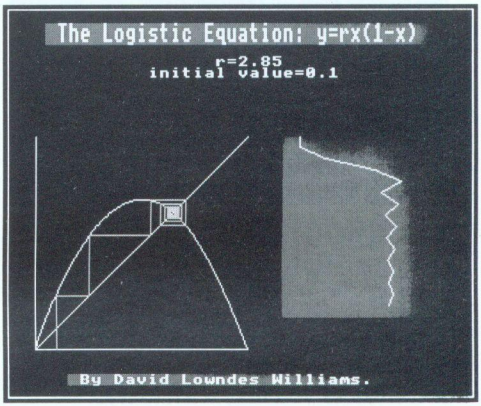
However, BBC micro users need to accept that all development and innovation is now in the 32-bit world, and while the BBC micro will continue to provide many loyal users with all their needs for the foreseeable future, Acorn has surely made the only decision it could. It is a measure of its support and the popularity of the Master 128 that the decision was not made sooner.

M.W.

Exploring the Logistic Equation y=x(1-rx)

David Lowndes Williams tells you what it means.

Chaos is a popular subject in computing circles because it demonstrates the curious paradox of arithmetical precision yielding, apparently, indeterminable behaviour. The program presented here visualises one of the simplest chaotic systems; the logistic equation.

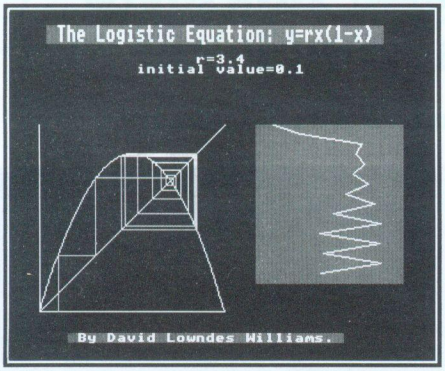


The logistic equation is used iteratively. A number is put into the equation, and the result fed back into the equation a second time for a subsequent iteration. There are two basic parameters in the logistic equation which together uniquely define the system. For the same pair of these values, identical behaviour will result. The parameter of fundamental importance is *r*, which is a constant that influences the type of behaviour that will occur. Values less than two give predictable, unchaotic behaviour and values close to four give very chaotic behaviour; the value of r must not go above 4. The value of the variable *initial* is a parameter that you put into the equation and must lie between 0 and 1 otherwise the iteration will diverge and chaos will not result. When in the chaotic regime; changing the value of *initial*, even by a small amount, changes the resulting output from the equation. This is known as the butterfly effect, referring to the idea that a very small change, like a butterfly landing on a leaf, can eventually lead to a huge result, like a tornado.

PROGRAM DESCRIPTION

Type in the listing below and save it. The values of *r*, *initial* and *threshold* are set at lines 130 to 150. When run, the program draws the curve $y=r(1-rx)$ and also the curve $y=x$ (a straight line). The iterative process then begins, starting with the initial value. This is depicted graphically; the program plots the line that a human would draw when reading the value of y from the graph. This line is then extended to the $x=y$ line and represents setting the new value of x to the old value of y in preparation for another iteration. This new value of x is to be substituted back into the equation, so the computer draws a line that represents reading this new value off the graph. This process repeats until two successive values lie close enough to each other for the program to assume some steady value has been reached and the iterative process terminates. This tolerance is set by the variable *threshold* so, when it's set to zero, the iteration never stops. This facility was included for the investigation of steady state solutions (i.e. r<3).

While this is all going on, the right hand side of the screen acts like a chart recorder, plotting values of the function.

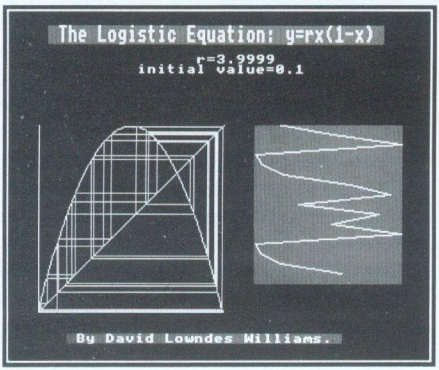


This is performed by defining a text window, drawing the trace and then forcing the window to scroll, using a PRINT statement. Holding down Ctrl-Shift will pause the program.

Some values you may wish to try:

r=2 *initial*=0.1 *threshold*=0.0001
r=2.85 *initial*=0.1 *threshold*=0.0001
r=3.1 *initial*=0.1 *threshold*=0
r=3.48 *initial*=0.1 *threshold*=0
r=3.7 *initial*=0.1 *threshold*=0

These show many of the interesting features of the equation.



CONCLUSION

Although programs like this have been published in BEEBUG before, this one is interesting in that both the chart recorder trace and pictorial representation are depicted simultaneously on the screen, giving a simple and straightforward demonstration of the onset of chaos.

```
 10 REM Program LOGISTIC
 20 REM Version B 0.1
 30 REM Author  David Lowndes Williams
 40 REM BEEBUG  June 1993
 50 REM Program subject to copyright
 60 :
100 MODE1
110 ON ERROR GOTO 320
120 :
130 r=3.99999
140 initial=0.1
150 threshold=0.0001
160 :
170 PROCinitScreen
180 x=initial
190 X=x:Y=FNy(x)
200 GCOL0,1:MOVE X*S,0:DRAW X*S,Y*S
210 REPEAT
220 y=FNy(x)
230 GCOL0,3:PROCdiagram
240 PROCtrace
250 GCOL0,1:PROCdiagram
260 X=x:Y=y
270 x=y
280 UNTIL ABS((X-y))<threshold
290 :
300 END
310 :
320 IF ERR=17 AND INKEY-1 END
330 REPORT:PRINT" at line ";ERL
340 END
350 :
360 :
370 DEF FNy(x)=r*x*(1-x)
380 :
390 DEF PROCdiagram
400 MOVE Y*S,Y*S
410 DRAW x*S,y*S
420 DRAW y*S,y*S
430 ENDPROC
440 :
450 DEF PROCdefineTRACE
460 VDU28,25,25,39,10
```

Continued on page 10

Big Numbers

Harry Fensom puts you in touch with the infinite (almost).

The following describes a set of routines which you can use in your own programs to perform arithmetic with numbers of up to several thousand digits. This can be of a great deal of use in educational applications for pure maths, it takes you well beyond the capabilities of any calculator, and will have its uses in programs that deal with astronomical investigations. It is also interesting in that it shows, once again, that the power of the small machine is often under-estimated and that we, as programmers, have still to explore its limits.

These programs will overcome the limitation of string lengths which are imposed on such programs as were described in the articles by Richard Beck in BEEBUG Vol.7 Nos.9 & 10. The actual calculation is done using 6502 machine code for speed, and peripheral operations use Basic. This enables the user to incorporate them easily in their own programs. Integer arithmetic is used for simplicity and accuracy; it is the required form when using these programs to solve puzzles, and to work with problems in number theory. By using appropriate multipliers, calculations can be done in fixed point arithmetic. The decimal point can then be fixed by inspection to achieve the effect of floating point if required. The limitations of the programs depends on the time taken to do the necessary iterations and ultimately on the computer memory available.

The operations provided this month are addition and subtraction; I will be covering multiplication, division and square root in a later article. All calculations use unsigned decimal integers; signs have to be sorted out by the user. There is no rounding - results are accurate to the last digit.

NOTES ON PROGRAMMING METHODS

Numbers are stored in byte arrays with two digits to a byte. Normally, when using indirect indexing of arrays, the maximum value which can be given to the Y register is 255. This limits the addresses which can be accessed, and thus the length of the integers, but this can be overcome in at least two ways. The method used here is the most flexible and helps, for example, in the square root program, so it has been used in all the others as well.

The method is to use the indirect construction as usual, but to leave the value of Y set to zero. Pointers used as indices are stored as two adjacent bytes in zero page and are added to the two corresponding bytes of a pointer to the base address of the array also in zero page. To index any element up to 65535 it is only necessary to increment, or change as needed, the two bytes of the sum pointer and use the construction (pointer),Y. Counters are treated in the same way by being represented as two bytes in zero page and can thus deal with very large numbers. These constructs in zero page effectively act as a set of two byte, (16 bit) registers.

PROGRAMS

The listing for the Addition program is given below, and you'll find it on the magazine disc as *ADDlarg*. To obtain a listing for the Subtraction program (*SUBlarg* on the disc) it is only necessary to change parts of nine lines of the former. These are as follows:

Line	Replace	With
10	ADDlarg	SUBlarg
20	Addition	Subtraction
150	Augend	Minuend
170	Addend	Subtrahend
180	add	sub
190	Sum	Difference
5000	add	sub
5120	CLC	SEC
5150	ADC	SBC

USING THE PROGRAMS

Enter the listing as given, save it and then run it. You will be asked for the maximum number of digits you wish to use. keep in mind the processing time here and enter a value which will avoid unnecessary time delay in performing the calculation. 5000 digits take a few seconds for on a BBC B for these two programs, but longer times for the other operations. The value chosen must take into account intermediate results if incorporated in your own programs for evaluating expressions. Follow this with Return. You will then be asked to enter the first and second numbers of the operation you want to carry out; numbers are entered with the most significant digit first, i.e. from left to right, the number is terminated with the Return key. The program introduces you here to a few mathematical terms; Augend and Minuend are the first numbers of an addition or subtraction operation respectively, Addend and Subtrahend are the second numbers in each calculation.

Once both numbers have been entered the calculation will take place and will be displayed when complete. Leading zeros are removed from the display but not from memory.

In my next article I shall look at the more complex operations of multiplication and division but, in the mean time, you might like to use these procedures to calculate the national debt!

```
Enter Augend as max 20 DEC digits;
 MS Digit first & end with <Return>
13029876
Enter Addend as max 20 DEC digits;
 MS Digit first & end with <Return>
165437676
Sum     =

178467552

>
```

Adding two very big numbers

```
  10 REM Program ADDlarg
  20 REM Version B1.0
  30 REM Author  Harry W Fensom
  40 REM BEEBUG  June 1993
  50 REM Program Subject to Copyright
  60 :
 100 INPUT"Maximum no. of digits "m%:ma
x%=m%+2:bytes%=max%/2
 110 PROCinit
 120 PROCassemble
 130 CLS
 134 ?ar%=?opd1p%:ar%?1=opd1p%?1:CALLcl
ear
 136 ?ar%=?opd2p%:ar%?1=opd2p%?1:CALLcl
ear
 138 ?ar%=?resp%:ar%?1=resp%?1:CALLclea
r
 140 ?temlp%=opd1%MOD256:temlp%?1=opd1%
DIV256
 150 PROCenter("Augend")
 160 ?temlp%=opd2%MOD256:temlp%?1=opd2%
DIV256
 170 PROCenter("Addend")
 180 CALLadd
 190 PROCdisplay("Sum   ",resp%)
 200 END
 210 :
1000 DEF PROCinit
1010 DIMbu% max%,opd1% bytes%,opd2% byt
es%,res% bytes%
1030 DIMmc% &180
1040 opd1p%=&70
1050 opd2p%=&72
1060 bup%=&74
1070 temlp%=&76
1080 tem2p%=&78
```

```
 1090 tem3p%=&7A
 1100 len%=&7C
 1110 mz%=&7E
 1120 x%=&80
 1130 flag%=&82
 1140 resp%=&84
 1150 cnt%=&86
 1154 ar%=&88
 1156 oswrch=&FFEE
 1160 ?opd1p%=opd1%MOD256:opd1p%?1=opd1%
DIV256
 1170 ?opd2p%=opd2%MOD256:opd2p%?1=opd2%
DIV256
 1180 ?bup%=bu%MOD256:bup%?1=bu%DIV256
 1190 ?resp%=res%MOD256:resp%?1=res%DIV2
56
 1200 ?mz%=bytes%MOD256:mz%?1=bytes%DIV2
56
 1210 ENDPROC
 1220 :
 2000 DEF PROCenter(N$)
 2010 PRINT'"Enter "N$" as max ";m%;" DE
C digits;"'" MS Digit first & end with <
Return>"
 2020 ?ar%=?bup%:ar%?1=bup%?1:CALLclear
 2030 I%=0
 2040 REPEAT
 2050 A1%=GET-48:IFA1%=79I%=I%-1:VDU127:
GOTO2050
 2060 IFA1%<10ANDA1%>-1PRINT;A1%;:I%?bu%
=A1%
 2070 I%=I%+1
 2080 UNTILI%=max%ORA1%=-35
 2090 IF I%=1 GOTO 130
 2100 ?len%=(I%-2)MOD256:len%?1=(I%-2)DI
V256
 2110 ?bup%=bu%MOD256:bup%?1=bu%DIV256
 2120 CALLinpt
 2130 ENDPROC
 2140 :
 2150 DEF PROCdisplay(N$,T%)
 2160 IFN$<>""PRINT'N$" = "'
 2170 ?ar%=?T%:ar%?1=T%?1
 2180 CALLdisp
 2190 PRINT'
 2200 ENDPROC
 2210 :
 2220 DEF PROCassemble
 2230 FORpass%=0TO2STEP2
 2240 P%=mc%:[OPTpass%
 2250 .inpt LDY#0
 2260 LDAlen%:STAcnt%
 2270 LDAlen%+1:STAcnt%+1
 2280 CLC:LDAbup%:ADCcnt%:STAbup%
 2290 LDAbup%+1:ADCcnt%+1:STAbup%+1
 2300 .iloop LDA(bup%),Y:STA(tem1p%),Y
 2310 SEC:LDAcnt%:SBC#1:STAcnt%
 2320 LDAcnt%+1:SBC#0:STAcnt%+1
 2330 BCCiexit
 2340 SEC:LDAbup%:SBC#1:STAbup%
 2350 LDAbup%+1:SBC#0:STAbup%+1
 2360 LDA(bup%),Y
 2370 ASLA:ASLA:ASLA:ASLA
 2380 ORA(tem1p%),Y:STA(tem1p%),Y
 2390 CLC:LDAtem1p%:ADC#1:STAtem1p%
 2400 LDAtem1p%+1:ADC#0:STAtem1p%+1
 2410 SEC:LDAbup%:SBC#1:STAbup%
 2420 LDAbup%+1:SBC#0:STAbup%+1
 2430 SEC:LDAcnt%:SBC#1:STAcnt%
 2440 LDAcnt%+1:SBC#0:STAcnt%+1
 2450 BCSiloop
 2460 .iexit RTS
 2470 :
 2480 .disp LDY#0:LDX#0
 2490 LDAmz%:STAcnt%:LDAmz%+1:STAcnt%+1
 2500 .brd CLC:LDAar%:ADCcnt%:STAtem1p%:
STAtem2p%
 2510 LDAar%+1:ADCcnt%+1:STAtem1p%+1:STA
tem2p%+1
 2520 LDA(tem1p%),Y:LSRA:LSRA:LSRA:LSRA
 2530 CPX#1:BEQpnt
 2540 CMP#0:BEQnxt1:LDX#1
 2550 .pnt CLC:ADC#48:JSRoswrch
 2560 .nxt1 LDA(tem2p%),Y:AND#&F
 2570 CPX#1:BEQpnt2
 2580 CMP#0:BEQrpt:LDX#1
 2590 .pnt2 CLC:ADC#48:JSRoswrch
 2600 .rpt SEC:LDAcnt%:SBC#1:STAcnt%
 2610 LDAcnt%+1:SBC#0:STAcnt%+1
 2620 BITcnt%+1:BPLbrd
 2630 TXA:BNEdexit
 2640 LDA#48:JSRoswrch
 2650 .dexit RTS
 2660 :
 2670 .clear LDY#0
 2680 LDAmz%:STAcnt%:LDAmz%+1:STAcnt%+1
```

```
2690 .brc CLC:LDAar%:ADCcnt%:STAtem2p%
2700 LDAar%+1:ADCcnt%+1:STAtem2p%+1
2710 LDA#0:STA(tem2p%),Y
2720 SEC:LDAcnt%:SBC#1:STAcnt%
2730 LDAcnt%+1:SBC#0:STAcnt%+1
2740 BITcnt%+1:BPLbrc
2750 RTS
2760 :
2770 .add LDY#0
2780 STYx%:STYx%+1
2790 CLC:LDAopd1p%:ADCx%:STAtem1p%
2800 LDAopd1p%+1:ADCx%+1:STAtem1p%+1
2810 CLC:LDAopd2p%:ADCx%:STAtem2p%
2820 LDAopd2p%+1:ADCx%+1:STAtem2p%+1
2830 CLC:LDAresp%:ADCx%:STAtem3p%
2840 LDAresp%+1:ADCx%+1:STAtem3p%+1
2850 LDA#0:STAflag%:PHP
2860 .aloop1 PLP
2870 SED
2880 LDAflag%:BNEaskip
2890 CLC
2900 .askip
```

```
2910 LDA(tem1p%),Y
2920 ADC(tem2p%),Y
2930 STA(tem3p%),Y
2940 PHP
2950 CLD
2960 CLC:LDAtem1p%:ADC#1:STAtem1p%
2970 LDAtem1p%+1:ADC#0:STAtem1p%+1
2980 CLC:LDAtem2p%:ADC#1:STAtem2p%
2990 LDAtem2p%+1:ADC#0:STAtem2p%+1
3000 CLC:LDAtem3p%:ADC#1:STAtem3p%
3010 LDAtem3p%+1:ADC#0:STAtem3p%+1
3020 CLC:LDAx%:ADC#1:STAx%
3030 LDAx%+1:ADC#0:STAx%+1
3040 LDA#1:STAflag%
3050 LDAx%:CMPmz%
3060 LDAx%+1:SBCmz%+1
3070 BCCaloop1
3080 PLP
3090 RTS
3100 ]:NEXT:ENDPROC
3110 :
3120 END
```

Exploring the Logistic Equation y=x(1-rx) (continued from page 6)

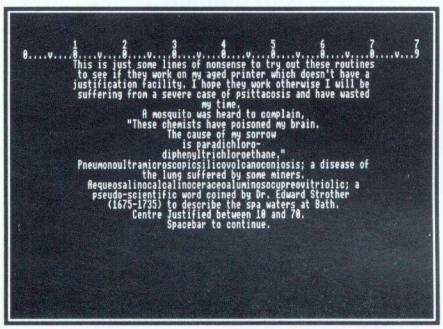
```
 470 COLOUR130:CLS:GCOL0,3
 480 FOR c%=1TO15:PRINT:NEXT
 490 ENDPROC
 500 :
 510 DEF PROCtrace
 520 v=x*480+700
 530 w=X*480+700
 540 MOVE v,92:DRAW w,92+32
 550 PRINT
 560 ENDPROC
 570 :
 580 DEF PROCinitScreen
 590 VDU19,2,4,0,0,0
 600 VDU29,100;100;
 610 COLOUR130
 620 PRINTTAB(4,0);:PROCdouble(CHR$32+"
The Logistic Equation:y=rx(1-x)"+CHR$32)
 630 PRINTTAB(6,31)CHR$32;"By David Low
ndes Williams.";CHR$32;
 640 COLOUR128
 650 PRINTTAB(19,3)"r=";r
 660 PRINTTAB(13,4)"initial value=";ini
tial
 670 S=600
```

```
 680 MOVE0,S:DRAW0,0:DRAWS,0
 690 MOVE0,0:DRAWS,S
 700 PROCdefineTRACE
 710 MOVE0,FNy(0)*S
 720 FOR x=0TO1 STEP0.01
 730 DRAW x*S,FNy(x)*S
 740 NEXT
 750 DRAW S,FNy(1)*S
 760 ENDPROC
 770 :
 780 PROCdouble(text$)
 790 DEF PROCdouble(text$)
 800 s%=&70
 810 FOR t%=1 TO LEN(text$)
 820 ?s%=ASC(MID$(text$,t%,1))
 830 X%=s% MOD 256:Y%=s% DIV 256:A%=10
 840 CALL&FFF1
 850 VDU23,128,?(s%+1),?(s%+1),?(s%+2),
?(s%+2),?(s%+3),?(s%+3),?(s%+4),?(s%+4)
 860 VDU23,129,?(s%+5),?(s%+5),?(s%+6),
?(s%+6),?(s%+7),?(s%+7),?(s%+8),?(s%+8)
 870 VDU128,8,10,129,11:NEXT
 880 ENDPROC
 890 :
```

Presentable Text

Derek Hoy helps you justify your existence.

When a program requires text to be sent to the screen, or to a printer without a justification facility, I use the two short routines *PROCformat()* and *FNpad()* given in the listing below (lines 1000 to 1190) to format and justify the text to my requirements. The only limitations to the text are that the original length must be less than 256 characters long and only include the normal printable characters. The routines are included in a demonstration program which shows the ways text, here in DATA statements, can be formatted.



Centred text

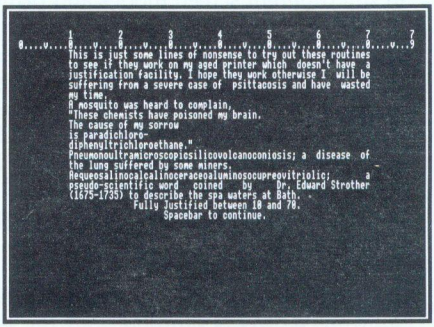
The routines themselves do not have any error trapping incorporated and will assume sensible parameters are sent to them. These include left and right margins and a minimum text width of two characters to allow for the hyphen in split words. The demonstration texts are output to 80 column screen and printer formats, so in the examples what you see is what you get.

The routines themselves should be compatible with any machine or printer since the program is pure Basic with no OS calls used, or printer codes other than VDU1,13 (a line feed). When typing in the listing, be careful not to omit the semi-colons in Lines 1070 and 1100.

HOW IT WORKS

Lines 100 to 440 set up left and right margins, select the justification required, screen or printer as the priority output, and print the demonstration text. The lines of print are padded out to 80 characters which effectively gives a line feed at each line of text.

PROCformat() uses four parameters, text string, left margin, right margin and a justification flag. For the latter, I have chosen the same values of 0 for Left, 1 for Centre, 2 for Right and 3 for Full as used by Epson on their printers with justification facilities. These values should not be altered since the flag is used to calculate padding spaces required. *FNpad()* uses two parameters, p$ and the justification flag, both passed from *PROCformat()*.



Fully justified text

PROCformat() initially calculates the text width, *w%*. If the text is longer than *w%*, it repeatedly splits the text string into two parts at the first space encountered

back from *w%*+2 or inserts a hyphen at the appropriate place if there is no space character in the first *w%* characters of the text. The left string is sent to *FNpad()* and the right string is further split as before until the residual string length is less than or equal to *w%*; whereupon the justification is changed to left if originally full and the residual string is sent to *FNpad()*. Text initially shorter than or equal to w% in length is sent here by the GOTO at line 1020

FNpad() returns a string padded with spaces as required by the type of justification to fill the length of the string to the text width. This is done by concatenating spaces to fill the left margin, adding leading spaces as calculated in line 1150 for right or centre justification, or randomly placing internal spaces next to an existing space character for full justification. It also adds trailing spaces to fill the length of the string to length w%.

MODIFICATIONS

The maximum value of the right margin, set in lines 170 and 180, may be changed to suit different screen modes and printer fonts; but it must be remembered that only the priority output will be properly formatted. Thus typical right margin maxima could be 19, 39 or 79 for screen modes 2, 7 or 3, and 39, 79, 95 or 136 for enlarged, pica, elite or condensed print fonts, and the value of 80 in line 1190 changed accordingly to right margin maximum+1.

Additionally, if the hyphen used in splitting words is omitted, and the text width is set to 1, *(R%=L%)* the same routines may be used to print vertically - to label graphs for example. This will only work on screen output and may be done by preceding the call to *PROCformat* by VDU31,x,y; and altering lines 1030 and 1060 to :

```
1030 REPEAT:i%=w%+1
1060 IF i%=1 l$=LEFT$(t$,w%):i%=w%
```

A demonstration of this may be seen by altering lines 1030 and 1060 as above, deleting lines 110-290, lines 320-420, line 440 and lines 1200 onward; and adding the following:

```
320 P%=FALSE 330 text$="X Axis"
340VDU 31,0,20:
PROCformat(text$,0,79,1)
350 text$="Y Axis" 360 VDU 31,0,5:
PROCformat(text$,5,5,0)
```

These routines should be useful for any program that requires a tidy output to screen or printer and add that little touch of class to your writing.

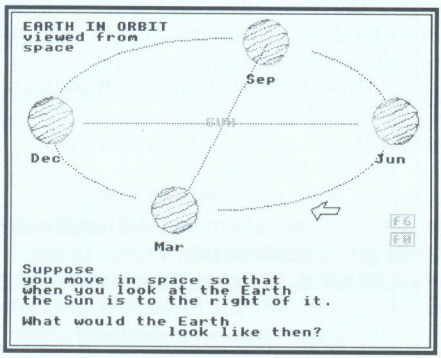
```
  10 REM Program JUSTIFY
  20 REM Version B1.0
  30 REM Author  Derek Hoy
  40 REM BEEBUG  June 1993
  50 REM Program Subject to Copyright
  60 :
 100 MODE 3:VDU19,1,2;0;:ON ERROR GOTO
470
 110 DIM j$(3):j$(0)="Left":j$(1)="Cen+
re":j$(2)="Right":j$(3)="Fully"
 120 REPEAT
 130 REPEAT
 140 CLS:PRINTTAB(0,5);"Enter Left Marg
in (0 to 78)";:INPUTL%
 150 UNTIL L%>=0 AND L%<79
 160 REPEAT:CLS
 170 PRINTTAB(0,5)"Enter Right Margin (
";L%+1;" to 79)";:INPUTR%
 180 UNTIL R%>=(L%+1) AND R%<80
 190 REPEAT
 200 REPEAT
 210 CLS:PRINTTAB(0,5)"Enter justificat
ion required"'"0 Left"'"1 Centre"'"2 Rig
ht"'"3 Full"'"4 Redefine Margins"'"5 End
":J%=GET AND &F
 220 UNTIL J%>=0 AND J%<6:IF J%=5 END
 230 IF J%=4 UNTIL TRUE:UNTIL J%<>4
```

```
  240 REPEAT
  250 CLS:PRINTTAB(0,5)"Printer or Scree
n (P/S) required":g%=GET AND &DF
  260 UNTIL g%=ASC"P" OR g%=ASC"S"
  270 IF g%=ASC"P" P%=TRUE ELSE P%=FALSE
  280 :
  290 CLS:IF P% VDU2 ELSE VDU14
  300 sf$=STRING$(9," "):text$=" "+sf$+"
1"+sf$+"2"+sf$+"3"+sf$+"4"+sf$+"5"+sf$+"
6"+sf$+"7"+LEFT$(sf$,8)+"7":PRINTtext$;
  310 sf$=STRING$(4,".")+"v"+STRING$(4,"
."):text$=LEFT$(STRING$(8,"0"+sf$),LENte
xt$-1)+"9":PRINTtext$
  320 text$="This is just some lines of
nonsense to try out these routines to se
e if they work on my aged printer which
doesn't have a justification facility."
  330 text$=text$+" I hope they work oth
erwise I will be suffering from a severe
 case of psittacosis and have wasted my
time."
  340 PROCformat(text$,L%,R%,J%):PRINT
  350 RESTORE1210:READ text$:PROCformat(
text$,L%,R%,J%)
  360 READ text$:PROCformat(text$,L%,R%,
J%)
  370 READ text$:PROCformat(text$,L%,R%,
J%)
  380 READ text$:PROCformat(text$,L%,R%,
J%)
  390 READ text$:PROCformat(text$,L%,R%,
J%):PRINT
  400 READ text$:PROCformat(text$,L%,R%,
J%):PRINT
  410 READ text$:PROCformat(text$,L%,R%,
J%):PRINT
  420 PROCformat(j$(J%)+" Justified betw
een "+STR$L%+" and "+STR$R%+".",0,79,1):
VDU3,15:PRINT
  430 PROCformat("Spacebar to continue."
,0,79,1):REPEAT UNTIL GET=32
  440 UNTIL FALSE
  450 END
  460 :
  470 ON ERROR OFF:REPORT:PRINT" at line
";ERL
  480 END
  490 :
 1000 DEF PROCformat(t$,l%,r%,j%)
 1010 LOCAL i%,w%,l$:w%=r%-l%+1
 1020 IF LENt$<=w% GOTO 1090
 1030 REPEAT:i%=w%+2
 1040 REPEAT:i%=i%-1:l$=LEFT$(t$,i%-1)
 1050 UNTIL MID$(t$,i%,1)=" " OR i%=1
 1060 IF i%=1 l$=LEFT$(t$,w%-1)+"-":i%=w
%-1
 1070 PRINTFNpad(l$,j%);:t$=RIGHT$(t$,LE
Nt$-i%)
 1080 UNTIL LENt$<=w%
 1090 IF j%=3 j%=0
 1100 PRINTFNpad(t$,j%);:IF P% VDU1,13
 1110 ENDPROC
 1120 :
 1130 DEF FNpad(p$,j%)
 1140 IF j%=0 OR LENp$=w% GOTO1190
 1150 IF j%<3 p$=STRING$((w%-LENp$)*j% D
IV 2," ")+p$:GOTO1190
 1160 LOCAL k%:IF INSTR(p$," ")=0 GOTO11
90
 1170 REPEAT:REPEAT:k%=RND(LENp$):UNTIL
MID$(p$,k%,1)=" "
 1180 p$=LEFT$(p$,k%)+" "+RIGHT$(p$,LENp
$-k%):UNTIL LENp$=w%
 1190 =STRING$(l%," ")+p$+STRING$(80-l%-
LENp$," ")
 1200 :
 1210 DATA"A mosquito was heard to compl
ain,"
 1220 DATA"""These chemists have poisone
d my brain."
 1230 DATA"The cause of my sorrow"
 1240 DATA"is paradichloro-"
 1250 DATA"diphenyltrichloroethane."""
 1260 :
 1270 DATA"Pneumonoultramicroscopicsilic
ovolcanoconiosis; a disease of the lung
suffered by some miners."
 1280 DATA"Aequeosalinocalcalinoceraceoa
luminosocupreovitriolic; a pseudo-scient
ific word coined by Dr. Edward Strother
(1675-1735) to describe the spa waters a
t Bath."
```

Astro

Lydia Wakefield gets her head in the clouds.

Astro is a package from the educational side of Topologica that will run on all versions of the Beeb. Aimed both at the school and the home it covers the astronomical bits of the National Curriculum around the top of Key Stage Two and into Key Stage Three (that's top primary/lower secondary) with some interesting extra bits.



The seasons

The package consists of a single disc with a technical information sheet, seven photocopiable worksheets and a book. The book is not just a manual but a proper, full colour text book that takes you through the various parts of the program with a full explanation of what it all means. There is no doubt that this will mean considerably less work for the teacher or parent using Astro, and it's a shame that some other packages are not so well supported.

The program presents a menu with information available on each of the seven parts of the package. A nice touch is that you can choose a particular program from the information pages so you don't have to keep going back to the menu.

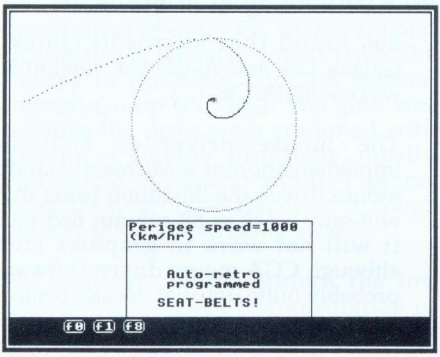
The first program looks at the track of the sun across the sky. The display shows the curve of the Earth with angles of elevation drawn above it. A short animated sequence demonstrates how these are arrived at and helps you understand exactly what you are looking at. You can set any latitude and month then step the clock round to see the track the sun makes. Because the package includes printer drivers for both Epson compatible and Integrex colour printers you can dump the screen at relevant times. Throughout the package this gives the option of setting up comparative screens to look at the effects of different latitudes and times of year. You could use a screen grabber program to do the same.

There are two programs that look at the sun from the point of view of the shadows it casts. The first shows a view of a stick, perpendicular to the ground, from above - again an animated sequence makes clear exactly what you are looking at. Setting the latitude and date is consistent with the other programs and, having done this, you step though the day with the shadows being cast by the stick. This is a simple and effective way of demonstrating the differences between summer and winter but it can also be used, along with the first program, to show how the length of days alter with latitude.

The second shadows program seems a little frivolous. It is essentially the same as the first but the angle of the stick can be changed - there is a certain amount of 'because we could' about this, but there is a short piece in the accompanying book about how sundials use this idea so it is not totally wasted.

The Globe section looks at the tilting of the Earth through the seasons. There is a detailed explanation available of what is going on as we orbit the sun. This is very clear and well-illustrated, and we then go on to look at the track

of the sun on the globe, again setting latitude and date.



A successful landing

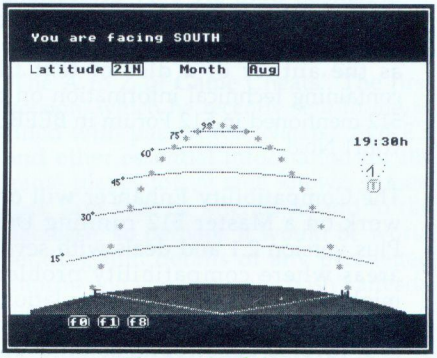
The Moon program shows the orbit of the moon around the Earth as it moves through space. First we see this in 'real time' and then we are shown several positions of the moon at once. Following this we can look at the shape of the moon appears to be from space and from Earth in any one of 12 points on its orbit.

Now we move on to the movements of less predictable objects in space; Comets and Spacecraft. A sub-menu here lets us look first at the way comets move. Those of you that have been following Cliff Blake's series on orbits in BEEBUG will find this section quite familiar as you try to change the path of a comet without losing it from the solar system. The program then moves on to explore spacecraft as 'comets with engines' and you get to try and land one.

Finally we get to the planets. This starts by showing us the orbits of the five inner planets around the sun and then goes on to show their orbits as viewed from Earth. This is an important idea, especially if you are interested in the ancient view of the universe.

At first sight, the package as a whole seems rather dry - what could you actually *do* with it? As it stands, the programs could be very useful as an animated blackboard in school to help get over the quite difficult concepts covered. However, once you start to use the worksheets and the book you will find whole areas of activity opened up.

What's really nice about all this is the way the authors have seen the computer as only part of the whole thing. The book, by Ray Hemmings and Derick Last, has you rushing about looking things up, observing the skies, building stuff and doing all sorts of experiments - you won't spend much time at the computer. Any busy teacher will find this pack invaluable as it gets children working.



The track of the sun

Home use is another question. It's always a problem with educational software that it's priced for schools. £30 (ex. VAT) for a site licence is actually quite cheap in those terms but it will seem a lot to the individual user. Having said that, if you have a child, aged around 10 to 15, who is interested in astronomy - or is studying it at school - this would be a very worthwhile investment if you were prepared to support all the activities.

Product	Astro
Supplier	Topologica PO Box 39, Peterborough PE7 3RL. Tel/fax (0733) 244682
Price	£30 ex. VAT (all formats)

Public Domain

A look at some Master 512 software and the latest BBC software to become available in this month's column by Alan Blundell.

Some time ago now, I said that I would come back to look again at software for the Master 512 co-processor when items of software particularly worthy of note came my way. This month, I would like to mention two programs, both of which are by the same author: David Harper. Both are specifically written for the Master 512 and will work only on that system, which puts them in a fairly exclusive range of software. The programs are a Microsoft-compatible mouse driver and PCCE, a PC Compatibility Enhancer. Both are very professionally produced and documented, to David's usual standard. You may remember him as the author of a disc of text files containing technical information on the 512 mentioned in 512 Forum in BEEBUG vol.11 No.8.

The Compatibility Enhancer will only work on a Master 512 running DOS-Plus version 2.1 and deals with several areas where compatibility problems might arise: PC keyboard emulation is improved; ASCII code 96 now appears as a reverse apostrophe as on a 'real' PC, rather than a pound sign; several minor incompatibilities are tidied up; and a way is provided to deal with what seems to me the most frustrating class of programs, those which work but slow down to a fraction of a usable speed. In some cases, this slowness is apparently due to a PC program trying (and failing) to speed up the system clock whilst it plays a sound - all the more frustrating since the sound isn't produced on the Master 512 anyway! All in all, this is a very pleasing program although it can't perform miracles and make all PC software work. David suggests that the most likely software to run with PCCE where it failed before includes games and 'pop-up' (TSR) programs, although he has found that Microsoft Quick C, Galaxy Lite and AsEasyAs version 5 all work with PCCE.

The mouse driver is a limited implementation of a Microsoft standard mouse driver, the limitation being that it will only work in 80 column text mode. It will not work in graphics mode, although CGA mouse driven software is probably quite unusual. Mouse control is most commonly of use in programs which do use 80 column text mode, such as word processors and spreadsheets, so the limitation is one that I could live with.

What impresses me most about these two programs is the depth of understanding of the 512 and its system software which is necessary before either of these could even be attempted as a programming task. I fancifully believe that I could make a reasonable stab at anything on the Beeb, having knocked out a few SWR programs and several other sundries, but I suspect that I am in the company of most 512 owners when it comes to system-level tricks on the 512, having little idea where to start.

David kindly made the source code to the mouse driver available and after studying it, I can see that I am a long way from reaching a level of understanding where I could even attempt such a program (so much for the RAM disc I planned to write when I first bought the co-pro...). Perhaps this explains why there have been so few software developments for the 512 (although those titles which have been produced are of a high standard); all the more reason to support and encourage those programmers who do produce enhancements.

David should also receive credit for volunteering to test some PC software for 512 compatibility for me last year. With a few pangs of guilt, I sent him a couple of dozen discs of older PC shareware and PD software, mostly taken from the library of the New York PC Special Interest Group (PC-SIG). The guilt was because the discs were crammed with dozens of smaller programs, mostly utilities and games, and because I had not found the time or the willpower to plough my way through the testing myself.

He manfully went through the lot, producing excellent documentation of what worked and what didn't, even continuing through a period when his letters to me got no response. I was struggling against a tide of correspondence for a time, with the delay aggravated by extra pressures in my 'real' job. I mention it here only because he deserves an apology and because there are a few others who may recognise the period. The net result was one 800K disc of assorted utilities, plus a few gems like KWSEARCH, a free text retrieval system, which essentially lets you use text files as a database, and COVER, which prints a 5" x 5" sleeve insert for discs, detailing their contents. Very handy if your disc collection gets out of hand! A BBC Micro version would be useful, if one doesn't exist already.

BBC PD

More new software has appeared for the standard BBC micro and Master 128. I thought a couple of months ago that the frequency of new developments was slowing down, but there are still mammoth tasks being undertaken. One such is the 'Yellow Discs' by John Bythell. 'Yellow Discs', which John developed over the last two years, is rather like the Telecom Yellow Pages.

Only recently completed, it is a cross-reference and index to all the major articles in all the major BBC micro magazines since they began. Because of the extent of the system and data, it only works with 80 track, double-sided DFS disc drives, although either one or two drives are acceptable. In my experience, over 90% of users with disc drives have 80 track double-sided drives anyway, so this perhaps isn't a limitation for most people. I haven't seen the comparable index systems which have been available commercially, so I can't compare 'Yellow Discs' with them, but it does seem entirely usable as a system to me. It won't be updated any further, though, because John has moved on from the BBC Micro.

Still under development is one of the few Australian packages which I have seen. Jock Smylie, who was mentioned here recently as having started a PD library for the BBC in Australia, has produced 'The A to Z of Game Cheats'. The first version, which he recently sent to me, is filled with passwords, cheats, 'POKE's and other essential information for the game player for 99 well known games. Jock says that he intends to add to the database, so perhaps it would be worth sending him details of any privileged information you have which isn't already included.

Finally for this month, it's a good time of year to mention a set of View text files which were put together by Mick Drury. Based on his own experience, they describe a practical way to save money on Gite holidays in France by booking them directly with the Gite owners and organising your own ferry tickets. Also, there is plenty of information about areas which are/are not worth visiting in terms of facilities which are available and, particularly for parents, areas which are good for entertaining the kids! There is also a guide to likely costs and even a ViewSheet holiday planner. I plan to make use of this material next time I go to France and would recommend it if you are planning a visit.

Hunt the Snib

Marshal Anderson invites you to improve on his efforts.

Having looked at all the wonderful and complex graphic-based games published in BEEBUG over the years it occurred to me that it might be a nice idea to go back to some of the more basic ideas behind some of those games and see if we can't generate something new.

SIT DOWN WHILE I TELL YOU A TALE

Hunt the Snib is a very straightforward hide and seek game. Type in the listing below and you'll see what I mean. The game takes place on a 20 by 20 grid: you hide in one position and the Snib hides in another. All inputs are co-ordinates separated by commas. You try to guess where the Snib is and enter your co-ordinates accordingly. Each time you do this the Snib will make its own guess as to where you are. After each guess you will be told, in compass points, in which direction the Snib is, you will also be told how close the Snib is to finding you. Whoever gets there first wins. When testing this on my own household there was considerable argument about clearing the screen after each set of co-ordinates; take out line 180 and see what you think.

SO WHAT?

I expect you will find the thrill of this particular chase palls after a while, though it can be more challenging than you think. The thing we want to do with this is look at ways of improving on the basic idea.

The first thing we might do is look at the way the Snib seeks us out. As it stands it's not really very good at it. Quite simply, it looks at the direction you are in and takes a random number between its last guess and the edge of the grid. This is unlikely to be the way you would search, you would be more likely to split the grid into smaller and smaller quadrants until you nail the little toe-rag. So, it might be an idea to get the Snib to remember where it has been so it doesn't keep going back there. This could be done fairly easily by using four variables that act as minimum and maximum x and y co-ordinates. If the Snib finds that you are, say, south of y co-ordinate 15 that could be stored as the maximum and not be exceeded.

Having made the Snib a better seeker we might like to make life easier for the human side of the equation. The best form of help would be a simple display of the grid, just dots would do, with compass points marked. A FOR - NEXT loop could do this and you could leave it on the screen with the text in a window at the bottom. A step further might be to have the chosen and guessed positions displayed on the grid. To do this it would be a good idea to move to a graphics mode.

Beyond that you could think about adding some sound and a rather friendlier input routine. What about giving the user some control over the size of the maze? The maze itself could be zapped up with a few nasty locations, how about mines or teleport squares that cause the position of the opponent to change, how about a two player version?

To dress it up graphically you could give it a specific location, dungeons under a

castle or corridors in a space station - you could be in space itself.

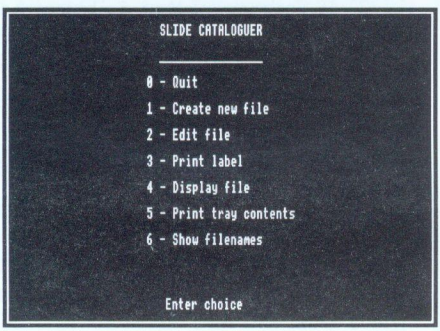
The next step is to take the idea of hide and seek out of the grid environment altogether - winding and unpredictable passages can be a lot of fun, but we'll leave that for another time.

```
   10 REM Program Snib
   20 REM Version B 1.0
   30 REM Author  Marshal Anderson
   40 REM BEEBUG  June 1993
   50 REM Program subject to copyright
   60 :
  100 MODE 7
  110 PROCgetstarted
  120 PROCplacesnib
  130 PROCgetguess
  140 IF Snibx%=Myguessx% AND Sniby%=Myg
uessy% PROCwin:GOTO 110
  150 PROCreply
  160 PROCsnibguess
  170 IF Snibguessx%=Myx% AND Snibguessy
%=Myy% PROClose:GOTO 110
  180 PROCpak:CLS
  190 GOTO130
  200 END
  210 :
 1000 DEF PROCgetstarted
 1010 CLS
 1020 PRINT"The Snib will hide somewhere
 on a 20x20 grid, can you find it before
 it finds"'"you?"''
 1030 PRINT"Where will you hide? Enter a
 grid"'"reference in the form x,y then p
ress"'"Return."
 1040 INPUT Myx%,Myy%
 1050 ENDPROC
 1060 :
 1070 DEF PROCplacesnib
 1080 Snibx%=RND(19)+1
 1090 Sniby%=RND(19)+1
 1100 Snibguessx%=RND(19)+1
 1110 Snibguessy%=RND(19)+1
 1120 PRINT"The Snib is hiding, ready or
 not..."
 1130 ENDPROC
 1140 :
 1150 DEF PROCgetguess
 1160 PRINT'"Type in your guess x,y then
 press"'"Return."
 1170 INPUT Myguessx%,Myguessy%
 1180 ENDPROC
 1190 :
 1200 DEF PROCreply
 1210 Reply$=""
 1220 IF Myguessy%<Sniby% Reply$=Reply$+
"north"
 1230 IF Myguessy%>Sniby% Reply$=Reply$+
"south"
 1240 IF Myguessx%<Snibx% Reply$=Reply$+
"east"
 1250 IF Myguessx%>Snibx% Reply$=Reply$+
"west"
 1260 IF Reply$="" ENDPROC
 1270 PRINT'"The snib is to the ";Reply$
 1280 ENDPROC
 1290 :
 1300 DEF PROCsnibguess
 1310 IF Snibguessy%<Myy% Snibguessy%=RN
D(19-Snibguessy%)+Snibguessy%
 1320 IF Snibguessy%>Myy% Snibguessy%=RN
D(Snibguessy%)-1
 1330 IF Snibguessx%>Myx% Snibguessx%=RN
D(Snibguessx%)-1
 1340 IF Snibguessx%<Myx% Snibguessx%=RN
D(19-Snibguessx%)+Snibguessx%
 1350 PRINT'"The Snib guesses ";Snibgues
sx%;",";Snibguessy%
 1360 ENDPROC
 1370 :
 1380 DEF PROCpak
 1390 PRINT'"Press a key"
 1400 A=GET
 1410 ENDPROC
 1420 :
 1430 DEF PROCwin
 1440 CLS:PRINT'''"Well done, another Sn
ib bites the dust!!"
 1450 PROCpak
 1460 ENDPROC
 1470 :
 1480 DEF PROClose
 1490 CLS:PRINT"Not good, the Snib has f
ound you."
 1500 PROCpak
 1510 ENDPROC
```

Slide Cataloguer (Part 2)

Jim Phillips concludes his Slide Cataloguing program.

Having typed in the first part of the program and possibly created some slide files you are no doubt eager to print something. Enter the second half of the program as listed below and save it under any temporary name in case of disaster. Now make a text file of it either by spooling or by typing the Basic command EDIT while the program is loaded. Save this under a different name by pressing f3 and return to Basic. Now load the SlCat1 program from last month, *EXEC the text file in and save the result as SlCat. Subscribers to the disc will find the complete Slcat on this months magazine disc but it must first be copied to an ADFS disc to work correctly.



The complete menu

On running the program you will find a much bigger menu and, of course, you will want to test each facility. I suggest you attempt the non-printing ones first.

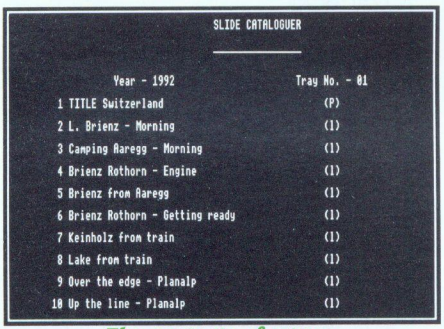
Press '6' and you will be asked for a year. First choose one for which you know a directory exists. A catalogue of this directory should be displayed. Now ask for a non existent one. The message - "No records for this year !" will be displayed for 4 seconds and then the program will return to the menu.

Now press '4' and follow the instructions. If you pick a non-existant directory you will get the display above. If the file itself does not exist, you will see "No record of this tray !". Otherwise the contents will be listed on the screen and can be scrolled with Shift.

Pick a short file for the printing tests - no point in wasting paper! The program is written for a standard Epson printer which you should now switch on. Selecting '5' and following the prompts will give a list of the tray contents and eject to a new page while '3' will give you the object of the exercise - the slide labels. First you will be asked if you want a tray label. 'Y' will give you a double width title of half the label width, on the left for even file numbers and on the right for odd. Next you will be asked whether you want to print all labels, selected labels or none. Labels will then be printed, six items to each adhesive label. Note that 'Portrait' shape labels have an asterisk on the right hand side and 'Landscape' on the left. When you affix the label to the slide this will orientate it in the projector (you wondered why you were asked that). All you now have to do is cut them out and stick them on.

We can now enter slide details and print the results, but what do we do if we get one wrong, want to change one or add

slides to an incomplete box? We press '2' and go into Edit. The title and number of the first slide will appear and if it is correct and 'Y' is typed then the next title will appear. If 'P' or 'L' is typed (for portrait or landscape) the entry box will appear and the new title can be written. On pressing Return the next title will appear as before and the process is repeated until there are no more slides.



The contents of a tray

When all corrections are done pressing 'A' will bring you to this point directly and 'S' will let you skip over correct entries. If there were less than 50, you will be told how many slots remain and asked if you wish to add more slides. If the answer is yes the entry program will be invoked starting with the next slide number and the program continues as if '1' had been pressed until the tray is full.

HOW IT WORKS

As in Part1 the procedures used are described below.

PROCshow uses *PROCinstrs* to load the file and then displays it on the screen in paged mode.

PROCinstrs asks for the year and uses *PROCstate(dir$)* to check it exists. If not, it puts a message on the screen using *PROCmsge*, otherwise it asks if E or U (Europe or UK) and which tray is required. It then checks for a valid file using *PROCstate(dir$+"."+file$)*. If the file isn't valid it writes an error message as before, otherwise it loads the file and returns.

PROClist uses *PROCinstrs* to load the file and then prints it out in double-strike Pica.

PROCprint uses *PROCinstrs* to load the file and then uses *PROCbox* to print the tray labels if required. Next it asks if 'A'll, 'S'elected, or 'N'o labels are to be printed, if the latter then it returns. If all are to be printed the procedure sets flag *F%* otherwise it clears it. Then it opens the file for input and sets the printer to compressed Elite and a suitable margin. Now it uses *PROCsend* to format and print two entries, prints a line of dashes as label cutting lines and repeats the sequence, finally using *PROCsend* once more to complete the first adhesive label. This process is repeated until all data has been printed.

PROCsend uses *PROCinput* to get one text string and sets it to form the first half of each line of data, then uses it again to form the second. It looks at the P or L character at the end of the string and puts an asterisk at the appropriate end of the second line. It prints the first half of the first line followed by "|" followed by the second half then does the same for the second line. Note: if the first string of the sequence is the last in the file the second is set to a dummy blank so that the final label may be printed.

PROCinput reads a string from the file and if the flag is set returns. If the flag

is not set it prints the string on the screen and asks "Print Label (Y/N)?". If 'Y' is pressed it returns and if 'N' is pressed gets the next string. Note: in order to cope with the fact that the occurrence of the end of a file is quite unpredictable, when selective printing is employed the flag *G%* is used to note when the last string has been read.

PROCcat asks for the year and then uses OSCLI"CAT "+dir$ to display the directory catalogue. As before *PROCstate(dir$)* is used to check it is there.

PROCskip asks for the target slide number and reads each entry in turn, comparing its number with the target minus 1. When this is reached the routine returns so that the next entry may be handled normally.

PROCeof reads in and discards each entry while keeping track of the slide number until the end of the file is reached. This enables the entry adding routines to function normally.

While the program as it stands has a very specific purpose the basic idea could be adapted to catalogue many different types of collections - good luck!

```
   10 REM Program Slide Catalogue Part 2
   20 REM Version M1.0
   30 REM Author  Jim Phillips
   40 REM BEEBUG  June 1993
   50 REM Program subject to copyright
   60 :
  150 IF ans$="1" PROCnew:ELSE IF ans$="
2" PROCedit:ELSE IF ans$="3" PROCprint:E
LSE IF ans$="4" PROCshow ELSE IF ans$="5
" PROClist ELSE IF ans$="6" PROCcat
 1140 PRINT
 1150 PRINTTAB(22)"2 - Edit file"
 1160 PRINT
 1170 PRINTTAB(22)"3 - Print label"
 1180 PRINT
 1190 PRINTTAB(22)"4 - Display File"
 1200 PRINT
 1210 PRINTTAB(22)"5 - Print tray conten
ts"
 1220 PRINT
 1230 PRINTTAB(22)"6 - Show filenames"
 1260 REPEAT:ans$=GET$:UNTIL INSTR("0123
456",ans$)
 2180 DEF PROCshow
 2190 PROCinstrs
 2200 IF A%=0 ENDPROC
 2210 ch%=OPENIN(file$):VDU14
 2220 CLS:PRINTTAB(5)"Year - ";dir$;TAB(
40)"Tray No. - ";t$:PRINT
 2230 REPEAT
 2240 IF EOF#ch% GOTO2280
 2250 INPUT#ch%,line$
 2260 No$=LEFT$(line$,2):text$=MID$(line
$,10,40):append$="("+RIGHT$(line$,1)+")"
 2270 PRINTNo$"  "text$""append$:PRINT
 2280 UNTIL EOF#ch%
 2290 PRINT"Press any key to continue":R
EPEAT UNTIL GET
 2300 CLOSE#ch%:VDU15:ENDPROC
 2310 :
 2320 DEF PROCedit
 2330 PROCinstrs
 2340 IF A%=0 ENDPROC
 2350 PROCkey:ch%=OPENUP(file$):VDU14:N%
=0
 2360 CLS:PRINTTAB(5)"Year - ";dir$;TAB(
40)"Tray No. - ";t$
 2370 REPEAT
 2380 IF EOF#ch% GOTO2500
 2390 pointer%=PTR#ch%
 2400 INPUT#ch%,line$
 2410 N%=VAL(LEFT$(line$,2))
```

```
 2420 title$=MID$(line$,10,40)
 2430 shape$=RIGHT$(line$,1):PROCsetshap
e
 2440 PRINTTAB(10)"Slide - "N%;
 2450 IF shape$="P" PRINTTAB(25)"Portrai
t style" ELSE PRINTTAB(25)"Landscape sty
le"
 2460 PRINT:PRINTTAB(10)title$
 2470 PRINTTAB(15)"Is title OK ? if so t
ype Y. If not type (P)ortrait or (L)ands
cape"
 2480 REPEAT:shape$=GET$:UNTIL INSTR("Yy
PpLl",shape$)
 2490 IFINSTR("PpLl",shape$) PROCsetshap
e:PROCsetptr:PROCgetitle:PRINT#ch%,line$
 2500 CLS:UNTIL EOF#ch%
 2510 IF N%=50 GOTO 2550
 2520 CLS:PRINTTAB(10)"Do you want to ad
d more slides (Y/N)?"
 2530 REPEAT:ans$=GET$:UNTIL INSTR("YyNn
",ans$)
 2540 IF INSTR("Yy",ans$) PROCadd
 2550 CLOSE#ch%:VDU15:ENDPROC
 2560 :
 2570 DEF PROCsetptr
 2580 PTR#ch%=pointer%
 2590 ENDPROC
 2600 :
 2610 DEF PROCprint
 2620 PROCinstrs
 2630 IF A%=0 ENDPROC
 2640 PROCbox:PRINT
 2650 PRINTTAB(10)"Print (A)ll, (S)elect
ed or (N)o contents labels ?"
 2660 REPEAT:ans$=GET$:UNTIL INSTR("AaSs
Nn",ans$)
 2670 IF INSTR("Nn",ans$) OSCLI"DIR $":E
NDPROC
 2680 IF INSTR("Aa",ans$) F%=TRUE ELSE F
%=FALSE
 2690 ch%=OPENIN(file$)
 2700 CLS:VDU2,1,27,1,71,1,15,1,27,1,77,
1,27,1,108,1,32,3
 2710 REPEAT:G%=FALSE
 2720 IF EOF#ch% G%=TRUE:GOTO2810
 2730 PROCsend
 2740 VDU2:PRINTSTRING$(55,"-"):VDU3
 2750 IF G% GOTO2800
 2760 PROCsend
 2770 VDU2:PRINTSTRING$(55,"-"):VDU3
 2780 IF G% GOTO2800
 2790 PROCsend
 2800 VDU2:PRINT:VDU3
 2810 UNTIL G%
 2820 CLOSE#ch%:VDU2,1,27,1,64,3
 2830 ENDPROC
 2840 :
 2850 DEF PROCsend
 2860 PROCinput:IF G%ENDPROC
 2870 part1$=LEFT$(line$,9)+" "+MID$(lin
e$,10,16)
 2880 part3$=MID$(line$,26,24):part5$=RI
GHT$(line$,1)
 2890 PROCinput
 2900 IF G% part2$=STRING$(26," "):part4
$=STRING$(26," "):GOTO 2940
 2910 part2$=LEFT$(line$,9)+" "+MID$(lin
e$,10,16)
 2920 part4$=MID$(line$,26,24):part6$=RI
GHT$(line$,1)
 2930 IF part6$="L" OR part6$="l" part4$
="* "+part4$+" " ELSE part4$="  "+part4$
+" *"
 2940 IF part5$="L" OR part5$="l" part3$
="* "+part3$+" " ELSE part3$="  "+part3$
+" *"
 2950 VDU2:PRINTpart1$;TAB(28);"|";part2
$
 2960 PRINTpart3$;TAB(28);"|";part4$
 2970 VDU3:ENDPROC
 2980 :
 2990 DEF PROClist
 3000 PROCinstrs
 3010 IF A%=0 ENDPROC
 3020 IF LEFT$(file$,1)="U" place$="U.K.
 " ELSE place$="Europe "
```

```
 3030 CLS:ch%=OPENIN(file$):VDU 15,2,1,2
7,1,108,1,8,1,27,1,120,1,1
 3040 PRINTTAB(25)"SLIDE CATALOGUE":PRIN
T
 3050 PRINTTAB(10)place$+" ";dir$;TAB(30
)"Tray No. - ";t$:PRINT
 3060 REPEAT
 3070 IF EOF#ch% GOTO3130
 3080 INPUT#ch%,line$
 3090 part1$=LEFT$(line$,2)
 3100 part2$=MID$(line$,10,40)
 3110 form$=RIGHT$(line$,1)
 3120 PRINTpart1$;"  ";part2$;"  (";form
$;")"
 3130 UNTIL EOF#ch%
 3140 CLOSE#ch%:VDU1,12,1,27,1,120,1,0,2
7,1,64,3:ENDPROC
 3150 :
 3160 DEF PROCinstrs
 3170 PRINTTAB(20)"What year?";:INPUTTAB
(35)""dir$
 3180 PROCstate(dir$)
 3190 IF A%=0 PROCmsge("No records for t
his year !"):ENDPROC
 3200 PRINTTAB(20,2)"(E)urope or (U).K ?
";
 3210 REPEAT:pl$=GET$:UNTIL INSTR("EeUu"
,pl$):pl$=CHR$(ASCpl$ AND &5F):PRINT"  "
pl$
 3220 PRINTTAB(20,4)"Which Tray?";:INPUT
TAB(35,4)""tr$
 3230 t$=RIGHT$("00"+tr$,2):M%=VAL(tr$)
 3240 file$=pl$+t$
 3250 PROCstate(dir$+"."+file$)
 3260 IF A%=0 PROCmsge("No record of thi
s tray !"):ENDPROC
 3270 OSCLI"DIR "+dir$
 3280 ENDPROC
 3290 :
 3300 DEF PROCbox:PRINT
 3310 PRINTTAB(10)"Print tray label (Y/N
) ?"
 3320 REPEAT:ans$=GET$:UNTIL INSTR("YyNn
",ans$)
 3330 IF INSTR("Nn",ans$) ENDPROC
 3340 IFVAL(t$) MOD2=1 tab%=1 ELSEtab%=9
 3350 IF LEFT$(file$,1)="E" place$="EURO
PE" ELSE place$=" U.K."
 3360 CLS:VDU2,1,27,1,71,1,14,1,27,1,80,
1,27,1,108,1,7,1,27,1,67,1,8
 3370 PRINTTAB(tab%)place$
 3380 PRINT:VDU1,14
 3390 PRINTTAB(tab%+1)dir$
 3400 PRINT:VDU1,14
 3410 PRINTTAB(tab%+2)tr$
 3420 VDU1,12,3
 3430 CLS:PRINTTAB(10)"Reset label posit
ion if required"
 3440 ENDPROC
 3450 :
 3460 DEF PROCinput
 3470 G%=FALSE
 3480 IF EOF#ch% G%=TRUE:ENDPROC
 3490 IF F% INPUT#ch%,line$:ENDPROC
 3500 REPEAT:IF EOF#ch% G%=TRUE:GOTO3550
 3510 INPUT#ch%,line$
 3520 CLS:PRINT line$:PRINT
 3530 PRINTTAB(10)"Print label (Y/N) ?"
 3540 REPEAT:ans$=GET$:UNTIL INSTR("YyNn
",ans$)
 3550 UNTIL INSTR("Yy",ans$) OR G%
 3560 ENDPROC
 3570 :
 3580 DEF PROCcat
 3590 CLS:PRINTTAB(15)"Year ?";:INPUTTAB
(30)""dir$
 3600 PROCstate(dir$)
 3610 IF A%=0 PROCmsge("No records for t
his year !"):ENDPROC
 3620 CLS
 3630 OSCLI "CAT "+dir$
 3640 PRINT
 3650 PRINTTAB(20)"Press any key to retu
rn to menu"
 3660 REPEAT UNTIL GET
 3670 ENDPROC
```

1st course

Sound (2)

Alan Wrigley continues his look at the use of sound on the BBC micro.

NOISE CONTROL

So far we have looked at the simple use of the SOUND command to produce a square wave whose pitch, amplitude and duration can all be specified in the command. However, even without using envelopes, there are other things you can do with the basic SOUND command. Last month I mentioned that there is a channel in the sound chip provided specifically for the production of noise, and so I will begin this month by describing its use.

To recap, the SOUND command has four parameters, referred to as C, A, P and D, which specify channel number, amplitude, pitch and duration respectively. Last month I showed you how to produce simple sounds using channels 1-3. If, however, the C parameter is given as zero, then this activates the noise channel. In this case, the A and D parameters have the same meanings as before, but pitch is to a certain extent irrelevant when producing noise, and so the P parameter has a different meaning. It can take a value between 0 and 7, as follows:

0 High frequency periodic noise
1 Medium frequency periodic noise
2 Low frequency periodic noise
3 Periodic noise of frequency determined by pitch setting of Channel 1
4 High frequency white noise
5 Medium frequency white noise
6 Low frequency white noise
7 Noise of frequency determined by pitch setting of Channel 1

These descriptions require a little more explanation. The first 3 values (0-2), although described as noise, actually produce a clearly recognisable note with a rasping quality similar to some sounds produced by synthesisers; the pitch of the note is approximately middle C for the medium frequency and an octave higher or lower for the high and low frequency variations respectively.

Specifying 3 as the value of the P parameter produces a sound with similar characteristics, but allowing you to control the pitch of the note by using the current pitch setting on channel 1. In other words, whatever pitch was specified when a sound was last issued on channel 1 will be used as the basis for the periodic noise. A simple way to control the noise, therefore, without having to actually play a note on channel 1 would be to issue the following command first:

```
SOUND 1,0,P,1
```

where P is the required pitch. Since the amplitude is zero, the note will not sound. You can now produce your periodic noise sound as follows:

```
SOUND 0,A,3,D
```

giving the required values for A and D.

The remaining values for the P parameter produce a sound which is much more akin to what we think of as noise. The sound is filtered so that the noise is limited to a fairly wide frequency band, rather than covering the whole spectrum. For values 4-6, these bands cover high, medium and low frequencies respectively, while for

value 7 the band is centred on the current setting of channel 1, just as with periodic noise. For example, the following command makes a noise rather like steam being let off:

```
SOUND 0,-10,4,30
```

while the following sounds like a burst of static on a mobile radio:

```
SOUND 0,-15,6,10
```

Using value 7 for the P parameter gives you great scope to synthesise a variety of sounds. For example, the following gives a passable imitation of a rocket taking off and then disappearing into the distance:

```
FOR P%=100 TO 255
SOUND 1,0,P%,1:SOUND 0,-15,7,1:NEXT
SOUND 0,-15,7,20
FOR A%=-15 TO 0:SOUND 0,A%,7,4:NEXT
```

Noise is very useful if you want to beef up games with some interesting sounds. Simple medium or low frequency white noise is often used for gunfire, for example. Another very useful technique is to mix conventional square wave sounds from channels 1-3 with a proportion of noise. To do this, however, we have to know how to produce sounds on more than one channel at the same time, so we will consider this next.

POLYPHONY

Since the program regains control as soon it has issued a sound command, irrespective of the length of the sound, this means it is free to carry out the next instruction while the sound is playing. This allows simple polyphony to be achieved just by stringing two or more sound commands together, each one specifying a different channel. For example, the following line will sound a two-note chord (C and G):

```
SOUND 1,-15,53,10:SOUND 2,-15,81,10
```

To see a more complex example, let's return to our piece of Mozart from last month:

```
 10 FOR I%=1 TO 30:READ P%,D%
 20 IF P%=-1 A%=0 ELSE A%=-15
 30 SOUND 1,A%,P%,D%
 40 TIME=0:REPEAT UNTIL TIME>=D%*5+1
 50 NEXT
 60 DATA 81,8,69,4,73,4,81,4,69,4,53,12
 70 DATA -1,4,33,4,53,4,53,4,49,4,49,4
 80 DATA 53,4,61,4,69,4,73,16,-1,8,73,8
 90 DATA 61,4,69,4,73,4,77,4,81,16,73,8
100 DATA 69,4,81,2,69,2,53,8,61,8,53,16
```

Now add the following line:

```
35 SOUND 2,A%,P%-20,D%
```

and run it again. This time you will hear a second note in harmony, five semitones below the first. In the real world, of course, the harmony would not invariably be 5 semitones below, but this gives you an idea of what you can achieve.

If we were to halve the length of the notes on channel 2 like this:

```
35 SOUND 2,A%,P%-20,D%/2
```

the two channels would still be synchronised because of line 40, which prevents the next note being played until a specified interval has elapsed. However, if we remove line 40 and run the program with line 35 amended as above, the synchronisation is totally lost. What is needed is some way to force synchronisation of notes on different channels, even if they are of different length. Needless to say, there is a way, by using extended sound control.

EXTENDED SOUND CONTROL

Up until now we have been working on the assumption that the first parameter to the sound command is a value between 0 and 3 which indicates the sound channel to be used. However, the C parameter can also take an extended form, in which case it is supplied as a four-byte hex

number. Each byte represents a separate parameter, as follows:

Byte	Range	Function
0	0-3	Channel number
1	0-1	Flush control
2	0-3	Synchronisation control
3	0-1	Continuation control

The continuation control is only relevant if you are using envelopes, and since we have not covered these yet we will not dwell on it here. So for the moment we can supply our extended parameter in the following form:

```
SOUND &SFC,A,P,D
```

The channel number is exactly as before, while the flush control determines whether any existing note on the same channel is allowed to finish (a value for F of 0), or terminated immediately (a value of 1). To take an example, enter the following line:

```
SOUND 1,-15,100,20:SOUND 1,-15,50,10
```

The first note will continue for its full period (1 second) before the second note starts. This is because the flush control is normally set to zero (1 is the same as &01). Now alter the line to:

```
SOUND 1,-15,100,20:SOUND &11,-15,50,10
```

The flush control for the second note is now set to 1, and as a result the second note will start almost immediately, stopping the first in its tracks as it does so.

SYNCHRONISATION

The S parameter (synchronisation control) is the one which allows us to synchronise notes on different channels. The value supplied determines the number of additional channels to be synchronised. In other words, if Basic receives a SOUND command with S set to 1, it will wait until a note is ready on one other channel which also has S set to 1, and then play both notes together. If S is set to 2, it will wait until 2 other channels are ready with S also set to 2, and so on. Going back to our Mozart program (having removed line 40), we can see how this works in practice by making the following amendments:

```
30 SOUND &101,A%,P%,D%
35 SOUND &102,A%,P%-20,D%/2
```

Synchronisation will now be restored, since the two harmonising notes will be started at the same time throughout, even though they are of different length.

A NOISY MIX

Before we finish this month, let's look at some examples of mixing noise with ordinary sounds. This can be done in exactly the same way as we used to create polyphony. A very simple example is as follows:

```
SOUND 1,-15,53,10:SOUND 0,-10,5,10
```

which adds a noisy background to a straightforward middle C. This in itself is not of much use, but the technique can be expanded. The following example will cause the noise gradually to obliterate the note:

```
SOUND 1,-10,53,30
FOR A%=1 to -15 STEP -1
SOUND 0,A%,5,2:NEXT
```

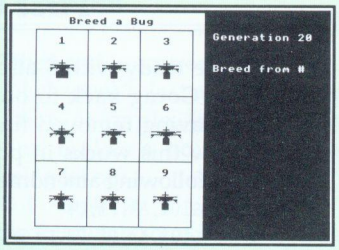
Finally, if you close your eyes and let your imagination wander, you might just be able to conjure up the space age with this:

```
SOUND 2,-15,5,35:SOUND 2,0,0,5
SOUND 2,-15,33,35:SOUND 2,0,0,5
SOUND 2,-15,53,35:SOUND 2,0,0,10
SOUND &102,-15,69,5:SOUND &103,-15,53,5
SOUND 2,0,0,2
SOUND &102,-15,65,50:SOUND &103,-15,37,50
FOR P%=100 TO 255:SOUND 1,0,P%,1
SOUND 0,-15,7,1:NEXT
FOR A%=-15 TO 0:SOUND 0,A%,7,4:NEXT
```

Next month I will describe how to use the ENVELOPE command to give even more control over the range of sounds you can produce.

Best of BEEBUG

Applications I Disc

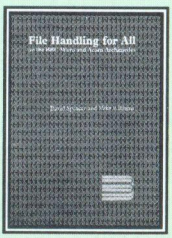


BUSINESS GRAPHICS - for producing graphs, charts and diagrams
VIDEO CATALOGUER - catalogue and print labels for your video cassettes
PHONE BOOK - an on-screen telephone book which can be easily edited and updated
PERSONALISED LETTER-HEADINGS - design a stylish logo for your letter heads
APPOINTMENTS DIARY - a computerised appointments diary
MAPPING THE BRITISH ISLES - draw a map of the British Isles at any size
SELECTIVE BREEDING - a superb graphical display of selective breeding of insects
THE EARTH FROM SPACE - draw a picture of the Earth as seen from any point in space
PERSONALISED ADDRESS BOOK - on-screen address and phone book
PAGE DESIGNER - a page-making package for Epson compatible printers
WORLD BY NIGHT AND DAY - a display of the world showing night and day for any time and date of the year

File Handling for All

on the BBC Micro and Acorn Archimedes

by David Spencer and Mike Williams



Computers are often used for file handling applications yet this is a subject which computer users find difficult when it comes to developing their own programs. *File Handling for All* aims to change that by providing an extensive and comprehensive introduction to the writing of file handling programs with particular reference to Basic.

File Handling for All, written by highly experienced authors and programmers David Spencer and Mike Williams, offers 144 pages of text supported by many useful program listings. It is aimed at Basic programmers, beginners and advanced users, and anybody interested in File Handling and Databases on the Beeb and the Arc. However, all the file handling concepts discussed are relevant to most computer systems, making this a suitable introduction to file handling for all.

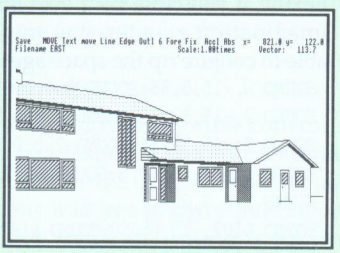
The book starts with an introduction to the basic principles of file handling, and in the following chapters develops an in-depth look at the handling of different types of files e.g. serial files, indexed files, direct access files, and searching and sorting. A separate chapter is devoted to hierarchical and relational database design, and the book concludes with a chapter of practical advice on how best to develop file handling programs.

The topics covered by the book include:

Card Index Files, Serial Files, File Headers, Disc and Record Buffering, Using Pointers, Indexing Files, Searching Techniques, Hashing Functions, Sorting Methods, Testing and Debugging, Networking Conflicts, File System Calls

The associated disc contains complete working programs based on the routines described in the book and a copy of Filer, a full-feature Database program originally published in BEEBUG magazine.

ASTAAD



Enhanced ASTAAD CAD program for the Master, offering the following features:

- full mouse and joystick control
- built-in printer dump
- speed improvement
- STEAMS image manipulator
- Keystrips for ASTAAD and STEAMS
- Comprehensive user guide
- Sample picture files

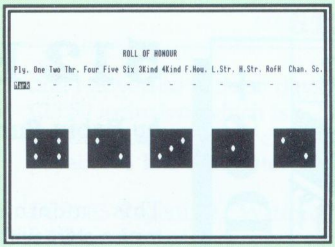
	Stock Code	Price		Stock Code	Price
ASTAAD (80 track DFS)	1407a	£5.95	**ASTAAD** (3.5" ADFS)	1408a	£5.95
Applications II (80 track DFS)	1411a	£4.00	**Applications II** (3.5" ADFS)	1412a	£4.00
Applications I Disc (40/80T DFS)	1404a	£4.00	**Applications I Disc** (3.5" ADFS)	1409a	£4.00
General Utilities Disc (40/80T DFS)	1405a	£4.00	**General Utilities Disc** (3.5" ADFS)	1413a	£4.00
Arcade Games (40/80 track DFS)	PAG1a	£5.95	**Arcade Games** (3.5" ADFS)	PAG2a	£5.95
Board Games (40/80 track DFS)	PBG1a	£5.95	**Board Games** (3.5" ADFS)	PBG2a	£5.95

All prices include VAT where appropriate. For p&p see Membership page.

RISC Developments Ltd, 117 Hatfield Road, St Albans, Herts AL1 4JS.

Best of BEEBUG

Board Games



Solitaire - an elegant implementation of this ancient and fascinating one-player game, and a complete solution for those who are unable to find it for themselves.

Roll of Honour - Score as many points as possible by throwing the five dice in this on-screen version of 'Yahtze'.

Patience - a very addictive version of one of the oldest and most popular games of Patience.

Elevenses - another popular version of Patience - lay down cards on the table in three by three grid and start turning them over until they add up to eleven.

Cribbage - an authentic implementation of this very traditional card game for two, where the object is to score points for various combinations and sequences of cards.

Twiddle - a close relative of .Sam Lloyd's sliding block puzzle and Rubik's cube, where you have to move numbers round a grid to match a pattern.

Chinese Chequers - a traditional board game for two players, where the object is to move your counters, following a pattern, and occupy the opponent's field.

Aces High - another addictive game of Patience, where the object is to remove the cards from the table and finish with the aces at the head of each column.

Applications II Disc

Crossword Editor - for designing, editing and solving crosswords

Monthly Desk Diary - a month-to-view calendar which can also be printed

3D Landscapes - generates three dimensional landscapes

Real Time Clock - a real time digital alarm clock displayed on the screen

Running Four Temperatures - calibrates and plots up to four temperatures

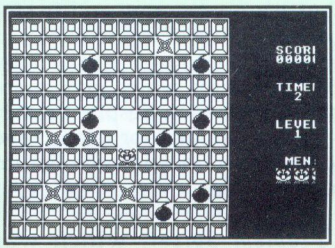
Julia Sets - fascinating extensions of the Mandelbrot set

Foreign Language Tester - foreign character definer and language tester

Share Investor - assists decision making when buying and selling shares

Label Processor - for designing and printing labels on Epson compatible printers

Arcade Games



George and the Dragon - Rescue 'Hideous Hilda' from the flames of the dragon, but beware the flying arrows and the moving holes on the floor.

Ebony Castle - You, the leader of a secret band, have been captured and thrown in the dungeons of the infamous Ebony Castle. Can you escape back to the countryside, fighting off the deadly spiders on the way and collecting the keys necessary to unlock the coloured doors?

Knight Quest - You are a Knight on a quest to find the lost crown, hidden deep in the ruins of a weird castle inhabited by dangerous monsters and protected by a greedy guardian.

Pitfall Pete - Collect all the diamonds on the screen, but try not to trap yourself when you dislodge the many boulders on your way.

Builder Bob - Bob is trapped on the bottom of a building that's being demolished. Can you help him build his way out?

Minefield - Find your way through this grid and try to defuse the mines before they explode, but beware the monsters which increasingly hinder your progress.

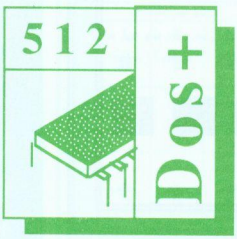
Manic Mechanic - Try to collect all the spanners and reach the broken-down generator, before the factory freezes up.

Quad - You will have hours of entertainment trying to get all these different shapes to fit.

	Stock Code	Price
File Handling for All Book	BK02b	£ 9.95
File Handling for All Disc (40/80T DFS)	BK05a	£ 4.75
Joint Offer book and disc (40/80T DFS)	BK04b	**£ 11.95**
Magscan (40 DFS)	0005a	£ 9.95
Magscan (80T DFS)	0006a	£ 9.95
Magscan (3.5" ADFS)	1457a	£ 9.95

	Stock Code	Price
File Handling for All Disc (3.5" ADFS)	BK07a	£ 4.75
Joint Offer book and disc (3.5" ADFS)	BK06b	**£ 11.95**
Magscan Upgrade (40 DFS)	0011a	£ 4.75
Magscan Upgrade (80T DFS)	0010a	£ 4.75
Magscan Upgrade (3.5" ADFS)	1458a	£ 4.75

All prices include VAT where appropriate. For p&p see Membership page.



512 Forum

by Robin Burton

This month's topics start with a couple of points raised regularly in readers letters. It also seems about time to remind you again, especially newer readers of 512 Forum, that if there's any particular aspect of the 512, DOS or software that you'd like me to deal with drop me a line via BEEBUG or Essential Software. I'll do what I can to oblige.

REGULAR QUERIES

Obviously as time passes more and more long-term users of the 512 are likely to upgrade to later and faster systems, probably a PC or an Archimedes.

It's also probable that the majority of those users aren't likely to 'moth-ball' or scrap their old system; they sell it to someone else who becomes a new user. I've mentioned this process before of course, but it's continuous and is still affecting the contents of my mail. As an example there are a couple of specially written 512 programs which generate a steady trickle of queries, despite the fact that they've been around for a long time and have been mentioned in the Forum on various occasions.

One is *Problem Solver*, which was written and sold by Shibumi Soft. It aimed to alleviate some of the 512's hardware deficiencies in software, and to an extent it succeeded. It was, though, much more likely to be of help with a game than with a serious application and I've made no secret of my opinion that it was drastically over-priced at £30.00. In fact I've never recommended anyone to buy it unless it was to help a specific program someone needed and which appeared on Shibumi's list.

Despite its laudable aims, when it was first released the program had a number of serious faults, though these were eventually corrected after a year or so. However, in the meantime quite a number of 512 users became frustrated by the very slow, or complete absence of response to letters. Shibumi Soft was, incidentally, located in Portugal and hence was out of reach and couldn't be pressured into action.

Later Shibumi set up a mailing address in Northampton. While this was intended to add to their outward credibility it did absolutely nothing for their service, which if anything grew even worse. Replies to orders regularly took five weeks or more even if you caught them on a good day, while queries or complaints took even longer, assuming you ever received a reply at all. Things didn't improve no matter who (me included) tried to urge them to get their act together.

Clear warnings about Shibumi Soft's dismal performance were given several times in the Forum but I make no apology for repeating them again because even worse was to follow. New 512 users might be tempted to write to Shibumi if they come across their address in a back issue. That's merely a waste of time, but what matters is that someone might send an order with money. DO NOT!

About a year and a half ago, for reasons known only to themselves, Shibumi Soft simply stopped supplying their program. That's their prerogative of course. The unacceptable part was that they didn't tell anyone and didn't have the courtesy to reply to any letters thereafter. Worse yet, they didn't return cheques and in one

case known to me plus a few more I've heard about they cashed cheques but didn't supply the goods.

I get asked from time to time if I know of a source for this program. The official answer is no, but in view of Shibumi's appalling record of rudeness, bad service and finally apparently even theft, I've decided to take an unprecedented step. However, before I continue I must state that this offer has nothing whatsoever to do with BEEBUG; I alone am responsible. If Shibumi Soft don't like it they know where to find me.

If anyone wants a copy of Problem Solver send a cheque for £2.00 payable to me at the address below. The charge is to cover costs only - I am not selling the program of course. The documentation will be a text file on the disc, though by not getting a copy of the printed 'manual' (I use the term very loosely) you're not missing a thing. However, read on before finally deciding on this offer.

TCS MOUSE DRIVER

Another program I'm asked about regularly is the Microsoft-compatible 512 mouse driver written by Cliff Bowman of Tull Computer Services. First though, let me make it quite clear that there's no suggestion of sub-standard service in this case.

Quite a few people have said they've been unable to locate TCS and asked if I could help. Yes I can. I phoned Cliff on the number he's had for the past three years and lo and behold he answered. Not too difficult! Frankly I don't know why anyone has had trouble, although perhaps some were misled because Tull moved after the 512 Technical Guide was published. The address and phone number shown there are therefore out of date.

Tull will happily still supply their mouse driver, although for obvious reasons they don't advertise it these days. The price is £23.50 inclusive, which is a 20% reduction on the original price. Send a cheque payable to Tull Computer Services to:

TCS, 65 Regent St., Watford, Herts WD2 5AT and you'll receive an 800K disc containing the software plus a comprehensive manual.

MORE 512 FREEBIES

You might remember the 512 information disc produced by David Harper and mentioned in the Forum in BEEBUG Vol.11 No.8, plus the series of four articles on GEM (Vol.10 Nos.7 to 10) for which David supplied the information. Well there's more good news, he's been at it again!

David sent me a couple of 512 programs that he's produced recently. These were originally written for his own use, but as he says, once the basics are in place it's relatively straightforward to continue to add extra features. This is something I can confirm: the usual problem is deciding when to stop!

In the event David continued with these programs until he felt that they might well be useful to others. What's even better is that, as he did with his 512 information disc, he's released both programs into the public domain. Yes, I realise I haven't yet told you what they do, I'm coming to that.

The first program is a PC compatibility enhancer for the 512 (called PCCE for short) which has broadly similar objectives to Problem Solver's.

The reason for this program being written was the fact that the editor in Quick C wouldn't work in David's 512, although both the compiler and linker did. The problem wasn't memory since David has an expanded memory machine and as the other programs did work David thought it likely that both

the cause and a cure for the editor problem could be found. As it turned out, while he was doing that he addressed a number of other 512 (in)compatibility problems too.

Without getting too technical I'll give you a brief outline of what PCCE does, but if any of the following doesn't mean much to you don't worry, just take it from me that PCCE will help a number of programs to run in the 512 that otherwise either do nothing at all, or hang the system.

One of the standard DOS calls in a PC is interrupt 9, which occurs every time a key is pressed. This can be regarded very much like events in the BBC micro. The interrupt doesn't actually do anything itself, but it can be used to detect that something has happened. Naturally, a good many DOS TSR programs (often called pop-ups) hang on INT 9 so that they are informed when a key has been pressed: they then check which key was pressed and activate if it was their assigned key (usually a combination).

The problem for 512 users is that Acorn didn't implement INT 9 in DOS Plus, and that's why many TSRs don't work: they simply don't know that keys have been pressed. These programs load OK, but that's the last you see of them, apart from their memory consumption.

PCCE corrects this omission, along with a couple of associated problems such as the lack of an 'in-DOS' flag and a meagre amount of system stack space without which various TSRs that otherwise would work will crash the system. PCCE also implements a better method of breaking out of loops caused by CGA snow-avoidance techniques, this particular 512 DOS Plus problem preventing the Galaxy Lite shareware word processor from working, for example.

Other additions are making the cursor behave as it should in a PC, avoiding the text (BBC hardware) cursor lying idly somewhere on the screen when a program doesn't use it. This is a common nuisance in menu driven programs, especially those that use a mouse. Other programs expect to change the shape of the cursor, but they won't in the 512 without PCCE.

Another, more topical item, is that Pkware have very recently released version 2.04G of their excellent data compression and disc archiving shareware. I have mentioned it before, but another Forum article is long overdue on this software. Watch this space. However, I was disappointed to find that unlike previous versions, the new one wouldn't run at all in the 512, not even as far as displaying the help screen. I happened to tell David about this yesterday and he'd just received the new version too. Today he rang me to say he'd extended PCCE to fix the problem.

So far the list of programs that PCCE allows to run in the 512 is based on David's own tests. It includes Galaxy Lite, PKZIP/UNZIP v.2.04G, Microsoft Quick C and QBasic, As-Easy-As v.5 and a number of games which otherwise hang the system. Some of these might need an expanded 512, but as David doesn't have a 512K machine he can't check this. Anyone who gets PCCE and does find out, or can add other programs to the list should let me know and I'll include the information in future issues.

There is more in PCCE, but of course explanations are bound to be technical and so aren't likely to be interest most users. There's simply not enough space here to explain everything anyway and we now need to move on to David's other new program, a Microsoft-compatible text mouse driver.

A mouse driver doesn't need much explanation but it is appropriate to observe that it's another PC standard Acorn failed to provide. This omission is amazing when you recall that the mouse itself was included as part of the 512 system, but it becomes truly stunning when you consider the fact that Acorn did go to the trouble of writing two non-standard mouse drivers for the 512's bundled GEM. After all, in a PC, GEM uses a standard mouse driver! I can only think that someone must have had a very warped sense of humour.

David calls his program a *Text Screen Mouse Driver*, because it operates in 80 column text mode only. The inspiration for this one was the pull-down menus in the Quick C editor, once it was working, hence the 80 column 'limitation'. That said, who wants to work with 40 column text anyway? I know of no programs that do so. Graphics in the 512 is a totally different matter, which is why even the TCS mouse driver mentioned above has limitations in this area.

Many of you know I'm not a mouse fan, and so I'm no authority on this subject, but I'm quite prepared to accept it when David tells me his mouse driver works correctly with some programs that Tull's driver doesn't, the pointer movement is smoother, so the mouse is easier to use, and of course best of all, the program is free.

AND FINALLY

And finally, here is yet another 512 special! Programs that allow you to edit the environment variables in PCs aren't new, are very useful but sadly don't work in the 512. Things are in different places in the 512 and this is another area where DOS Plus and MS-DOS don't work in quite the same way.

Richard Poynter recently sent me a program which does this job in the 512. The program is very neat, with a nice front end and it reinstates the screen contents when you leave it after an edit. Within the program you can directly edit any of the 512's environment variables while the system is running.

Why? Well for example when you set a default PATH, normally in your AUTOEXEC.BAT, you can't change any of it it without typing the entire line again. A temporary addition to PATH, to test a new program say, then putting it back afterwards is very tedious, especially if your PATH string is as long as mine. The screenshot shows the editor in action on my path, plus the editing options. With 'ENVEDIT' you can simply add or delete only the part of the entry you want to change instead. This is much quicker, easier and far less likely to result in an error.

Richard retains the copyright of his program so it's neither public domain nor shareware, but he is happy for me to supply it to 512 Forum readers.

HOW TO GET THEM

You can obtain a copy of David Harper's programs from BBC PD (see the PD column in this issue as well) for Problem Solver. Alternatively, I'll provide a copy of the two plus Richard's environment editor for the princely sum of £2.00 inclusive.

Note, however, that I am not prepared to mix legitimate PD software or freeware with (strictly speaking illicit, though Shibumi Soft certainly deserve the treatment) commercial software on the same disc. If you want Problem Solver and the other programs these are two separate requests, though you can knock 38p off the combined price for the reduction in p&p.

For any of the above programs (except the TCS mouse driver) write to me directly: PO Box 5, Groby, Leicestershire LE6 0ZB.

Next month we'll take a look at PC magazine cover discs plus Pkware's software.

Machine Code Corner

Mr T dishes more dirt on HAZEL, LYNNE and ANDY.

Greetings, fellow mortals. Obviously nobody has been silly enough to code up a routine to print the Nine Billion Names Of God. Or maybe they have, but it's a bit slow. We shall never know.

Only kidding. Anyhow, last month we did FRED, JIM and SHEILA, so now for the three parvenus, HAZEL, LYNNE and ANDY. If you have a B+ you've got LYNNE, but only Masters have all three. So far as I can see, the Acorn Manual for the Master uses none of these names, though it often uses the other three, but all the other books I've got refer to HAZEL, LYNNE and ANDY.

Every computer has to have *screen memory*, an area of RAM into which information is written by the CPU and from which the video circuitry reads it and displays it on the screen. The older Beebs have a severe memory problem with only 32K of RAM being shared by the screen memory, the program space, the MOS and the filing systems' workspace, plus the workspace any other assorted ROMs may grab. The concept of the various display modes was a brilliant part-solution, in that as well as the Teletext mode you can select various versions of the 20, 40 and 80-column layouts, using less memory with the two-colour and non-graphics options where these will do for your purpose. Still, screen and program share the same stretch of RAM from &3000 to &7FFF, and the more the screen uses, the less there is for the program. A 20K screen mode leaves precious little for Basic when PAGE is up around &1200.

When designing the newer machines, Acorn couldn't simply stick in more RAM just anywhere without losing compatibility with all the software written for the old machine. That's why LYNNE was born; she's *shadow RAM* - a 20K chunk of an extra 32K RAM chip - and is mapped in to the same addresses as the 'old' screen at &3000 to &7FFF. Not only can new software benefit from shadow screen modes, but older software running on the new machine can be made to use them too. A program can now use all the main memory up to &7FFF for code and data; the video chip reads screen memory in shadow RAM. Mind you, not all BBC B software will gain from running in a shadow mode; it depends on whether it needs a lot of storage space and whether it can detect and use any extra provided. By keeping LYNNE adresses identical to those used by earlier Beebs, Acorn have made it very easy for software to run both methods; it's the hardware that makes Mr T's eyes boggle.

Access to LYNNE is controlled by ACCON at &FE34: set bit 0 and any read instructions which come from the video controller - a 6845 CRTC chip - are directed to her; reset it and you're back to the BBC B situation. What about write access, though? Before the video controller can display bit maps from LYNNE, they've got to be written to her by the 6502. If the hardware simply redirects all writes to LYNNE, then as soon as, say, the top of a Basic program goes beyond &3000, its variables and any other data which it writes are going to end up in the wrong chip. Furthermore, there are times when the screen routines need to read from, as well as write to, screen memory. Ouch. Somehow, shadow screen memory and program memory have got to be kept totally separate.

Now the code in ROM which contains the *screen drivers* - the routines for all the display operations which result in a

write to screen memory - is mapped in from &C000 to &DFFF. Cunning old Auntie Acorn has arranged that if you set bit 1 of ACCON, the execution of any opcode located in the screen driver region will result in a read to or a write from LYNNE, so long as (a) it's not an opcode fetch (i.e. not an instruction) and (b) the address for reading or writing is between &C000 and &DFFF - of course. Any instruction coming from anywhere other than the screen drivers causes a read or write in main memory. Putting it at its simplest: when the screen drivers are controlling the 6502, the 6502 talks and listens to LYNNE, otherwise it ignores her. If bit 0 as well as bit 1 of ACCON is set, the video controller will read from LYNNE as well: shadow memory has been achieved. Mr T frankly admits that he can't begin to understand how it's done, but it is.

Geniuses such as yourselves will have already spotted the fact that with shadow RAM there is no point at all in selecting a memory saving mode and that some of the old ones, such as mode 2, are needed only to maintain compatibility with old software, since with a shadow screen, any mode which uses less than 20K simply leaves spare space lying around in LYNNE. In fact, in mode 135 it works out to just over 19K - not far off twice the total free space in a BBC B with DFS and a few other ROMs when running in mode 5. Does the machine use it? Only for transfers between filing systems, so far as I know. Can we use it? From machine-code, yes, easily. Auntie Acorn has provided that bit 2 of ACCON, if set, pages LYNNE in unconditionally over the last 20K of RAM. Easy enough, then, to store things in LYNNE, up to 19K of data under shadow mode 7, but don't forget that the top of her is screen memory, else you get some right foul-ups. In fact, ACCON in the hands of a careless programmer is like a chainsaw handed to a chimp. The faint-hearted may use OSBYTES &6C, &70 and &71.

You may also have spotted that by switching from main to shadow modes and back you can flip between screens, updating one while the other is being displayed. Incredible speed can be obtained this way, but it's hardly needed except for arcade games.

We said earlier that LYNNE is only part of the extra 32K RAM chip which together with the original 32K and the 64K of sideways RAM gives the Master its eponymous 128K. So what happens to the other 12K? Acorn have used it to ensure that PAGE never goes above &E00, unless a ROM written for the older machine is active and claims workspace; thus program space is extended at the bottom as well as at the top.

This is achieved as follows: on the Master an 8K segment of the spare 12K is mapped in from &C000 to &DFFF; you will note that this means that it is paged in over the screen drivers referred to earlier, but since it's never used during VDU routines, that's fine. This section is sometimes referred to as HAZEL, but that name is used even less consistently than the others. It's also sometimes known as *hidden RAM*, which is misleading, because it's almost always paged in; the MOS selects the screen drivers when needed, by clearing bit 3 of ACCON temporarily. Programmers can rely on finding RAM in this area whenever they want it. The stretch of HAZEL from &C000 to &DCFF is called *private workspace* and is for the filing systems and other ROMs, which get a chance to claim a chunk of it on reset. This frees space at the bottom end, as we observed earlier. On the B+ the whole 12K is officially called *Private RAM* and runs from &8000 to &AFFF; it is also controlled by ACCON at &FE34.

RAM in HAZEL from &DD00 to &DEFF is called the *transient utility workspace;* the machine uses it only for *MOVE

operations, so it's pretty secure. The manual says it's the best place for small machine-code applications. Mr T makes a lot of use of it, but has seldom, if ever, seen it used in programs other than his own. Page &DC is the 'CLI buffer'; the MOS uses it for processing star-commands, copying them to &DC00 and doing:

```
LDX #(low-byte of address of first
meaningful character)
LDY #&DC
JSR oscli
```

Now, star commands are not very long, therefore the majority of page &DC is fairly secure from corruption. You can safely use all of it for transitory purposes, as the star commands aren't needed once they've been processed. A memory editor will show that the various ROMs waste further huge chunks of HAZEL, but it's too broken-up for the programmer to make much use of it.

The final 4K of the spare 12K is ANDY. He's mapped from &8000 to &8FFF and you select him by setting the top bit of ROMSEL, not ACCON, because he's in the first quarter of the sideways area. On older machines bit 7 of ROMSEL does nothing. As you'd expect in view of his location, ANDY is only paged in temporarily when needed, else the first 4K of the currently active ROM would vanish. When he is selected, the bottom three quarters of the current ROM remain in place, unless you also deliberately switch ROMs when writing to ROMSEL. The first 1K of ANDY contains the soft key strings and the last bit contains a RAM copy of the font, which frees space at the bottom of main RAM, pages &0B and &0C. Instead of using this bonus to lower PAGE to &0C, Auntie Acorn has reserved it for ECONET. Mr T reckons that was a bad decision, albeit taken in the interests of compatibility, but you can always thumb your nose and stick code or data from &0B00 to &0CFF. The middle of ANDY is used by VDU routines.

Mr T's soft key utility ROM *Softie* (published elsewhere) and his Fontz Rom (BEEBUG Vol.10 No.1) both rely on the fact that any ROM can switch ANDY in and out, provided only that the code which does so is outside ANDY's area, i.e. from &9000 onwards. There's none of this fancy splitting as with the list of names or with CPU writes to LYNNE - either ANDY is in or he's out. If the ROM is small and doesn't get to &9000, you will have to drop out of the assembler by writing a closing bracket, reset P% to &9000 and go back into the assembler again to page in ANDY by a deft backhand flip of ROMSEL bit 7:

```
LDA &F4:ORA #80:STA romsel \ or there's
TSB if you prefer it.
```

To deselect ANDY, the code is, of course:

```
LDA &F4:STA romsel
```

Purists might say that the first bit should end with :STA &F4, in which case the second snippet must have :AND #&0F: in the middle. Mr T is unsure of the gospel on this point - it's just possible, I spoze, that a sneaky interrupt from somewhere might creep up and interrogate bit 7 of &F4 in the middle of your chat with ANDY.

Finally, what is the origin of all these names? Mr T has no firm information, and although it's not hard to guess the general idea, it would be most interesting to have the details. If you know, please write in; we'll call it this month's competition: prize, one 'I'M A SWOT' badge. Next month we explore SHARON, GAVIN, LEE, TRACY, KYLIE and KEVIN. JASON is a bad lad and can get lost for all Mr T cares. Bye for now, frog-fanciers.

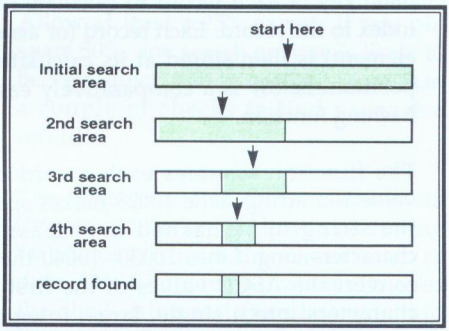
Searching

by David Fell

In the last two Workshops I went into some detail about how to sort lists into order. Sorting is often part of a file management system and, this month, we'll see another aspect, how to search for a particular item in a whole set of data. Although these examples use arrays, the techniques can equally well be applied to files of records on disc.

When we search a list, we look for the entry with a particular value - the *Search Key*. We could search by looking at every record in turn until we find the correct one, but that might take ages. A better way is a *Binary Search*, which relies on the list already being sorted. It is then rather like using a dictionary. You start in the middle and narrow your search to one half. Then find the correct quarter, and so on until you hit the word you are looking for.

```
BINARY SEARCH
 10000 DEF FNbinsrch(st%
,fin%,match)
 10010 LOCAL mid%,test
 10020 mid%=(st%+fin%)DIV 2
 10030 test=array(mid%)
 10040 REPEAT
 10050   IF test>match THEN
         fin%=mid%
 10060   IF test<match THEN
         st%=mid%
 10070   mid%=(st%+fin%)DIV 2
 10080   IF test<>match THEN
         test=array(mid%)
 10090   UNTIL test=match OR
         st%>=(fin%-2)
 10100 IF test<>match THEN mid%=-1
 10110 IF test<>match AND
       array(fin%)=match THEN
       mid%=fin%
 10120 =mid%
```



Example of binary search

The job of FNbinsrch, given the start and finish indices and the search key, is to return the index of the element in *array()* which holds *match*. If it does not find it, it returns the value -1. The core of the function is a REPEAT-UNTIL loop which halves the gap between the indices until it either finds the right value, or the 2 indices are adjacent. If the latter, it checks whether the top index holds the search key.

In your programs, you should replace every reference to *array()* with the name of the actual array used, already sorted in ascending order. The array could contain numeric or string data sorted using any of the techniques described in previous Workshops. If you are using an array of sorted pointers as I described last month, then you will need references such as:

```
array$(ptr%(mid%))
```

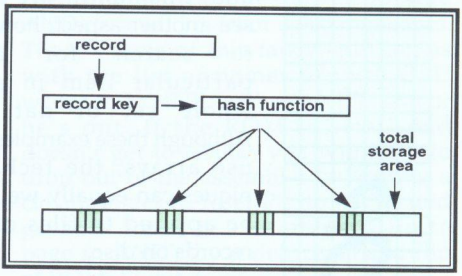
To give an idea of its speed, a binary search of a 1000-element list will find any value with no more than 9 tries.

That's one way to do it, if the data is already sorted. Another way needs no previous sorting but, since it tends to waste space, is best used for disc files. A *hashing* technique uses the value of the main key of each record to calculate the index to the record. Each record (or array element) is then stored at its calculated position. Below is a comparatively easy hashing function.

The function assumes each record or value is a string. Line 10020 makes sure the string to be hashed is at least 5 characters long. Lines 10030- 10060 then convert the ASCII values of the first 5 characters into a single, large, integer. Line 10070 uses this 'nonsense' number to force the random number system into a new sequence. Line 10080 skips the first number generated, while the function returns (line 10090) a random number in the range *1-tsize%*, where *tsize%* is the total size of the storage (array).

HASH NUMBER GENERATOR

```
10000 DEF FNhash(strg$)
10010 LOCAL hash%,i%
10020 strg$=LEFT$(strg$+"      ",5)
10030 hash%=0
10040 FOR i%=1 TO 5
10050   hash%=hash%*10+
        ASC(MID$(strg$,i%,1))
10060   NEXT
10070 hash%=RND(-hash%)
10080 hash%=RND(tsize%)
10090 =RND(tsize%)
```



Searching by hashing

Using random numbers makes use of the fact that the generator can be reset with a RND(-n) command. It then produces a new set of totally repeatable pseudo-random numbers, whose sequence is controlled only by the value of *n*. In this case, the same string always gives the same index. Once a string has been stored at its calculated position, it can always be found again by carrying out the same hashing calculation.

Be warned - even the best hashing algorithm will produce duplicate codes for totally different strings. We must be ready for this when both storing and recovering data.

To store data, hash the key and look at the corresponding location. If it is empty, save the data. If it is occupied, search forward for the first empty location or until the array (or file) is all filled. The latter should never happen, but you

never know... Otherwise, put the data in the first spare position after the hash index - if you do reach the end of the array (or file) in searching from a mid-point, go back to the very start.

Finding a record is similar. The hash code shows where to start looking - search until you find either the record or a blank, or you have checked everything. Finding a blank means that the key does not exist in the file.

```
HASH HANDLERS
11000 DEF PROChashstore(strg$)
11010 LOCAL iters%,ptr%
11020 ptr%=FNhash(strg$)
11030 iters%=0
11040 REPEAT
11050   IF array$(ptr%)<>"" AND
        array$(ptr%)<>strg$ THEN
        ptr%=ptr%+1
11060   IF ptr%>tsize% THEN ptr%=1
11070   iters%=iters%+1
11080   UNTIL array$(ptr%)="" OR
        array$(ptr%)=strg$ OR
        iters%=tsize%+1
11090 IF iters%<=tsize% THEN
  array$(ptr%)=strg$
11100 ENDPROC
11990 :
12000 DEF FNhashfind(strg$)
12010 LOCAL iters%,ptr%
12020 ptr%=FNhash(strg$)
12030 iters%=0
12040 REPEAT
12050   IF array$(ptr%)<>strg$ AND
        array$(ptr%)<>"" THEN
        ptr%=ptr%+1
12060   IF ptr%>tsize% THEN ptr%=1
12070   iters%=iters%+1
12080   UNTIL array$(ptr%)=strg$ OR
        iters%=tsize%+1 OR
        array$(ptr%)=""
12090 IF iters%>tsize% OR
        array$(ptr%)="" THEN ptr%=-1
12100 =ptr%
```

PROChashstore stores *strg$* in the previously set up *array$()*, with line 11060 providing the index wrap-around during the search. If it finds the key field already in the file, it overwrites it.

FNhashfind looks for *strg$* in *array$()*, and returns its index. If the search fails, it returns '-1'. In both routines, *iters%* counts the checks, to detect a full circle, and *tsize%* holds the maximum size of the storage.

Hashing is a very powerful random-access technique, although sorting can be quite tedious. To use it safely, the storage MUST have space for more than the maximum number of records expected. Allow at least a 25% excess; if you can spare 50%, any search or store is likely to be very quick, rarely needing more than a couple of checks to find or store a record.

The magazine disc contains complete demonstrations of both searching techniques using 4-character strings as data. The first demonstration also uses the Shell string sort from last month to presort the data for the binary search.

Points Arising. . . Points Arising. . .

BARTEM2
(BEEBUG Vol.12 No.1)

The updates to this program given in Postbag were afflicted by the inadvertent inclusion of extra brackets. In lines 1450 and 1480 remove the empty () following 'STR$(Y'. The updated program on the magazine disc was unaffected.

Gravity and Orbits (Part 4)

This month Cliff Blake plays asteroids.

A large planet may be taken as spherical with its mass acting through its physical centre, but small asteroids are often of irregular shape and may have eccentric mass. Imagine that you are orbiting round the lower half of a giant pear, all is going well, and the altimeter is steady; suddenly you are faced with the tapered section at the top. Thus surveying the surface of a small body has its own dangers.

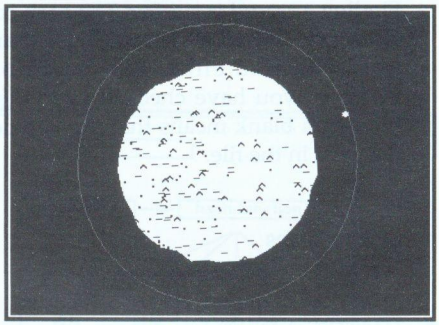
THE PROGRAM

In this scenario, you are trying to survey a small moon which appears to be round, but has heavy nodes of metal deposited away from the centre. Because you are so close there are local variations, so you can't even assume an effective common point of action. Thinking in terms of a gravity pit, it is as though the funnel has six vertical undulations over which the vehicle must travel.

```
NODAL MOON

A scout ship is surveying a small
moon. The moon's mass is irregularly
distributed with denser nodes,
so that the gravitational pull
varies over the surface.
Try maintaining the ship below the
maximum survey altitude, marked by a
circle, without crashing.

Gently holding down the < key will
fire the retro unit to slow the ship.

Gently holding down the > key will
fire the drive to accelerate the ship.

Press any key to start.

Press R to clear the screen & Repeat.

Press Q to Quit.
```

Your missions

Each time the program is run from the start, a new set of random nodes is set up. Pressing key 'R' to repeat maintains the current nodes so that a learning process can take place. If you crash, you'll probably bounce across the moon, before being hurled into space, but press key 'R' to try again.



The star is you

To reduce height, it is necessary to press the less-than '<' key to slow down on the opposite side of the moon. Similarly to gain height, accelerate with the greater-than '>' key on the opposite side of the moon. Try to complete three orbits within the height circle.

Next time we'll consider an approach to programming these simulations avoiding those nasty trig. functions, and we'll plot a gravity map.

```
  10 REM Program NODAL
  20 REM Version B2.0
  30 REM Author  Cliff Blake
  40 REM BEEBUG  June 1993
  50 REM Program subject to copyright
  60 :
 100 DIM Xn(6),Yn(6),Mn(6)
 110 ENVELOPE1,8,1,-1,1,1,1,1,121,-10,-
5,-2,120,120
 120 MODE 7:*FX11
 130 PROCinfo:g%=GET
 140 MODE1:VDU5
 150 PROCnodes
 160 REPEAT
```

```
  170 quit%=FALSE
  180 CLS:PROCmoon:PROCspaceship:PROCcir
cle
  190 REPEAT
  200 rerun%=FALSE
  210 PROCmove:PROCgravity
  220 PROCthrust:PROCflags
  230 UNTIL rerun%
  240 UNTIL quit%
  250 VDU4:*FX12
  260 CLS:*FX21
  270 END
  280 :
 1000 DEF PROCmove
 1010 MOVE Xs-16,Ys+16:PRINT"*"
 1020 Xs=Xs+Xv:Ys=Ys+Yv
 1030 IF POINT(Xs,Ys)=2 THEN MOVE Xs-16,
Ys+16:PRINT"*":SOUND0,1,0,5
 1040 MOVE Xs-16,Ys+16:PRINT"*"
 1050 ENDPROC
 1060 :
 1070 DEF PROCaltitude
 1080 Xd=Xn(node%)-Xs:Yd=Yn(node%)-Ys
 1090 Rds=Xd*Xd+Yd*Yd:Rd=SQR(Rds)
 1100 ENDPROC
 1110 :
 1120 DEF PROCgravity
 1130 Xgt=0:Ygt=0
 1140 FOR node%=0 TO 6
 1150 PROCaltitude
 1160 Xg=800*Mn(node%)*Xd/Rd/Rds:Yg=800*
Mn(node%)*Yd/Rd/Rds
 1170 Xgt=Xgt+Xg:Ygt=Ygt+Yg
 1180 NEXT node%
 1190 Xv=Xv+Xgt:Yv=Yv+Ygt
 1200 Rvs=Xv*Xv+Yv*Yv:Rv=SQR(Rvs)
 1210 ENDPROC
 1220 :
 1230 DEF PROCthrust
 1240 Xt=0.03*Xv/Rv:Yt=0.03*Yv/Rv
 1250 IF INKEY(-103) THEN Xv=Xv-Xt:Yv=Yv
-Yt:SOUND0,-7,5,9
 1260 IF INKEY(-104) THEN Xv=Xv+Xt:Yv=Yv
+Yt:SOUND0,-7,5,9
 1270 ENDPROC
 1280 :
 1290 DEF PROCflags
 1300 IF INKEY(-52) THEN rerun%=TRUE
 1310 IF INKEY(-17) THEN rerun%=TRUE:qui
t%=TRUE
 1320 ENDPROC
 1330 :
 1340 DEF PROCspaceship
 1350 Xs=640:Ys=985
 1360 Xv=27:Yv=0
 1370 GCOL0,3:MOVE Xs-16,Ys+16:PRINT"*"
 1380 ENDPROC
 1390 :
 1400 DEF PROCmoon
 1410 GCOL0,2
 1420 Ca=COS(PI/20):Sa=SIN(PI/20)
 1430 CA=1:SA=0:MOVE 640+350,512
 1440 FOR A=1 TO 40
 1450 Cp=CA:Sp=SA
 1460 CA=Cp*Ca-Sp*Sa:SA=Sp*Ca+Cp*Sa
 1470 x=350*CA+640:y=350*SA+512
 1480 MOVE 640,512:PLOT85,x-10+RND(20),y
-10+RND(20)
 1490 NEXT A
 1500 GCOL0,0
 1510 FOR n%=1 TO 50
 1520 MOVE 290+RND(700),162+RND(700):PRI
NT"^"
 1530 MOVE 290+RND(700),162+RND(700):PRI
NT"-"
 1540 MOVE 290+RND(700),162+RND(700):PRI
NT"."
 1550 MOVE 290+RND(350),162+RND(700):PRI
NT"-"
 1560 MOVE 290+RND(350),162+RND(700):PRI
NT"."
 1570 NEXT n%
 1580 ENDPROC
 1590 :
 1600 DEF PROCnodes
 1610 Xn(0)=640:Yn(0)=512:Mn(0)=200
 1620 angle=RND(1)
 1630 FOR node%=1 TO 6
 1640 A=angle+node%-1
 1650 Xn(node%)=640+300*COS(A):Yn(node%)
=512+300*SIN(A):Mn(node%)=RND(10)+20
 1660 NEXT node%
```

```
1670 ENDPROC
1680 :
1690 DEF PROCcircle
1700 GCOL0,1
1710 CA=1:SA=0:MOVE 640+500,512
1720 FOR A=1 TO 40
1730 Cp=CA:Sp=SA
1740 CA=Cp*Ca-Sp*Sa:SA=Sp*Ca+Cp*Sa
1750 x=500*CA+640:y=500*SA+512
1760 DRAW x,y
1770 NEXT A
1780 GCOL3,3
1790 ENDPROC
1800 :
1810 DEF PROCinfo
1820 y$=CHR$131:c$=CHR$134:w$=CHR$135
1830 PRINT TAB(13,2)y$+"NODAL MOON"
1840 PRINT'c$+"A scout ship is surveyin
g a small"
1850 PRINT c$+"moon. The moon's mass is
 irregularly"
1860 PRINT c$+"distributed with denser
nodes,"
1870 PRINT c$+"so that the gravitationa
l pull"
1880 PRINT c$+"varies over the surface.
"
1890 PRINT c$+"Try maintaining the ship
 below the"
1900 PRINT c$+"maximum survey altitude,
 marked by a"
1910 PRINT c$+"circle, without crashing
 "
1920 PRINT'w$+"Gently holding down the
< key will"
1930 PRINT w$+"fire the retro unit to s
low the ship."
1940 PRINT'w$+"Gently holding down the
> key will"
1950 PRINT w$+"fire the drive to accele
rate the ship."
1960 PRINT'w$+"Press any key to start."
1970 PRINT'w$+"Press R to clear the scr
een & Repeat."
1980 PRINT'w$+"Press Q to Quit."
1990 ENDPROC
```


Magscan

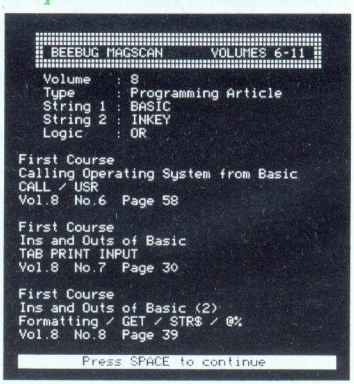
Comprehensive Magazine Database for the BBC Micro and the Master 128

***An updated version of Magscan, which contains the complete indexes to all BEEBUG magazines from* Volume 1 Issue 1 *to* Volume 11**

Magscan allows you to locate instantly all references to any chosen subject mentioned anywhere in the 110 issues of BEEBUG magazine to date.

Just type in one or two descriptive words (using AND/OR logic), and you can find any article or program you need, together with a brief description and reference to the volume, issue and page numbers. You can also perform a search by article type and/or volume number.

The Magscan database can be easily updated to include future magazines. Annual updates are available from BEEBUG for existing Magscan users.



Some of the features Magscan offers include:

- full access to all BEEBUG magazines
- rapid keyword search
- flexible search by volume number, article type and up to two keywords
- keyword entry with selectable AND/OR logic
- extensive on-screen help
- hard copy option
- easily updatable to include future magazines
- yearly updates available from BEEBUG

Magscan with disc and manual **£9.95**+*p&p*

Stock codes: 0005a 5.25"disc 40 track DFS
0006a 5.25"disc 80 track DFS
1457a 3.5" ADFS disc

Magscan update **£4.75**+*p&p*

Stock codes: 0011a 5.25"disc 40 track DFS
0010a 5.25"disc 80 track DFS
1458a 3.5" ADFS disc

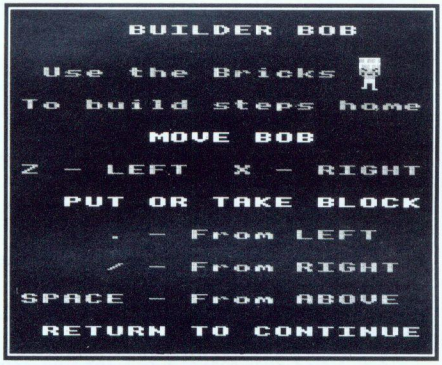
RISC Developments Ltd, 117 Hatfield Road, St Albans, Herts AL1 4JS. Tel 0727 840303 Fax 0727 860263

Builder Bob

by Richard Lewis

Bob is trapped on the bottom floor of a building that is being demolished. Will Bob be able to escape with your guidance or will he become just another brick in the wall?

Builder Bob is a fast action one player game of skill and strategy, where you have to pile up bricks to help Bob build his way out of the condemned building. This all sounds easy, until you meet the workman's crane on the second level which drops odd bricks into the rubble, and occasionally onto Bob's head. As Bob is not wearing any safety gear, a brick on the head is quite painful and fatal.

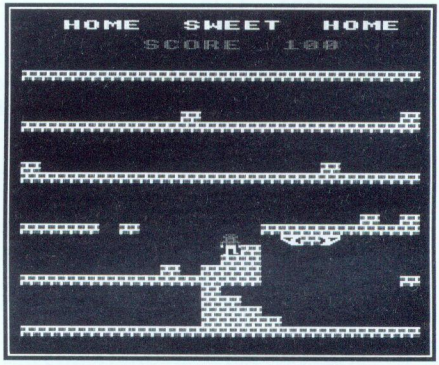


The basic instructions

To complete the game, Bob must make his way to the top platform without being killed three times. There are only two ways in which Bob can lose a life. The first is by being hit by a brick as described above, and the second is if he is hit by the crane itself.

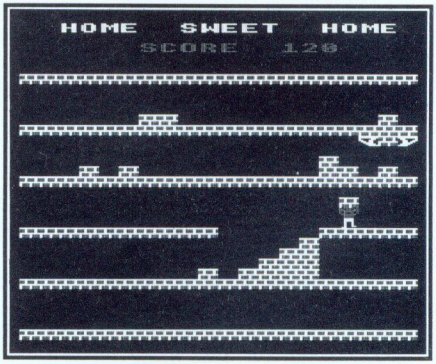
The keys for playing are included in the programs title page. They are as follows: Z and X move Bob left and right respectively; > and ? pick and drop bricks from Bob's left or right; and the space bar makes him jump to take or replace a brick from above.

There are not a lot of procedures in this program so we have outlined each one below.



First target achieved - level 2

PROCinit	Initialise variables
PROCmanch	Define characters for man
PROCmake	Define strings for characters
PROCman	Move and test man
PROCinst	Instructions
PROCscore	Print score sheet
PROCcrane	Position and move crane
PROCcheck	Test for Bob being hit by a crane
PROCcheck2	Test for Bob being hit by a brick



Now we've reached level 3

One thing to note about this game is that the crane will only appear after you have successfully completed the first level.

After that, the crane will only drop bricks between your level and the top.

Please take care when typing this program into your computer, especially with the string and character definitions in lines 1130 to 1440.

Note: this classic game program was first published in BEEBUG Vol.4 No.4. For those who haven't seen it before, it is an illustration of what can be achieved.

```
   10 REM Program Builder Bob
   20 REM Version B0.3
   30 REM Author  Richard Lewis
   40 REM BEEBUG  June 1993
   50 REM Program Subject to Copyright
   60 :
  100 ON ERROR GOTO 200
  110 MODE5:VDU23,1,0;0;0;0;:VDU19,3,5,0
,0,0:D%=0
  120 DIM C$(6),A%300
  130 ?&D0=2:?A%=0:A%=A%+1:A%?147=10:HSC
%=0:VDU23,1,0;0;0;0;:VDU23,242,0,221,221
,221,0,119,119,119:F%=0
  140 PROCinst
  150 PROCscore:PROCinit
  160 REPEAT:PROCman:PROCcrane:UNTIL F%>
4 OR ER%
  170 FOR I%=1TO4:PROCcrane:NEXT:IF MEN%
<1 OR F%>4 THEN PROCscore:MEN%=3:SC%=0
  180 GOTO150
  190 :
  200 ON ERROR OFF
  210 MODE 7:?&D0=0:IF ERR=17 END
  220 REPORT:PRINT " at line ";ERL
  230 END
  240 :
 1000 DEF PROCinit
 1010 CLS:COLOUR3:PRINTTAB(2,1)"HOME  SW
EET  HOME";:COLOUR1:PRINTTAB(6,3)"SCORE
 ";SC%;"   ";
 1020 COLOUR130:FORJ%=1TO6:FOR I%=0TO19
 1030 PRINTTAB(I%,J%*5+1);CHR$242;
 1040 K%=I%+1+J%*21+A%:?K%=1:IF J%=1 THE
N K%?-21=0:NEXT ELSE NEXT
 1050 K%=J%*21+A%:?K%=10:NEXT
 1060 COLOUR128:CP%=0:BD%=0:TA%=0:ER%=FA
LSE:BL%=0
 1070 VDU23,243,129,255,129,129,129,255,
129,129
 1080 VDU23,244,255,255,134,204,108,60,3
0,15
 1090 VDU23,245,255,255,97,51,54,60,120,
240
 1100 M%=0:IM%=0:IB%=0:IX%=0:X%=0:IY%=0:
Y%=30-5*F%:V%=0:B%=0:PROCmanch:PROCmake:
E$=E2$:M$=M2$
 1110 ENDPROC
 1120 :
 1130 DEF PROCmanch
 1140 ON 2*M%+IM%+1 GOTO 1150,1160,1170,
1180,1190,1200,1150,1160
 1150 IF B%=0 THEN VDU23,230,189,189,60,
36,36,36,100,6:GOTO 1210 ELSE VDU23,230,
60,60,60,36,36,36,100,6:GOTO 1210
 1160 IF B%=0 THEN VDU23,230,189,189,60,
36,36,36,38,96:GOTO1210 ELSE VDU23,230,6
0,60,60,36,36,36,38,96:GOTO1210
 1170 VDU23,230,60,60,24,24,28,22,19,50:
GOTO 1210
 1180 VDU23,230,60,60,24,24,56,232,72,24
:GOTO 1210
 1190 VDU23,230,60,60,24,24,56,104,200,7
6:GOTO 1210
 1200 VDU23,230,60,60,24,24,28,23,18,24:
GOTO 1210
 1210 ON M%+B%+1 GOTO 1220,1230,1240,128
0,1250,1260,1270,1280
 1220 VDU23,231,60,90,126,60,36,126,255,
189:GOTO 1280
 1230 VDU23,231,28,6,62,20,28,60,126,189
:GOTO 1280
 1240 VDU23,231,28,48,62,20,28,60,126,18
9:GOTO 1280
 1250 VDU23,231,189,219,255,189,165,126,
60,60:GOTO 1280
 1260 VDU23,231,157,135,191,149,157,126,
60,60:GOTO 1280
 1270 VDU23,231,157,177,191,149,157,126,
60,60:GOTO 1280
 1280 ENDPROC
 1290 :
 1300 DEF PROCmake
 1310 M2$=CHR$17+CHR$3+CHR$230+CHR$8+CHR
$11+CHR$17+CHR$1+CHR$231
 1320 B$=CHR$17+CHR$130+CHR$242+CHR$17+C
HR$128
 1330 M3$=M2$+CHR$8+CHR$11+B$
 1340 E1$=CHR$32
 1350 E2$=CHR$32+CHR$8+CHR$11+CHR$32
 1360 E3$=E2$+CHR$8+CHR$11+CHR$32
 1370 C$(0)=E1$
```

```
 1380 C$(1)=CHR$245+E1$
 1390 C$(2)=E1$+CHR$245+E1$
 1400 C$(3)=CHR$244+C$(2)
 1410 C$(4)=CHR$244+E1$+CHR$245
 1420 C$(5)=CHR$244+E1$
 1430 C$(6)=CHR$244
 1440 ENDPROC
 1450 :
 1460 DEF PROCman
 1470 IF IM%=0 THEN IM%=1 ELSE IM%=0
 1480 IF Y%<26-F%*5 THEN F%=F%+1:IF 10*(
F%+1)-BL%>0 SC%=SC%+50*(F%+1)-5*BL%:COLO
UR1:PRINTTAB(13,3);SC%;
 1490 K%=X%+1+((Y%+4)DIV5)*21+A%:M%=0:CO
LOUR1
 1500 IF NOT(INKEY-104) OR V%<>0 THEN GO
TO 1530
 1510 M%=1:H%=K%?-1:d%=?K%-H%:IF B%=4 AN
D H%<5 THEN B%=0:K%?-1=H%+1:PRINTTAB(X%-
1,Y%+d%);B$;TAB(X%,Y%-2);E1$;:SOUND1,1,1
00,6:GOTO 1560
 1520 IF B%=0 AND H%>1 AND H%<6 AND (K%?
-21=0 OR ?K%<3) THEN B%=4:K%?-1=H%-1:PRI
NTTAB(X%-1,Y%+d%+1);E1$;:SOUND1,1,100,6:
GOTO 1560
 1530 IF NOT(INKEY-105) OR V%<>0 THEN GO
TO 1560
 1540 M%=2:H%=K%?+1:d%=?K%-H%:IF B%=4 AN
D H%<5 THEN B%=0:K%?1=H%+1:PRINTTAB(X%+1
,Y%+d%);B$;TAB(X%,Y%-2);E1$;:SOUND1,1,10
0,6:GOTO 1560
 1550 IF B%=0 AND H%>1 AND H%<6 AND (K%?
-21=0 OR ?K%<3) THEN B%=4:K%?+1=H%-1:PRI
NTTAB(X%+1,Y%+d%+1);E1$;:SOUND1,1,100,6:
GOTO 1560
 1560 IF INKEY-67 AND V%=0 THEN IX%=1:M%
=2
 1570 IF INKEY-98 AND V%=0 THEN IX%=-1:M
%=1
 1580 IF INKEY-99 AND V%=0 AND (B%=0 OR
(B%=4 AND K%?-21=0)) AND IX%=0 AND ?K%<3
 THEN V%=-?K%:IY%=-1:SOUND1,-15,120,2:GO
TO 1690
 1590 IF V%=0 THEN IY%=0:GOTO 1680 ELSE
IF V%>0 THEN V%=V%+1:IY%=1:M%=0:SOUND1,-
15,(7-V%)*20,4
 1600 IF V%>6-?K% THEN V%=0
 1610 IF V%=-1 THEN IY%=-1:V%=-2:SOUND1,
-15,150,2:GOTO1690
 1620 IF V%=-3 AND TA%=1 THEN TA%=0:PRIN
TTAB(X%,Y%-3);B$;TAB(X%,Y%-3-K%?-21);E1$
 1630 IF V%=-3 THEN V%=0:IF ?K%=1 THEN I
Y%=1
 1640 IF V%=-2 AND B%=0 AND K%?-21>0 THE
N V%=-3:IY%=1:B%=4:K%?-21=(K%?-21)-1:SOU
ND1,1,100,5:TA%=1+((K%?-21)=0):GOTO 1680
 1650 IF V%=-2 AND B%=4 THEN V%=-3:IY%=1
:B%=0:K%?-21=1:SOUND1,1,100,5:GOTO 1680
 1660 IF V%=-2 THEN V%=-3:IY%=1:SOUND1,-
15,180,5
 1670 IF V%=5 AND B%=4 THEN B%=0:K%?-21=
1
 1680 IF B%=0 THEN M$=M2$:E$=E2$ ELSE M$
=M3$:E$=E3$
 1690 PROCmanch:IF IY%=0 AND IX%=0 GOTO
1760
 1700 IF K%?IX%=0 AND ?K%=1 AND Y%<26 AN
D K%?(IX%+21)<5 THEN V%=1:GOTO 1750
 1710 IF ?K%=5 AND K%?(IX%-21)=1 THEN D%
=-1:GOTO 1750
 1720 IF ?K%=1 AND Y%<26 AND K%?IX%=0 AN
D K%?(IX%+21)=5 THEN D%=1:GOTO 1750
 1730 IF K%?(IX%-21)<>0 AND K%?IX%<>10 A
ND K%?IX%>3-B% DIV4 THEN IX%=0:GOTO 1750
 1740 D%=?K%-K%?IX%:IF ABS(D%)>6 THEN D%
=0 ELSE IF ABS(D%)>1 IX%=0:D%=0
 1750 PRINTTAB(X%,Y%);E$;
 1760 Y%=Y%+IY%+D%:D%=0:IY%=0:X%=X%+IX%:
IX%=0:IF X%<0 THEN X%=19:Y%=Y%+?K%-K%?19
+(K%?19=0):V%=-(K%?19=0) ELSE IF X%>19 T
HEN X%=0:Y%=Y%+?K%-K%?-19+(K%?-19=0):V%=
-(K%?-19=0) ELSE GOTO 1780
 1770 IF B%=4 THEN B%=0:M$=M2$:SOUND 1,1
,100,10
 1780 PRINTTAB(X%,Y%);M$;
 1790 ENDPROC
 1800 :
 1810 DEF PROCinst
 1820 CLS:COLOUR3:PRINTTAB(5,2)"BUILDER
BOB";
 1830 M%=0:IM%=0:B%=4:PROCmanch:PROCmake
:PRINTTAB(16,7)M3$;
 1840 PRINTTAB(1,6)"Use the Bricks";
 1850 PRINTTAB(0,9)"To build steps home"
;
 1860 COLOUR2:PRINTTAB(6,12)"MOVE BOB";
 1870 COLOUR1:PRINTTAB(0,15)"Z - LEFT  X
 - RIGHT";
 1880 COLOUR2:PRINTTAB(2,18)"PUT OR TAKE
 BLOCK";
 1890 COLOUR1:PRINTTAB(4,21)". - From LE
FT";TAB(4,24)"/ - From RIGHT";
 1900 PRINTTAB(0,27)"SPACE - From ABOVE"
```

```
;
 1910 COLOUR3:PRINTTAB(1,30)"RETURN TO C
ONTINUE";
 1920 REPEAT UNTIL INKEY-74
 1930 SC%=0:MEN%=3:ENVELOPE 1,1,6,0,-6,2
00,100,200,100,2,0,-1,120,110
 1940 ENDPROC
 1950 :
 1960 DEF PROCscore
 1970 M%=0:B%=0:CLS:COLOUR3:PRINTTAB(4,2
)"SCORE SHEET";
 1980 IF MEN%<1 OR F%>4 THEN F%=0:COLOUR
3:PRINTTAB(4,6)"GAME OVER";:GOTO 2020
 1990 PRINTTAB(3,10)"MEN LEFT";
 2000 PRINTTAB(1,13)"RETURN TO CONTINUE"
;
 2010 PROCmanch:FOR I%=1 TO MEN%:PRINTTA
B(10+2*I%,10)M2$:NEXT:GOTO 2050
 2020 PRINTTAB(2,10)"FINAL SCORE ";SC%;:
IF SC%>HSC% THEN HSC%=SC%:COLOUR2:PRINTT
AB(2,14)"NEW ";
 2030 COLOUR2:PRINTTAB(6,14)"HIGH SCORE"
;:PRINTTAB(10,16);HSC%;
 2040 COLOUR1:PRINTTAB(4,20)"PRESS RETUR
N";:PRINTTAB(3,22)"FOR A NEW GAME";:GOTO
 2060
 2050 COLOUR2:PRINTTAB(2,6)"Score so far
";SC%;
 2060 BX%=1:BI%=1:M%=2
 2070 COLOUR1:COLOUR130:PRINTTAB(0,31);S
TRING$(20,CHR$242);TAB(0,26);STRING$(3,C
HR$242);TAB(6,26);STRING$(8,CHR$242);TAB
(17,26);STRING$(3,CHR$242);
 2080 FORI%=1TO4:PRINTTAB(3,26+I%);STRIN
G$(I%,CHR$242);TAB(17-I%,26+I%);STRING$(
I%,CHR$242);:NEXT:COLOUR128
 2090 OBY%=25:OBX%=0
 2100 REPEAT
 2110 BX%=BX%+BI%:IF BX%<0 THEN BX%=0:BI
%=1:M%=2 ELSE IF BX%>19 THEN BX%=19:BI%=
-1:M%=1
 2120 BY%=25:IF BX%>6 AND BX%<13 THEN BY
%=30:GOTO 2150
 2130 IF BX%>2 AND BX%<7 THEN BY%=23+BX%
:GOTO 2150
 2140 IF BX%>12 AND BX%<17 THEN BY%=42-B
X%
 2150 PRINTTAB(OBX%,OBY%)E2$;:PRINTTAB(B
X%,BY%)M2$;:OBY%=BY%:OBX%=BX%:FORW%=0TO3
00:NEXT
 2160 IF IM%=0 THEN IM%=1:IB%=1 ELSE IM%
=0:IB%=0
 2170 PROCmanch:UNTIL INKEY-74
 2180 ENDPROC
 2190 :
 2200 DEF PROCcrane
 2210 IF F%<1 THEN FOR W%=1TO100:NEXT:EN
DPROC
 2220 IF CP%>25 THEN CP%=CP%-1:ENDPROC
 2230 IF CP%>0 THEN CP%=CP%-1:GOTO 2250
ELSE CP%=30:VDU23,244,255,255,134,204,10
8,60,30,15:VDU23,245,255,255,97,51,54,60
,120,240
 2240 FL%=5-RND(6-F%):FL%=FL%-(FL%<F%):B
D%=0:DR%=RND(20)-1:P%=DR%+1+(6-FL%)*21+A
%:IF ?P%>3 THEN DR%=-6:ENDPROC ELSE BL%=
BL%+1:ENDPROC
 2250 IF CP%>22 THEN CR$=C$(CP%-19):GOTO
 2280
 2260 IF CP%<7 AND CP%>2 THEN CR$=C$(CP%
-3):GOTO 2280
 2270 CR$=C$(3)
 2280 IF CP%=DR%+5 THEN VDU23,244,255,25
5,134,204,120,56,24,12:VDU23,245,255,255
,97,51,30,28,24,48:BD%=5:PROCcheck2
 2290 IF ?K%>2-B% DIV4 THEN PROCcheck
 2300 IF CP%>5 COLOUR2:PRINTTAB(CP%-6,27
-FL%*5);CR$;:GOTO 2320
 2310 IF CP%>2 COLOUR2:PRINTTAB(0,27-FL%
*5);CR$;
 2320 IF CP%>4 AND CP%<25 AND CP%>DR%+5
THEN COLOUR1:PRINTTAB(CP%-5,27-FL%*5);B$
;
 2330 Q%=CP%-2+(6-FL%)*21+A%:IF CP%>2 AN
D CP%<23 AND ?Q%=5 THEN COLOUR1:PRINTTAB
(CP%-3,27-FL%*5);B$;
 2340 IF BD%>?P%+1 THEN PROCcheck2:BD%=B
D%-1:COLOUR1:PRINTTAB(DR%,31-FL%*5-BD%);
E1$;TAB(DR%,32-FL%*5-BD%);B$;:IF BD%=?P%
+1  ?P%=?P%+1
 2350 ENDPROC
 2360 :
 2370 DEF PROCcheck
 2380 IF X%=CP%-5 AND FL%=6-(Y%+4)DIV5 A
ND ER%=FALSE THEN ER%=TRUE:MEN%=MEN%-1:S
C%=SC% DIV2
 2390 ENDPROC
 2400 :
 2410 DEF PROCcheck2
 2420 IF X%=DR% AND ER%=FALSE AND ((FL%=
7-(Y%+4)DIV5 AND ?K%>3-B% DIV4) OR FL%=6
-(Y%+4)DIV5) THEN ER%=TRUE:MEN%=MEN%-1:S
C%=SC% DIV2
 2430 ENDPROC
```

Wordwise User's Notebook:
Change Parameters

David Polak provides an easy means of changing parameters in VARPRN, a printing utility with programmable line spacing published in Vol.10 no.10.

I found the article *Printing Utility with Programmable Line Spacing* by Mr C.W. Robertson published in Wordwise User's Notebook for BEEBUG Vol.10 No.10 (April 1992) very interesting and useful. If you feel the same, then this program which works alongside it should prove just as helpful.

Here is an extension program which includes a simple menu to enable changes to be made as required to the number of lines (L%), the text file from which data is extracted (K$), the line feeds (M$) or the increments (N$). Simply load Mr. Robertson's program into Segment 1 and this Change Parameters program into Segment 7.

Pressing Shift-f7 reveals a simple menu to select the action required. Shift-f1 then enables printing to proceed with the new line spacing or whatever.

PART ONE

Part one of the Change Parameters program will amend Segment 1 if this has not already been done; L$ is added so that all the parameters are expressed as strings. The *forget* label enables the program to work with Segment 1 as printed in the April 1992 issue or after it has been amended. L%=7 (if present) will be replaced with L$=7 followed by a new line L%=VAL(L$).

N$ is modified to include four extra characters so that when N$ and M$ are seen expressed in pairs of digits, e.g. 05 not 5, it is easy to relate their corresponding values in Segment 1.

The Note N1 refers to removing the three letters rem to activate the line above. This will enable the program to work more quickly, but slightly differently! Its worth trying - I prefer the faster version which avoids repeating Part TWO.

PART TWO

Part twos does ensure that 'old' values are always read-in for comparison purposes and the old values are then lost.

PART THREE

Part three does the changing and is pretty staightforward. Note that when either M$ or N$ is changed, only the numerical values (in pairs of digits) should be entered, as the program inserts the words and commas automatically. It asks for the second digit if not entered.

The Note N2 refers to the following 7 lines which are a bit long winded to cater for a two-line definition of M$ or N$. I t uses Mr. Robertson's attractive procedure, in this case to find the end quotation marks.

Note: both the Segment program listed here and Mr. Robertson's original are included on this month's magazine disc for convenience.

```
REM..... Put all this in SEGMENT 7

REM..... CHANGE PARAMETERS PROGRAM
REM.....    by  D Polak  JUNE 1993

REM.....       For the
REM..... "Printing Utility - with
```

```
REM..... Programmable Line Spacing"
REM.....    by C W Robertson
REM..... see BEEBUG - APRIL 1992

REM... ================       PART ONE
REM... Modifies "VARPRN1" which should
REM... be loaded into Segment 1

rem IF K$<>""THEN GOTO notfirst
REM Remove "rem" above? - see notes..N1
SEL.SEG.1
CURSOR TOP
FIND"L%=7"
IF EOT THEN G.feed
REPLACE"%=7","$=""7"""
CURSOR AT 0
TYPE"L%=VALL$|R"
IF EOT THEN G.feed
REPLACE"Lfeeds","Lfeeds...."
.feed

REM... ================       PART TWO
REM.. Collects segment 1 strings. These
REM   are also found by running Seg 1.
SEL.SEG.1
CURSOR TOP
FIND "L$="
L$=GLT$
FIND "K$="
K$=GLT$
FIND "M$="
M$=GLT$
FIND "N$="
N$=GLT$

SEL.SEG.8
DEL.TEXT
TYPE L$+"|R"
TYPE K$+"|R"
TYPE M$+"|R"
TYPE N$+"|R"
TYPE"|R"
CURSOR TOP

DOTHIS
 DEL. AT 4
  REPEAT
   Z$=GCT$
  UNTIL Z$=""""
  DEL.L.
  CURSOR DOWN
  CURSOR AT 0
TIMES 4
CURSOR TOP
L$=GLT$
K$=GLT$
M$=GLT$
N$=GLT$
.notfirst


REM... ================     PART THREE
REM... menu & enables changes

PROCselect
PROCchange
PROCputstring
PROCmessage
END

.select
REPEAT
  CLS
  VDU31,3,5,131
  P."1 - TITLE"
  VDU31,3,6,131
  P."2 - No. of lines"
  VDU31,3,7,131
  P."3 - Lfeeds"
  VDU31,3,8,131
  P."4 - Increments"
  VDU31,3,10,134
  P." Please select   "
  *FX15,1
  S$=GCK$
  S%=valS$
UNTIL S%<5 AND S%>0
IF S%=1 THEN Z$=K$
IF S%=2 THEN Z$=L$
IF S%=3 THEN Z$=M$
IF S%=4 THEN Z$=N$
ENDPROC
```

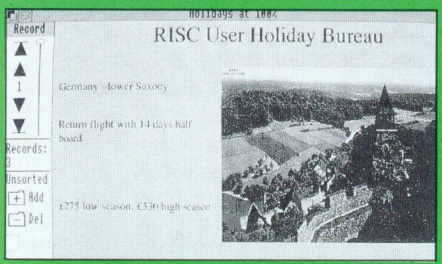
```
.change
REPEAT
   REPEAT
    CLS
    VDU31,8,3,131
    P."Value NOW"
    VDU31,4,5,134
    P.Z$
    VDU31,3,7,131
    P."Change this ? (Y/N)     "
    *FX15,1
    A%=GET AND &DF
   UNTIL A%=89 OR A%=78
  IF A%<>89 THEN G.nohome
  Z$=""
  VDU31,3,7,133
  P."KEY IN the NEW requirement"
  VDU31,5,10,133
  P."End LINE with <RETURN>"
  IFS%<3 THEN PROCordinary
  IFS%>2 THEN PROCpairs_of_digits
UNTIL A%<>89
.nohome
ENDPROC

.ordinary
REPEAT
  X$=GCK$
  IF ASC(X$)=13 THEN G.home
  Z$=Z$+X$
  .home
  VDU31,3,12,134
  P.Z$
UNTIL ASC(X$)=13
ENDPROC

.pairs_of_digits
IF S%=3 THEN Z$=Z$+"Lfeeds....,"
IF S%=4 THEN Z$=Z$+"Increments,"
VDU31,5,9,134
P."TYPE PAIRS eg 06 etc "
REPEAT
  Y$=GCK$
  VDU31,3,12,134
  P.Z$+Y$
  IF ASC(Y$)=13 THEN G.home2
  REPEAT
     X$=GCK$
  VDU31,2,12,130
       IF ASC(X$)=13 THEN P."TYPE NEXT
NUMBER-"
  UNTIL ASC(X$)<>13
  Z$=Z$+Y$+X$+","
  .home2
  VDU31,3,12,134
  P.Z$
UNTIL ASC(Y$)=13
ENDPROC

.putstring
SEL.SEG.1
CURSOR TOP
IF S%=1 THEN FIND "K$="
IF S%=2 THEN FIND "L$="
IF S%=3 THEN FIND "M$="
IF S%=4 THEN FIND "N$="
REM Remove old data. see note...N2
CURSOR RIGHT 3
FK.3
CURSOR R.
  REPEAT
   Q$=GCT$
  UNTIL Q$="""
FK.3
FK.7
REM Put new data
TYPE"""+Z$+"""
DIS.
ENDPROC

.message
VDU31,0,9,133
P."PRESS ANY KEY          THEN:-   "
VDU31,0,10,131
P."<SHIFT/f1> to continue          "
VDU31,0,11,131
P."<SHIFT/f7> to change parameters"
P."                                "
A%=GET
DISPLAY
ENDPROC
```

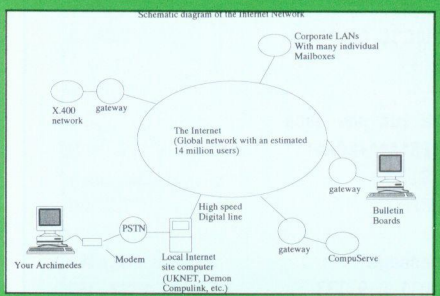
RISC user

The number one subscription magazine for the Archimedes

RISC User, the highly popular magazine for Archimedes users, is bigger and better. The new RISC User is now B5 size which offers a sophisticated design, bigger colour illustrations and bigger pages with more information. Altogether better value and no increase in price.

RISC User is still a convenient size to assemble into an easy-to-use reference library, containing all the information you need as an Archimedes user. Every issue of RISC User offers a wealth of articles and programs with professionally written reviews, lively news, help and advice for beginners and experienced users, and items of home entertainment. Altogether RISC User has established a reputation for accurate, objective and informed articles of real practical use to all users of Acorn's range of RISC computers.





SUBSCRIPTION DETAILS

As a member of BEEBUG you may subscribe to RISC User for the reduced rate of £18.40 (a saving of £1.50 on the normal subscription rate). Overseas subscription rates are as follows: £27.50 Europe and Eire, £33.50 Middle East, £36.50 Americas & Africa, £39.50 Elsewhere

WIMP TOPICS

A new series aimed at readers interested in Wimp programs and Wimp programming. Each article will look at aspects of a particular topic.

ORGANISE YOURSELF

Almanac is a computerised personal organiser which includes a diary, address book, notepad etc and looks just like the real thing.

THE ADVANCE DATABASE

A closer look at the database in Acorn's new Advance integrated package.

INTRODUCING INTERNET

Internet is an email network, which offers access to a vast number of users very cheaply. The article explains more about Internet and how to join it.

3D ICONS FOR ALL

Smarten up your applications with this new feature of RISC OS 3.

SERENADE YOUR ARC

The latest MIDI sequencer from Clares, powerful and easy to use, is aimed at education.

BOOK REVIEWS

Two books reviewed - Graphics on the ARM Machines and Dabhand Guide to Impression.

WRITE-BACK

The readers' section of RISC User for comment, help, information - a magazine version of a bulletin board.

INTO THE ARC

A regular series for beginners.

TECHNICAL QUERIES

A column which answers your technical queries.

THE DOS SURVIVAL GUIDE

A series of articles on how to use the PC Emulator.

HINTS and tips HINTS and tips HINTS and tips HINTS and tips HINTS and tips

Please keep sending in any tips for all BBC and Master computers. Remember, if your hint gets published, there's money in it!

AVOIDING PROGRAM LOSSES

by John Nicholson

If you have suffered the loss of a program through a power cut or other accident there are two simple ways to reduce the dangers. Insert the following lines, to set up two function keys, into whatever start-up utility you use:

```
*KEY8 SAVE ":0."+programname$|M
*KEY9 SAVE ":0.WORK"|M
```

I set up all the function keys to perform useful routines in this way. Now, every time you write a program, start it with a title line as follows:

```
10 programname$="EXAMPLE":REM program name
```

Whenever you need to save a program you can do so by pressing function key f9, and the file called *WORK* will hold the current version. Pressing f8 instead will ensure the program is saved under its correct name. For the latter to work, your program must have been run after the last change has been made to it.

AVOIDING TEXT LOSSES

by D.Jowers

Many word processing packages automatically save text at preset intervals, so in the event of hardware failure little is lost. If the following function key definitions are loaded as part of the View !Boot file, it is easy to save text frequently, and without any need to return to the command screen:

```
*FX228,1
*KEYi |[LOAD V.WORK|M*FX125|M
*KEYj |[SAVE V.WORK|M*FX125|M
```

The first line enables Ctrl-Shift-function keys, the second performs a load, and the third a save, where 'i' and 'j' are any two function key numbers. At any time, Ctrl-Shift-j will save the current text file, and Ctrl-Shift-i will load the text last saved.

QUITTING *EXEC

by Bill Walker

Normally a *EXEC file that runs a series of programs in sequence, as !Boot files often do, will still try, for example, to run a third program even when there has been an error in the second. This can be avoided by including a CLOSE#0 instruction in any error trapping routine in each program. This will close any open files including the *EXEC file, thus halting the process.

SINGLE KEY STRING SEARCH

by David Jupe

The following key definition sets up function key f0 to search for any specified string in a Basic program:

```
*KEY|N|!hLINEN$:P=PA.+1:@%=5|!uN=256*P+P?1:
  M=P+3:P=P+P?2:|!G|!$M,N$)||qN;:|!uA%=?M:
  |!V&B50E:M=M+1:U.A%=13:|!q:U.?P=255EL.U.
  ?P=255|M
```

Enter the search string after the prompt. All lines containing the string are then displayed although line numbers following GOTO and GOSUB statements are not displayed correctly. Entering a null string for the search string displays the whole program at a reduced rate. If you still happen to have Basic I then replace &B50E with &B53A. Master users have Edit built into their machine so have no need of this function, but if you want to try it out the equivalent value is &BD37 on a Master 128 and BC97 on a Master Compact.

Personal Ads

BEEBUG members may advertise unwanted computer hardware and software through personal ads (including 'wants') in BEEBUG. These are completely free of charge but please keep your ad as short as possible. Although we will try to include all ads received, we reserve the right to edit or reject any if necessary. Any ads which cannot be accommodated in one issue will be held over to the next, so please advise us if you do not wish us to do this. We will accept adverts for software, but prospective purchasers should ensure that they always receive original copies including documentation to avoid any abuse of this facility.

We also accept members' Business Ads at the rate of 40p per word (inclusive of VAT) and these will be featured separately. Please send all ads (personal and business) to MEMBERS' ADS, BEEBUG, 117 Hatfield Road, St. Albans, Herts AL1 4JS.

BBC B issue 3 with DDFS, 32k Shadow RAM and Sideways ROM/RAM boards, View Professional, Toolkit Plus, Exmon, Sleuth, 5.25" & 3.5" DS 80T disc drives, Revs, Aviator £160, BEEBUG magazines Vols. 1-11 complete and bound £60. Tel. (0923) 239788.

For Sale Epson FX80 printer, good condition £50. Tel. (0245) 353750.

WANTED: Copy of complete BBC manual (not Master). Tel. (0533) 775145.

WANTED: for M128, secondhand or very cheap but reliable screen to printer dump program and instructions, also printer fonts for Panasonic KX P1081. Tel. Steve Fowler 071-796 3171 (work) or 081-529 4470 (home) after 6.30pm or w/ends.

Music Software; with microphone interface, 5.25" disc, handbook, Mupados Recorder Tutor with Ensemble, Duet and Classroom Network packs (5 x 5.25" discs, handbooks and cassettes). Micro Maestro with 5.25" disc and 6 cassettes, all for £28 including postage. (worth £179) Tel. (0256) 27018.

WANTED: Adventure games for Master 128, must be original with full documentation, disc version where available, Enthar7, Blood of the Mutineers, Village of Lost Souls, Silicon Dreams, Jewels of Darkness, Zork Trilogy, Oxbridge, Programmers Revenge, What's Eeyore's Locks of Luck, Adventure Soft/Scott Adams series, reasonable prices paid. Tel. (0268) 794928.

WANTED: For a BBC model B, a sideways RAM/ROM board like Watford BEA 0260 or similar, a 32k shadow RAM printer buffer board, Interword Spelling ROM (Spell-Master) by Computer Concepts, and also the Command ROM or any other sideways ROM that will drive a modem. Tel. (0286) 830312.

WANTED: News from any other BBC users in North Wales area. Tel. (0286) 830312.

WANTED: Educational software for a BBC B with sideways RAM - Best Four Adventure, Doom, Hooray for Henrietta etc. etc. - Anyone know what happens to the schools software when they trade up? Also instructions for 'Replay' ROM, Wordwise + and Ultracalc, I have just bought them only to find all the mags etc. writing-off the BEEB even though it's still used in classrooms at school! HELP PLEASE!! Tel. (0934) 622209.

3 x BBC B, issue 7 fitted to viglen 'PC look-alike' beige case with integral 5.25" disc drive and space for monitor, separate keyboard via curly cable, in working order and each with DFS, Wordwise+ and Intersheet £60 o.n.o. Philips monitors at £20 each. Tel. (0821) 642652 eves.

Acorn Z80 2nd processor, £35, BBC B+/Torch Z80 disc pack with twin D/S 80T drives and own PSU £85 o.n.o. BBC B+ with 1770 DFS, Wordwise+, Wordaid, Spellcheck and Intersheet £60 o.n.o. Tel. (0821) 642652.

M128 fitted with BEEBUG Master ROM, Acorn alternative ROM, BEEBUG Master modem, Wapping Editor DTP, Wapping Scanner, Video Digitiser and User Port splitter, Printmaster, WYSIWYG, manuals including Master Reference manuals I & II, Masterfile II, Command, Wapping Art and Font discs, Quest mouse, Superior Software Speech, Acorn printer driver generator, Epson LQ500 printer c/w cut sheet feeder, Technomatic 40/80 twin drives in bridge, Philips CM8833 monitor c/w Rediffusion TV tuner, BEEBUG mags and discs Vol.6 to date £750 the lot. Tel. 081-449 4490.

Dual 5.25" disc drive £40, also Master Reference manuals I & II, New Advanced User Guide, Advanced SRAM User Guide, various Golf/Bridge 5.25" discs, Master ROM, Dumpmaster on cartridge, all £5 each. Tel. 081-950 2218 5pm to 6.30pm.

Nidd Valley Illustrator, Colour Box and mouse - with manuals £20 o.n.o. Tel. 073 129 372.

Misco Omniscope, relieves eye strain and magnifies text, (price new £205), being sold for £25. Tel. 081-318 5155.

WANTED: Master Turbo Welcome Manual and disc, also good Master 512 software write to; Peter Ellis, The Time-Trail of Roses, Westfield Road, Wells, Somerset BA5 2RB.

WANTED: Teletext adaptor, numeric keypad, Plotmate, Graphpad, Symphony keyboard, Penman Plotter. Tel. (0308) 862972.

HELP! Has anyone got manuals for sale/loan for the Epson FX-80 printer and/or the Epson HI-80 plotter. Also wanted for the plotter are pens, software and anything else that might be useful to help me get it up and running on a BBC B or Master 128. Tel. (0727) 830264.

W.E. Video Digitiser £55, Casio HT-3000 keyboard/synthesiser (5 octaves full size keys, Midi, Sustain pedal, foot volume, power pack, RA100 RAM card) £195, Casio MT-540 keyboard (4 octave mini keys, Midi) £35, Casio FX-700P programmable calculator + FA-3 cassette interface £25, Acorn Teletext adaptor + advanced teletext ROM £20, PMS multifont NTQ + utility & 4 font discs £20, Advanced disc investigator, Snatch, CJE Multifont NLQ + 4 fonts discs, Graphito, Navex v3.1 + charts, Holed Out Extra courses 1&2, Nidd Valley Digimouse - all £9 each. Tel. (0883) 345294 eves.

BEEBUG Command software ROM and manual & Magic modem £25, Viewspell ROM, discs and manual £6, BEEBUG magazines Vol.6 No.3 through to Vol.12 No.1 £10. Tel. 081-529 4470 after 6pm w/days and anytime w/ends.

WANTED: Discovering BBC Micro Machine Code and Get More from BBC Machine Code by A.P. Stephenson, publishers Granada Publishing. Tel. (0642) 822958.

BBC B issue 7 with Opus DDOS and double 5.25" disc drive, Watford ROM/RAM board with Wordwise, BEEBUG C and Replay, Music 500, CUB monitor, loads of software (games and educational software), also BEEBUG vols. 1-11, Acorn User, A&B Computing and Micro User 1983-1990, plus other books, sensible offer secures - will split. Tel. (0793) 538060.

BBC/Master software - mostly games, Write with s.a.e. for full list to; Mr RJ Hesketh, Horwood Hall, Keele University, Staffs ST5 5BG.

A410/1, 4Mb RAM, 40Mb IDE HD, RISC OS 3.1, Acorn M/S monitor, manuals and lots of software, including Learning Curve, £950 o.n.o. Tel. 071-703 5675.

WANTED: Viglen PC style case for Master 128. Tel. (0727) 830264.

WANTED: Printed circuit design software, Apricote Account/Invoice program, Mewsoft Filofax software to run on expanded BBC B. Tel. 081-891 5827.

WANTED: The Advanced Basic ROM User Guide For The BBC (by Colin Pharo). Tel. 071-260 2363 day or (0932) 225383 eves.

*SRWRITE/*SRREAD ON A BEEB

I note Mr.Parson's plea for *SRWRITE/*SRREAD emulations for the BBC Model B (*Postbag*, BEEBUG Vol.11 No.9). Yet from the details he gives he must have the 1770 DFS fitted, and his machine will therefore recognise these commands.

These seem to be undocumented, but *SRWRITE does work on my Model B, (with a 1770 DFS) and it recognises *SRREAD, although so far I haven't been able to fathom the syntax, and get various error messages when I try. I also find my machine recognises *SRDATA, *SRLOAD and *SRROM, but again can't get them to work, due to lack of information on the syntax.

Both Gareth Leyshon and Mr.Toad in recent articles helped a little, but they assume readers are au fait with the syntax of these commands, yet even Master 128 owners will get no help from their User Guide - I know because I have looked. To confuse the issue, Mr.Toad tells us that the syntax for *SRLOAD and *SRWRITE commands differ slightly in the Compact, so one is left wondering if there is any further difference in the 1770 DFS of the Model B. Can we have an article on this subject please?

Bernard Beeston

*We will look into the suggestion of an article on this subject. One reference for information is Volume One of the Master Reference Manual which costs £14.95. According to this the syntax for *SRREAD and *SRWRITE is as follows:*

```
*<command> <start address> <end address>
 <sideways start> [<rom id>]
```

or:

```
*<command> <start address>+<length>
 <sideways start> [<rom id>]
```

*Both commands have the same syntax. *SRREAD copies from sideways RAM to main memory, *SRWRITE copies from main memory to sideways RAM. If the sideways RAM address is in the range &8000 - &BFFF then <rom id> must be given, if this address is in the range &0 - &FFBF then <rom id> is omitted as the address implicitly specifies the RAM bank.*

PERMUTATIONS

With reference to Mr. Lindsell's letter in *Postbag*, BEEBUG Vol.11 No.10, a simple method of generating permutations is as follows. Set out the first few elements as:

```
1 2 3   3 1 2   2 1 3   1 3 2   2 3 1   3 2 1
```

Now insert in all possible positions a new element x:

```
x 1 2 3   1 x 2 3   1 2 x 3   1 2 3 x
x 3 1 2   3 x 1 2   3 1 x 2   3 1 2 x
x 2 1 3   2 x 1 3   2 1 x 3   2 1 3 x
```

etc.

This will enumerate all possible permutations. I once saw a very elegant method of doing it recursively, but I suspect it would fill the stack very quickly.

Chris Spicer

Mr.N.P.Fay also writes to say that he has written programs for permutations of strings, permutations of combinations from strings, and permutations of combinations from 4 byte integers. If anyone is interested they can write to Mr.Fay c/o BEEBUG and we will forward the letter. We also hope to publish a short item on this subject in the next issue.

BULLETIN BOARDS WANTED

Can anyone supply details of any bulletin boards supporting the BBC micro in the London area? Please phone 081 529 4470 after 6.30pm.

Steve Fowler

BEEBUG MEMBERSHIP

Send applications for membership renewals, membership queries and orders for back issues to the address below. All membership fees, including overseas, should be in pounds sterling drawn (for cheques) on a UK bank. Members may also subscribe to RISC User at a special reduced rate.

RENEWAL RATES FOR BEEBUG MAGAZINE AND MAGAZINE DISC SUBSCRIPTIONS

See May 1993 Editorial for further explanation

The table below shows the renewal rate applying after the June issue 1993 according to the first issue of the renewal period. For joint BEEBUG/RISC User subscriptions add half the appropriate BEEBUG renewal rate to the full RISC User renewal rate; (UK £18.40, Europe & Eire £27.50, Middle East £33.50, Americas & Africa £36.50, Elsewhere £39.50).

Renewal Issue	Issues to go	Mag UK	Mag Europe	Mag Mid-E	Mag Am+Af	Mag Else	Disc UK	Disc O'Seas
Jun	9	16.56	24.75	30.15	32.85	35.55	45.00	50.40
Jul	8	14.72	22.00	26.80	29.20	31.60	40.00	44.80
A/S	7	12.88	19.25	23.45	25.55	27.65	35.00	39.20
Oct	6	11.04	16.50	20.10	21.90	23.70	30.00	33.60
Nov	5	9.20	13.75	16.75	18.25	19.75	25.00	28.00
Dec	4	7.36	11.00	13.40	14.60	15.80	20.00	22.40
J/F '94	3	5.52	8.25	10.05	10.95	11.85	15.00	16.80
Mar '94	2	3.68	5.50	6.70	7.30	7.90	10.00	11.20
Apr '94	1	1.84	2.75	3.35	3.65	3.95	5.00	5.60

BACK ISSUE PRICES (per issue)

All overseas items are sent airmail. We will accept official UK orders for subscriptions and back issues, but please note that there will be a £1 handling charge for orders under £10 which require an invoice. There is no VAT on magazines.

Volume	Magazine	5"Disc	3.5"Disc
6	£1.00	£3.00	£3.00
7	£1.10	£3.50	£3.50
8	£1.30	£4.00	£4.00
9	£1.60	£4.00	£4.00
10	£1.60	£4.75	£4.75
11	£1.90	£4.75	£4.75

POST AND PACKING

Magazines and discs are postcode a
Please add the cost of p&p when ordering.
When ordering several items use the highest price code, plus half the price of each subsequent code. UK maximum £8.

Post Code	UK, BFPO Ch.I	Europe, Eire	Americas,Africa, Mid East	Elsewhere
a	£1.00	£1.60	£2.40	£2.60
b	£2.00	£3.00	£5.00	£5.50

BEEBUG
117 Hatfield Road, St.Albans, Herts AL1 4JS
Tel. St.Albans (0727) 840303, FAX: (0727) 860263
Office hours: 9am-5pm Mon-Fri Showroom hours: 9am-5pm Monday to Saturday
(24hr Answerphone for Connect/Access/Visa orders and subscriptions)

BEEBUG MAGAZINE is produced by RISC Developments Ltd.

Editor: Mike Williams
Assistant Editor: Kristina Lucas
Technical Assistant: Nick Mellor
Editorial Assistance: Marshal Anderson
Production Assistant: Sheila Stoneman
Advertising: Sarah Shrive
Subscriptions: Helen O'Sullivan
Managing Editor: Sheridan Williams

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, RISC Developments Limited.

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 £50 per page, but please give us warning of anything substantial that you intend to write. A leaflet 'Notes to Contributors' is available on receipt of an A5 (or larger) SAE.

Please submit your contributions on disc in machine readable form using plain text format if possible for text, but please ensure an adequate written description is also included of your submission and the contents/format of your disc.

In all communication, please quote your membership number.

RISC Developments Ltd (c) 1993

Printed by Arlon Printers (0923) 268328 ISSN - 0263 - 7561

Magazine Disc

June 1993

CHAOS: Exploring the Logistix Equation - Chaos theory is very much the mathematical subject of the moment and this quite short program allows you to explore some of its manifestations for yourself.

BIG NUMBERS - Have you ever needed to perform arithmetic on numbers larger than can normally be handled by Basic? The two programs provided on the disc show how this can be done for addition and subtraction.

PRESENTABLE TEXT - This is a demonstration of two routines which allow any unformatted text to be printed in justified format (left-justified, right-justified or centred) with control over the positions of left-hand and right-hand margins - a handy utility.

HUNT THE SNIB - This short game program is designed as an illustration of how to write simple games. It also makes an excellent and challenging game in its own right. Can you find the Snib?

SLIDE CATALOGUER - The disc contains the complete program for cataloguing your slide collection as described in this and the previous issue of BEEBUG. While written for the ADFS, it should not be difficult to convert for DFS use.

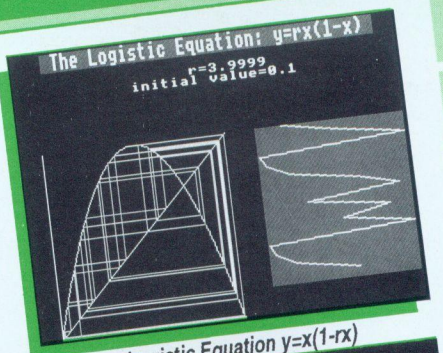
GRAVITY & ORBITS - We normally assume that gravity is constant over the surface of a moon or planet. But what if it isn't? Find out if you can cope with the consequences with this month's program in this series.

BEEBUG WORKSHOP - Two programs provide complete demonstrations of the Binary Search technique and of Hash sorting and searching.

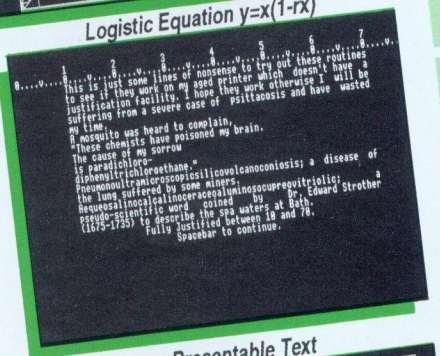
BUILDER BOB - This is a classic Beeb game which offers a challenge to all users, young and old. can you help Bob to build his way out of the stricken building before he gets killed.

WORDWISE USER'S NOTEBOOK - We have included the complete segment program from this month's magazine together with the original VARPRN program from BEEBUG Vol.10 No.10.

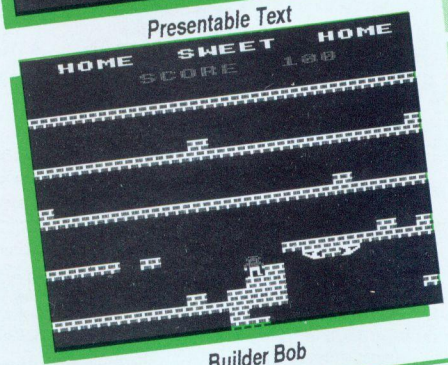
MAGSCAN DATA - Bibliography for this issue of BEEBUG (Vol.12 No.2).



Logistic Equation y=x(1-rx)



Presentable Text



Builder Bob

ALL THIS FOR £4.75 (5.25" & 3.5" DISC) + £1 P&P (50p FOR EACH ADDITIONAL ITEM)

Back issues (5.25" and 3.5" discs from Vol.6 No.1) available at the same prices.

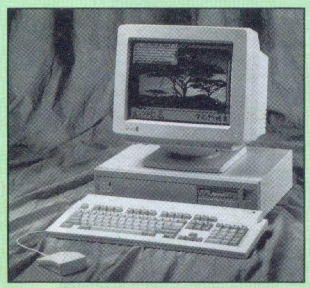
FOR DISC (5.25" or 3.5") RENEWAL SUBSCRIPTION RATES (UK AND OVERSEAS) PLEASE SEE TABLE ON FACING PAGE

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

Upgrading to an Archimedes

We know that many BEEBUG readers have already upgraded to an Archimedes, and no doubt many more will choose to follow a similar route. For their benefit we offer our advice to help them make a sensible decision on whether to upgrade and if so, what path to take.

Any prices quoted relate to our associated company Beebug Ltd., but note that all prices, particularly those on trade-ins and secondhand items, are likely to change without notice. You should always telephone or write for the latest information.



Archimedes A5000

What System to Choose

All new Archimedes systems are now supplied with the RISC OS 3.10 operating system. Any secondhand system should be upgraded to this. Based on the experience of existing users, we would strongly recommend a minimum of 2Mb of RAM. Most users find a hard disc adds significantly to the convenience of using an Archimedes, but you can always add a low-cost hard drive later, and more memory, but check on the likely price of future expansions - it is not necessarily the same for all machines. If you might be interested in more specialised add-ons (scanners, digitisers, etc.) then check the expansion capability of your preferred system.

Compatibility and Transferability

You will need to decide to what extent you wish to continue using existing discs and disc drives on an Archimedes. An Archimedes and a BBC micro can be directly connected for transfer of files. You can also connect a 5.25" drive to an Archimedes via an additional interface to continue to access 5.25" discs (ADFS format).

Our DFS Reader will also allow files to be transferred to the Arc from DFS format discs. However, none of this is possible with the latest A3010/A3020/A4000 systems.

Much BBC micro software will run directly on an Archimedes, or via the 6502 emulator. However, consider this carefully; in our experience, despite prior misgivings, most Archimedes users find that they rapidly adjust to the Desktop environment of the Archimedes, and quickly abandon the software and data of their old system after an initial period.

Software for the Archimedes

The Archimedes is supplied complete with a range of basic applications software. Before embarking on any further purchases it may be better to familiarise yourself with the new machine. Most users look for a word processor (or DTP package), maybe a spreadsheet, or a database, plus other more specialist software. We cannot give detailed guidance here, but back issues of RISC User contain a wealth of useful information - we can advise on suitable issues.

General Advice

It is advisable to discuss your requirements with the BEEBUG technical team before making a final decision on what you want. Try to anticipate future expansion needs at the outset. Note: the price on some systems includes a monitor; in other cases a choice of monitor is available at an additional cost. The details given in the table are minimum specifications of the different Archimedes models.

The A3010

It may also be possible to trade in an existing monitor and/or disc drive, but check if your existing monitor is suitable for use with an Archimedes first. You may find it better to advertise your BBC system in BEEBUG and sell privately - this applies particularly to any software and hardware add-ons which cannot be accepted for a trade-in. In future, all personal ads for Archimedes systems in RISC User will also be included in BEEBUG. You may also defer a trade-in until a later date provided you make this clear at the time of purchase.

Archimedes Systems - Typical or Current Prices

		Secondhand	New
+	A310 1Mb RAM	£350	
+	A410/1 1Mb RAM	£565	
+	A420/1 2Mb RAM, 20Mb hard drive	£650	
+	A440/1 4Mb RAM, 40Mb hard drive	£725	
+*	A3000 1Mb RAM	£350	
*	A3010 1Mb RAM, Family Solution		£ 499.00
*	A3020 2Mb RAM, 60Mb hard drive		£1056.33
*	A4000 2Mb RAM, 80Mb hard drive		£1115.08
*	A5000 2Mb RAM, 80Mb hard drive		£1643.83
+*	Acorn standard colour monitor	£145	£ 258.50

All systems above include a single floppy disc drive.
New (*) and secondhand (+) - all prices inc. VAT.
The A5000 price includes a multiscan colour monitor, A3020/A4000 price includes standard colour monitor.

BBC Micros - Typical Trade-in Prices

Model B (Issue 7)	£ 35
Model B (issue 7) + DFS	£ 75
Master 128	£125
Master Compact	£ 50

For further information on all Archimedes systems contact:
BEEBUG Ltd. 117 Hatfield Road, St Albans, Herts AL1 4JS. Tel. 0727 840303 Fax 0727 860263