UTILITIES

```
   G.MUNCHMA                 G.ROBOT
   T.ANAGRAM                 T.DIAMOND
   T.FIVDICE                 T.HIGHLO
   T.LIFE                    T.MARSLAN
   U.3DHEAD                  U.BADPROG
   U.DOUBLE                  U.RESCUE
   U.WIZARD                  V.3DROT
   V.ELLIPT                  V.SCPLAY
   V.SQDANCE                 V.UJACK

20 files of 31 on 80 tracks
>CH."U.RESCUE"
          0          E00
       3328          EFC
          1          EFD
       3328          FF9
          1          FFA
       3328         10F6
          1         10F7
         10         1100
         20         110C
         30         111F
         40         112B
         50         113E
         60         1143
         70         1151
         80         115A
         90         115F
        100         116B
        110         1179
        120         1181
        130
```

SOUND WIZARD
by Alan Webster

With 'Sound Wizard' you can produce a variety of sound effects by combining any one of the 8 preset sound envelopes with your own choice of sound parameters. This allows you to produce off-the-shelf sound effects, and takes the work out of designing envelopes for your programs.

When you run the program you will be required to enter 4 parameters:

Sound channel (0 or 1)

Envelope number (1-8)

Pitch (0-255)

Duration (0-255)

The sound will then be heard, and the sound parameters displayed. You may now enter a new set of parameters, or if you press ESCAPE the computer will tell you at which program line the appropriate envelope is to be found (so that you may transcribe it into another program).

If you wish to add envelopes (or change them), you are advised not to renumber them as it can lead to confusion.

It should be noted that you can have only one Envelope defined at any time so the Envelopes have to be redefined when required, hence the use of PROCENV(E) where E is the Envelope number.

Given below are some examples to try:

Sound	Envelope	Pitch	Duration
1	1	200	40
1	2	200	50
1	3	4	100
0	3	5	40
1	4	50	100
1	5	200	150
1	6	0	10
1	7	20	100
0	8	6	20

```
   10  REM Sound Wizard
   20  REM by A.Webster
   30  REM BEEBUG
   40  REM VERSION P 1.0
   50 :
   60  ON ERROR GOTO 450
   70  E=0
   80  MODE 6
   90  PRINTTAB(13,0)"SOUND WIZARD"
  100  PRINTTAB(12)"by Alan Webster"
  110  PRINT'TAB(9)"8 ENVELOPES DEFINED"
  120  PRINTTAB(0,5);" Channel  Envelope  Pitch  Du
ration"
  130  INPUTTAB(1,7),C:INPUTTAB(10,7),E
  140  INPUTTAB(20,7),P:INPUTTAB(27,7),D
  150  PROCENV(E)
  160  SOUND C,1,P,D
  170  PRINTTAB(0,16);"SOUND ";C;",";E;",";P;",";D;
SPC(100);
  180  PRINTTAB(0,7);SPC(40);TAB(0,15);"ENVELOPE ";
E;SPC(5);:VDU30
```

```
  190  GOTO 90
  200  DEFPROCENV(X)
  210  ON X GOTO 230,250,270,290,310,330,350,370
  220  GOTO 380
  230  ENVELOPE 1,1,-15,-15,-15,230,230,230,126,0,0
,-126,126,126
  240  ENDPROC
  250  ENVELOPE 1,1,4,2,-2,5,5,5,126,0,0,-126,126,1
26
  260  ENDPROC
  270  ENVELOPE1,1,6,0,-6,200,100,200,126,0,0,-126,
126,126
  280  ENDPROC
  290  ENVELOPE 1,1,-50,-50,-50,20,-20,20,126,0,0,-
126,126,126
  300  ENDPROC
  310  ENVELOPE 1,1,-100,100,-100,100,-100,100,126,
0,0,-126,126,126
  320  ENDPROC
  330  ENVELOPE 1,1,10,10,10,230,230,230,126,0,0,-1
26,126,126
  340  ENDPROC
  350  ENVELOPE 1,3,0,1,0,0,255,0,126,0,0,-126,126,
126
  360  ENDPROC
  370  ENVELOPE 1,3,0,1,0,0,255,0,126,0,0,-126,126,
126
  380  ENDPROC
  390 :
  400  REM Add Extra Envelopes Here.
  410  REM
  420  REM
  430  REM
  440 :
  450  ON ERROR OFF
  460  IF ERR<>17 REPORT:PRINT " at line ";ERL:END
  470  IF E=0 PRINT'''':END
  480  PRINTTAB(0,20);"LIST line ";(20*E)+210
  490  END
```

RESCUING A BAD PROGRAM
by C. Opie

One of the most dreaded error messages that your Electron will produce is ' BadProgram' .Once that has been displayed you will find that you can no longer access your current Basic program. Now we come to your rescue with a program that will restore your program to health with almost magical efficiency. It can also be used to recover programs that refuse to load correctly from cassette.

The error message ' BadProgram' is produced by Basic when it detects that the program in memory has been corrupted, or has been loaded incorrectly. Once the message has been issued, the user cannot directly modify his program in any way. LIST does not work and typing new lines has no effect. This can be disastrous if a small programming error results in a correupted program for which you have no backup copy on tape. The ' Rescue'program presented here will search through a ' BadProgram' for the corrupted parts, and fix these in such a way that the program can once again be listed and edited. This may not restore the original program completely but it does allow you to gain access to the program so that damaged lines can be deleted or replacd.

Using 'Rescue' to recover bad programs
First type it in, and save it on cassette. (SAVE "RESCUE"). It is reasonably short but must be typed in very accurately.

When you get a ' BadProgram' message type the following:

MODE 6 <return>
PAGE=TOP+&100 <return>
CHAIN "RESCUE" <return>

Changing the value of PAGE ensures that the ' Rescue' program occupies a separate area of memory to the corrupt program. The ' Rescue'program will then attempt to patch up the program currently resident in memory. When it finishes, you should be able to list and edit your repaired program as normal.

On recovering a corrupt program, save it on tape with SAVE"name". This will safeguard it against any further corruption. Having saved the program, list it out and examine the lines of code. Some lines may contain rubbish, whilst others may have disappeared completely, depending on how serious the corruption was. You may now re-type or replace lines as needed.

Something to look out for are ' #'characters appearing in the program after initial recovery. The rescue operation replaces any control characters in the program with the hash character (#).

Recovering cassette programs
Sometimes you will find it impossible to load correctly. Then you will be plagued with a variety of error messages often followed by the words ' Rewind Tape' .Trying to reload, will sometimes work but not always. This is how the ' Rescue'program can, once again, avert potential disaster.

Before attempting to re-load the offending program from cassette, type the following line into your Electron:

*OPT2,0 <return>

Now proceed to load the problem cassette as before. This time, although error messages may still appear, the program will continue to be loaded into the computer, including any corrupted sections. Once the corrupt program is in memory, follow the steps already described to restore as much of your program as possible. You should also, at this stage, restore cassette loading to its normal state by typing:

*OPT <return>

Of course, this process will generally work only with your own programs, as you will need to know how to make corrections yourself, once ' Rescue' has done its job. It will not generally help with programs that you have bought or obtained elsewhere, especially if the ' BadProgram' message arises as a result of copy protection put on the software by the manufacturer.

Some final hints

' Rescue' is an extremely effective utility, but it is still better to avoid losing your program in the first place. One way to reduce the chances of loss is to save your programs frequently onto cassette during development. Saving two copies will also help to avoid loading problems later. Moving back to a previous version is usually much easier than recovering a damaged program and should be quicker overall.

Explaining PAGE

Part of your computer' s memory is reserved for its own use. The point where your own programs tarts is always stored for reference in a special variable called PAGE. On the Electron, this is normally set to &E00 (3584 in decimal), when you first switch the computer on. To check, type PRINT PAGE <return>. PAGE can be set to other values for special purposes, usually so that a program can be located in a different part of memory. It is possible, in this way, to have more than one program in memory at the same time. This happens when the ' Rescue' program and your program are in memory together.

```
10 REM Rescue
20 REM by Colin Opie
30 REM BEEBUG
40 REM VERSION P 1.0
```

```
   50 :
   60 ON ERROR GOTO 320
   70 PROCinit
   80 REPEAT
   90 PROCrecover
  100 UNTIL finished
  110 PAGE=P%
  120 END
  130 :
  140 DEF PROCrecover
  150 ?line=&0D: lenpos=line+3: count=1
  160 IF ?line=&0D AND line?1=&FF THEN finished=TRU
E:ENDPROC
  170 PRINT (line?1)*256+(line?2);
  180 REPEAT
  190 IF line?count<>&0D THEN count=count+1
  200 IF line?count<>&0D AND line?count<32 AND coun
t>4 THEN line?count=35
  210 IF count>250 THEN line?(count+1)=&0D
  220 UNTIL line?count=&0D
  230 PRINT ~line
  240 line=line+count: ?lenpos=count
  250 ENDPROC
  260 :
  270 DEF PROCinit
  280 P%=&E00
  290 line=P%: line?1=0: finished=FALSE
  300 ENDPROC
  310 :
  320 ON ERROR OFF
  330 MODE 6:IF ERR=17 END
  340 REPORT:PRINT" at line ";ERL
  350 END
```

BAD PROGRAM LISTER
by E.Hanson

This program will list any Basic program (or any text file) stored in the computer, in a format similar to that produced by the LIST command. Its particular value is in listing programs which the command, LIST, will not control; i.e. programs whih have become ' damagedin the computer. In this respect it forms a useful adjunct to the program ' Rescue' In fact, ' Lister'will even list the remains of a program that has been partly overwritten when another shorter program has been loaded on top of it.

' Lister'works by looking at the Basic tokens as they are stored in memory, and converting them back to their original keyword format. The program is quite small, just under two blocks in length, as it uses the look-up table in the Basic ROM at &8071 to make the conversions.

LISTER will not convert the line numbers following a GOTO, as these are stored in code within the computer to save space. Note that for the sake of speed (and the irregularity of the way the tokens are set up in memory), the word "AND" is printed as "ND".

Using ' Lister'
Type in the program and save a copy before use, as you would any Basic program.

Asuming that you want to use ' Lister'to list a program already in the computer, you will need to change PAGE before you load in ' Lister'This avoids overwriting the program already resident. Proceed as follows:

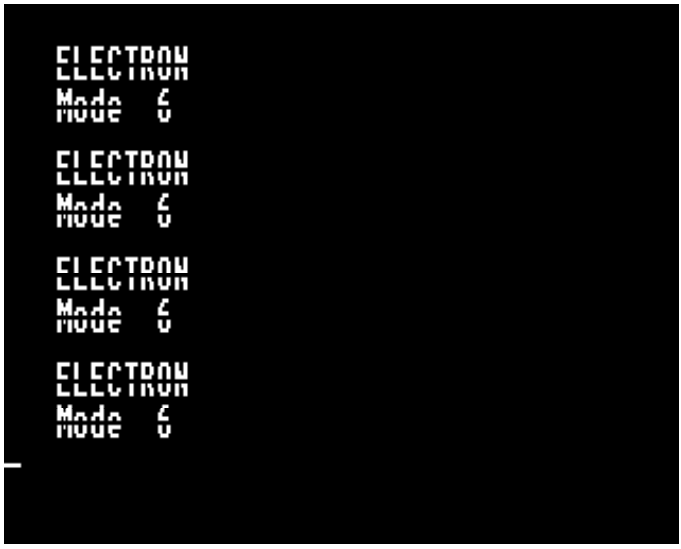
MODE6 <return>
PAGE=TOP+&100 <return>
CHAIN"LISTER" <return>

' Lister'will then run and print ' StartAddress?' . This will normally be &e00, but you can type in any address that you want. The listing will then start and will print in 1k blocks. After each block it will beep, and wait for you to press any key. It will then print the current address in hex followed by the next 1k of program, and so on until you stop the program by pressing ESCAPE.

```
   10 REM Bad Program Lister
   20 REM by Elizabeth Hanson
   30 REM BEEBUG
   40 REM VERSION P 1.0
   50 :
   60 MODE6
   70 INPUT'"Start Address",A$
   80 @%=1:J%=255:A%=EVAL(A$)
   90 REPEAT:PRINT'" ";~A%
  100 FORI%=0TOJ%:X%=A%?I%
  110 IFX%=&0D PRINT'A%?(I%+1)*256+A%?(I%+2);" ";:I
%=I%+3:GOTO140
  120 IFX%>&7F ANDX%<&FF PROCL:GOTO140
  130 IFX%>31PRINTCHR$(X%);ELSEPRINT~X%;
  140 NEXT
  150 A%=A%+J%+1:VDU7:B$=GET$:UNTILFALSE
  160 END
  170 :
  180 DEFPROCL:K%=-1:T%=&8071
  190 REPEAT:K%=K%+1:Y%=T%?K%:UNTILY%=X%ORK%>&300:L
%=K%
```

```
  200 REPEAT:K%=K%-1:Z%=T%?K%:UNTILZ%>&7F ORK%<0
  210 IFT%?(K%+2)<40K%=K%+3ELSEK%=K%+2
  220 REPEAT:PRINTCHR$(T%?K%);:K%=K%+1:UNTILK%>=L%:
ENDPROC
```

DOUBLE HEIGHT
by B. Grand



Enlarge your text displays with this useful utility that will really make your programs stand out.

Large size text on the screen can add impact to your programs, but with the Electron there is no simple facility for generating double height characters. To overcome this we present a procedure which allows double height printing in any mode. This is extremely useful for making more prominent displays, required, for example, in education and in games. The procedure, called PROCdouble, is defined in lines 10000 to 10130 and is very easy to use. Simply add it to the end of your program, and it can then be called as a normal procedure. For

example

PROCdouble("FORTY-TWO",5,10)

will display "FORTY-TWO" in double height characters, and in a position on the screen, 10 characters down from the top and 5 characters in form the lefthand side. The string can be string variable, andthe co-ordinates can also be variables.

We have put the procedure PROCdouble following a short program in lines 10 to 150 to show you how to use it. This program calls the procedure three times in order to display three lines of double-height text on the screen. Each time that you press the space bar the program re-runs in a different mode.

If you want to use this proceudre in one of your own programs, you are recommended to consult the Electron User Guide, pages 200 and 201, which tell you how merge Basic programs together.

```
   10  REM Double Height Text
   20  REM by B.Grand
   30  REM BEEBUG
   40  REM VERSION P 1.0
   50 :
   60  FOR mode=0 TO 6
   70  MODE mode
   80  FOR A=2 TO 17 STEP 5
   90  PROCdouble("ELECTRON",2,A)
  100  PROCdouble("Mode",2,A+2)
  110  PROCdouble(STR$(mode),8,A+2)
  120  NEXT
  130  WAIT=GET
  140  NEXT
  150  END
  160 :
10000  DEF PROCdouble(A$,K,L)
10010  REM Version E 1.0
10020  LOCAL N
10030  A%=&A:X%=0:Y%=&A
10040  D=&A00
10050  FORN=1 TO LEN(A$)
```

```
10060   B$=MID$(A$,N,1)
10070   ?D=ASC(B$)
10080   CALL (&FFF1)
10090   VDU23,240,D?1,D?1,D?2,D?2,D?3,D?3,D?4,D?4
10100   VDU23,241,D?5,D?5,D?6,D?6,D?7,D?7,D?8,D?8
10110   PRINT TAB(K+N,L);CHR$(240);TAB(K+N,L+1);CHR$
(241)
10120   NEXT N
10130   ENDPROC
>
```

3-D LETTERING
by G. Weston

This short program produces a 3-D effect on lettering used for headings, and considerably improves the look of the title page of programs in which it is incorporated. In this particular version the text is printed in white with a black shadow against a blue background.

To use the program for your own purposes,

you really only need the procedure PROCDOUBLE. This is defined in lines 1000 onwards. To call the procedure you need one or more lines of the kind appearing on lines 70 to 100. In each case three parameters follow the procedure call. The first, in quotation marks, is the text to be printed. The second two parameters give the x and y co-ordinates of the printing position on the screen. A little experiment is needed here - but remember that 0,0 is the bottom left hand corner, and 1279,1023 is the top right hand corner of the screen.

The technique used in this program is to print the text in white, and then to overprint it in black 8 positions down to the left (hence line 1070). Overprinting might be expected partially to erase the original text, but this is avoided by using the GCOL statement in line 1030 to perform so-called ' OR' plotting. See the User Guide p.153 for further details.

```
  10  REM 3-D Lettering
  20  REM by G.Weston
  30  REM BEEBUG
  40  REM VERSION P 1.0
  50 :
  60  MODE 5
  70  PROCDOUBLE("BEEBUG",450,800)
  80  PROCDOUBLE("for the",400,600)
  90  PROCDOUBLE("Acorn",450,400)
 100  PROCDOUBLE("Electron",360,200)
 110  END
 120 :
1000  DEFPROCDOUBLE(A$,X,Y)
1010  VDU5
1020  VDU19,0,4,0,0,0:VDU19,1,0,0,0,0
1030  GCOL1,3
1040  MOVE X,Y
1050  PRINT A$
1060  GCOL 1,1
1070  MOVE X-8,Y-8
1080  PRINT A$
1090  ENDPROC
```

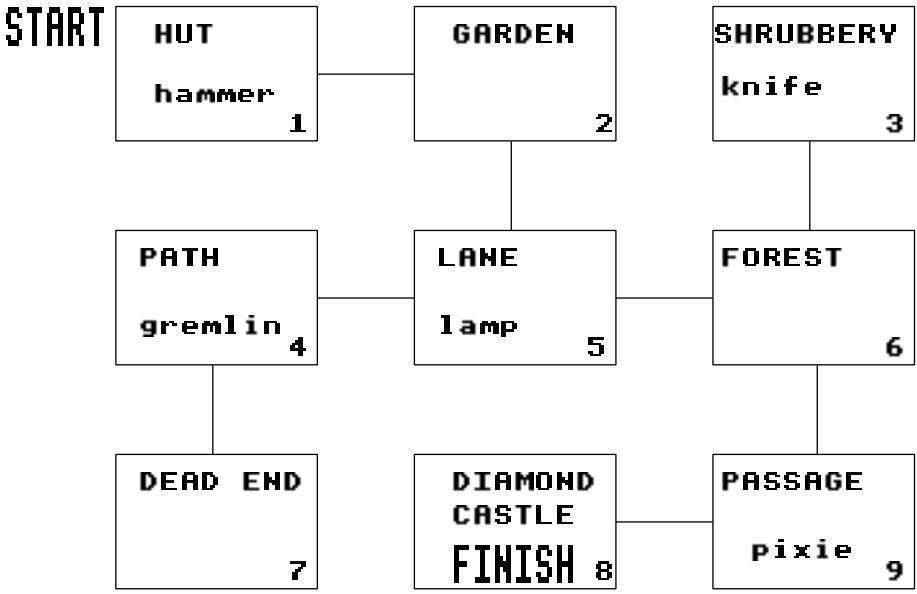


Fig 1 MAP OF RETURN OF THE DIAMOND

INTERESTED IN THE LATEST COMPUTER BOOKS AND SOFTWARE?

Penguin have many exciting future projects to share with you. There will be books on new models and machines, specific handbooks on graphics, sound and other functions, plus a terrific range of Penguin Software covering everything from arcade games to dieting!

We will keep you regularly in touch with the latest news. Just send your name, address and any special interest to:

Penguin Books Dept. CMD
536 Kings Road
London SW10 0UH

Getting the Most from Your . . .

This exciting new series provides comprehensive and carefully designed introductions to a whole range of machines. The books are compiled by professional writers and jouranlists and, through the use of diagrams, colour photographs, programs, entertaining examples and informative appendices, they take you, in a clear and painless way, from the elemtns of computing through to mastery of the machine.

If you want to understand what your particular computer is really capable of doing, then this new series has the right book for you.

Already published
Dragon 32
Sinclair Spectrum
Vic-20

Forthcoming
ZX 81
Commodore 64
BBC 'B'
Acorn Electron

The Acorn Guide to the Electron
Neil and Pat Cryer

The Acorn Electron (described in Which Micro? as ' a winner') is probably, for the price, the most advanced personal computer on the market. This guide, published with the full cooperation of the manufactuers, describes and explains everything a non-technical owner needs to know in order to get the most from this versatile and amazing new machine.

The Electron is designed to be fun, useful and, above all, the best introduction to the new Age of Computers. It has been developed by the people responsible for the BBC Micro -- the machine that is part of the syllabus of over 80 per cent of our schools. Both computers understand the same language and both have been designed to grow with your understanding of their capability and you needs.

You may be thinking of buying this book because you have just bought the new Acorn Eletron or maybe you have bought the new Acorn Electron or maybe you have bought the machine as a present for a friend or relative. Whichever is the case, The Acorn Guide to the Electron is the indispensable companion to your machine.