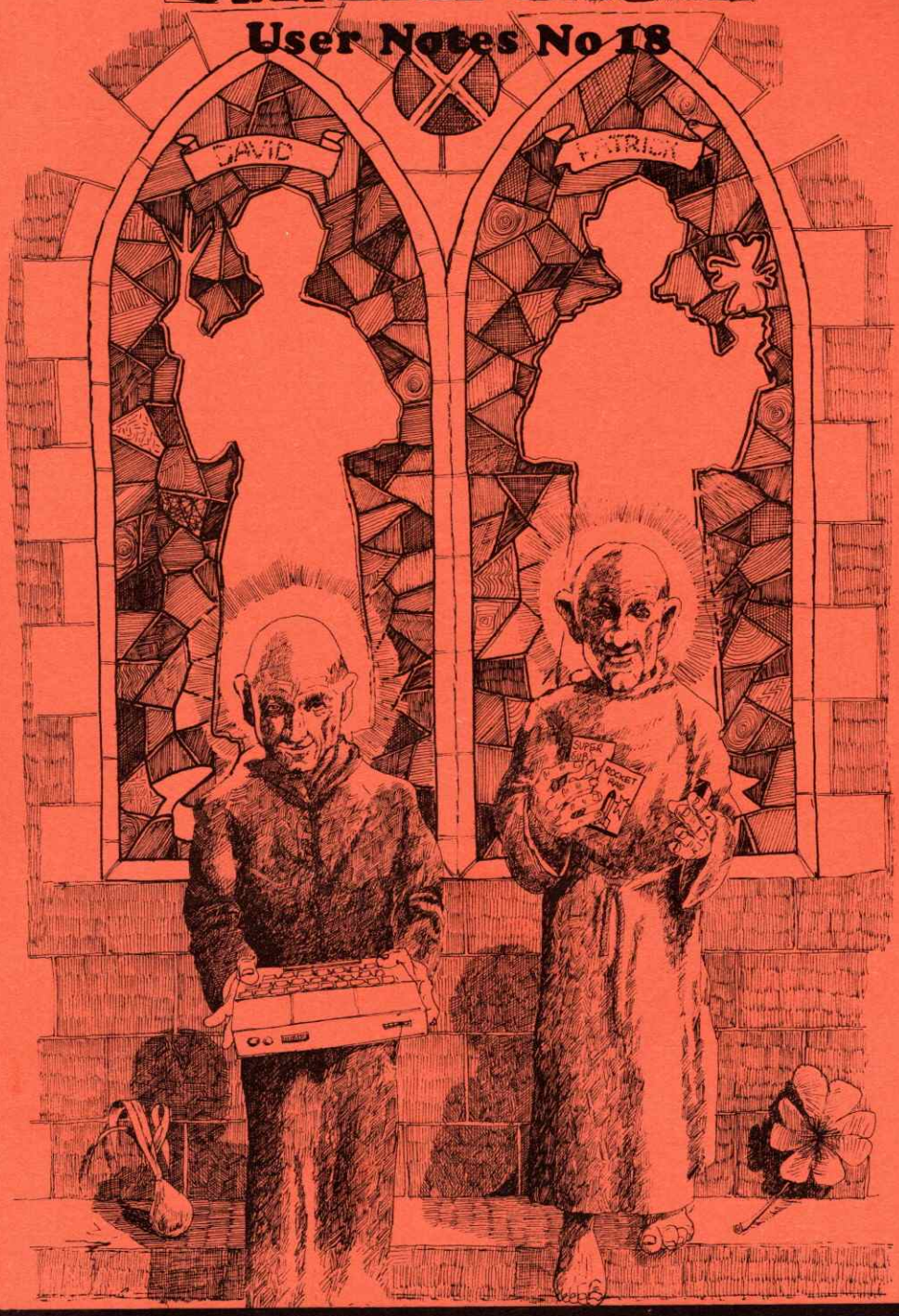
SHARPSOFT

User Notes No 18



SHARPSOFT USER NOTES

ISSUE NO:18

C O N T E N T S

SHARPSOFT USER NOTES

ISSUE NO:18

Editorial

This isssue brings us to the end of an era - we finally find that there is not enough new information from our subscribers to continue publishing the USER NOTES on a regular basis. Readers will probably realise that the number of subscribers has fallen to a level where it will, in the future, be no longer economic for us to publish three Issues a year. We will of course still be here, supporting your systems, supplying software and hardware add-ons, whilst still available. It is only the actual regular issues of the User Notes that are going into retirement. However, if in 1986 we find that over a period of months we can collect together enough information to publish one or more additional Issues, then all our subscribers on the mailing list will be informed by post.

As you know the MZ-700 User Notes are going strong and are indeed keeping the name S.U.N. alive. We also have plans to start a new Sharp Pocket Computer Notes, so any Sharp P.C. owners interested - get in touch.

Finally our thanks to all Sharp fans who have supported our efforts.

Best wishes,

MIKE BRINSON
and
THE SHARPSOFT TEAM

MACHINE CODE INPUT/OUTPUT SUBROUTINES

by R.F. Bateman

This package contains 34 subroutines and a small demonstration program and includes all the facilities you are likely to require to handle the V.D.U., printer and cassette unti in a straightforward manner. The accompanying source listings will allow you to modify the software, should this be necessary.

The software was produced with the aid of a ZEN Assembler, but can be easily modified for any other assembler since the source is supplied on cassette. For those users who work in machine code the listing contains the hex equivalent of the source. ZEN requires both the source and object code to be store resident during assembly and to ensure that this is possible for the 20K version of the MZ80K, the program has been split into three sections. The tape supplied has therefore four files recorded on it:-

1. IOPROCS/VPSOURCE

   This is the source code for all the VDU and printer subroutines.

2. IOPROCS/TSOURCE

   This is the source code for all the tape subroutines.

3. SYSTEM

   This contains the source code of a small demonstration program which illustrates the use of a few of the subroutines and may also prove useful in its own right. It consists of a command loop which allows programs to be loaded and run, converts two integers from decimal to hex or vice versa (and finds also the sum and difference) and allows the user to return to the monitor. In addition it provides a convenient place for a user program to return to in the event of a failure or at program termination and it illustrates also the method of handling the standard errors detected by the subroutine package.

4. IOPROCS/OBJ

   This contains the object code for 1, 2 and 3. The start address for SYSTEM and the subroutine entry points may be obtained from the listing and may, of course, be changed to suit the user's system. All the entries are defined using EQUs at the start of each section.

a. A number of printers have been connected to the MZ80K via non-standard interfaces and with different software requirements. It is unlikely, therefore, that any of the printer subroutines will work correctly on your system. In this event you must supply your own version of subroutine OUTP, which is the base subroutine used by all the other printer subroutines. OUTP simply outputs the character in the 'A' register to the printer following a status check. Please ensure that the registers are stacked and unstacked correctly.

b. All programs using these subroutines must load the stack pointer first.

c. All the standard errors detected by the subroutines are vectored to the address contained in the last word in store (i.e. 5FFE as delivered). The contents of 5FFE are preset to point at a dynamic stop loop to ensure that no damage results to the user's software as a result of the program running wild. To enable sensible error messages to be produced, and to regain control, the programmer must place the location of an error handler into 5FFE. This is illustrated in SYSTEM.

The following section describes the subroutines supplied.

NLV NLP NLT

Outputs a newline to the VDU, printer or tape. The registers remain unchanged.

SPV SPP SPT

Outputs a space to the VDU, printer or tape. The registers remain unchanged.

OUTV OUTP OUTT

Outputs a character from the 'A' register to the VDU, printer or tape. The registers are unchanged.

EOF

Outputs an END OF FILE character (value 255) to the tape. This will be detected on input. (See INTAPE, etc.) The registers are unaffected.

COPYV COPYP COPYT

Outputs a string of characters to the VDU, printer or tape. The address of the string must be loaded into the 'DE' register before the call and the string must be terminated

by a CR.
If a CR is required to trigger your printer, it will have to be supplied by a call of NLP. The registers, including 'DE', remain unchanged.

LINEV LINEP LINET

These are equivalent to a call of newline, followed by a copy. The registers are unaffected.

PRINTV PRINTP PRINTT

Prints an integer to the VDU, printer or tape in the range:-

-32768 to 32767 (or 8000H to 7FFFH)

Prior to the call, the integer to be printed must be loaded into the 'HL' register and the carry flag cleared to print a signed decimal and set to print a hex. integer. The registers are unaffected.

INKEY INTAPE

Inputs one character from the keys or tape into the 'A' register, waiting on the call until the character is available.
INKEY modifies certain characters as follows:-

DEL	returns	60H
INST	returns	61H
CAP	returns	62H
SMALL	returns	63H

SHIFT/BREAK causes the carry flag to be set on return from INKEY.
INTAPE will detect an END OF FILE character and set the carry fla.

STRINGKEY STRINGTAPE

Inputs a string of characters, terminated by a CR, into a buffer pointed at by 'DE' from the keys or tape. On exit 'DE' points at the start of the string, unlike the 'USER' call in ROM. The registers remain unchanged.

READKEY READTAPE

Reads an integer from keys or tape into the 'HL' register. All non-digits preceding the first decimal digit are ignored and the integer terminates on the first non-digit. The range of integers is:-

-32768 to 32767 (or 8000H to 7FFFH).

Characters other than a hexadecimal digit preceding an 'H' will cause an error. All registers other than 'HL' are unchanged.

OPENW

The tape file must be opened for writing before any calls of NLT, SPT, OUTT, COPYT, LINET or PRINTT. These calls transfer the appropriate characters into a user designated buffer area, the contents of which will be transferred to tape when the buffer is full or when an END OF FILE character is detected. Thus, data will be transferred in standard length records with inevitable stop/start gaps in between. To minimise the transfer time and also maximi tne amount of data that may be stored on a tape, the user should make the buffer as long as possible. Although any length of buffer is acceptable, including 1, it is recommended that at least 100 bytes should be allocated for this purpose.

Load 'DE' with the start of the buffer area and 'BC' with its length preceding the call. During the call a prompt will ask for the file name (up to 16 characters) and the user may then type a filename, terminated by a CR, or CR. Following the subsequent 'RECORD-PLAY' prompt the file header in the form of the filename (or CR) and the file parameters will be written to tape.
The registers are unaffected.

OPENR

This is required before any call of INTAPE, STRINGTAPE or READTAPE. These subroutines use the same buffer area designated by the OPENW call which generated the file. OPENR prompts for a filename (or CR) and then searches th tape for the filename (or takes the first file header it finds if a CR was typed) and extracts the file parameters from the header. The registers are unchanged.

READFILE

Opens and reads a complete source file from tape into store at the location specified by the header. The user will be prompted for a file name. No registers are affected.

WRITEFILE

Opens and writes a complete source file to tape. Enter with the start location in 'DE' and file length in 'BC'. A prompt will request the filename. The registers are unchanged.

VERIFYFILE

Opens, reads and checks a tape file against a copy held in store. See READFILE for conditions of use.

WRITEOBJ

Opens and writes an object file to tape. The user will be prompted for the start address in store, the last address in store and the execution address. This will be followed by a prompt for the filename.

READOBJ

Opens and reads an object file from tape. The code may then be executed by:-

```
LD HL,(1106H)
JP (HL)
```

VERIFYOBJ

Opens, reads and checks an object file from tape against a copy held in store.

DEBUG

Prints the contents of all the registers on the printer. It may be easily modified to use the VDU by changing the calls of LINEP, NLP and PRINTP to LINEV, NLV and PRINTV. It would also be sensible to incorporate a temporary halt to prevent scrolling by adding a call of INKEY.
The print format is:-

```
PC AF BC DE HL IX IY   = HEX INTEGERS
SP AF' BC' DE' HL' IR' = HEX INTEGERS
```

```
1                       ;SYSTEM.   **DEMONSTRATION PROGRAM*
2
3                                  ORG  58EAH
4                                  LOAD 58EAH
5
6
7                       CR:        EQU  13
8                       EXEC:      EQU  1106H
9                       OUTV:      EQU  5CBAH
10                      NLV:       EQU  5CC4H
11                      NLP:       EQU  5CCCH
12                      SPV:       EQU  5CD4H
13                      SPP:       EQU  5CDCH
14                      LINEV:     EQU  5CE4H
15                      LINEP:     EQU  5CEBH
16                      COPYV:     EQU  5CF2H
17                      COPYP:     EQU  5D02H
18                      PRINTV:    EQU  5D12H
19                      PRINTP:    EQU  5D1BH
20                      OUTP:      EQU  5DE3H
21                      INKEY:     EQU  5E2FH
22                      STRINGKEY: EQU  5E59H
23                      READKEY:   EQU  5E67H
24                      DEBUG:     EQU  5F20H
25                      OUTT:      EQU  5A7FH
26                      INTAPE:    EQU  5AB4H
27                      NLT:       EQU  5AEEH
28                      SPT:       EQU  5AF6H
29                      EOF:       EQU  5AFEH
30                      LINET:     EQU  5B06H
31                      COPYT:     EQU  5B0DH
32                      PRINTT:    EQU  5B1BH
33                      STRINGTAPE: EQU 5B24H
34                      READTAPE:  EQU  5B3AH
35                      OPENR:     EQU  5B47H
36                      OPENW:     EQU  5BD8H
37                      READFILE:  EQU  5C30H
38                      VERIFYFILE: EQU 5C3AH
39                      WRITEFILE: EQU  5C44H
40                      WRITEOBJ:  EQU  5C52H
41                      READOBJ:   EQU  5CB2H
42                      VERIFYOBJ: EQU  5CB6H
43
44
45  58EA 31F010                    LD   SP,10F0H
46  58ED 21D959                    LD   HL,ERROR          ; ERROR RE
47  58F0 22FE5F                    LD   (5FFEH),HL
48  58F3 111559        REPT:       LD   DE,SMESS
49  58F6 CDE45C                    CALL LINEV
50  58F9 113159                    LD   DE,SMESS1
51  58FC CDE45C                    CALL LINEV
52  58FF CD2F5E                    CALL INKEY
53  5902 FE4C                      CP   'L'
54  5904 283C                      JR   Z,LOAD
55  5906 FE52                      CP   'R'
56  5908 283D                      JR   Z,RUN
57  590A FE44                      CP   'D'
58  590C 2840                      JR   Z,DECHEX
59  590E FE4B                      CP   'K'
60  5910 CA0000                    JP   Z,0
```

```
 51 5913 18DE                    JR   REPT
 62
 63 5915 4C4F4144 SMESS:         DB   'LOAD,RUN,D/H CONV
 63 5919 2C52554E
 63 591D 2C442F48
 63 5921 20434F4E
 63 5925 562E204F
 63 5929 52
 64 592A 204B494C                DB   ' KILL.',CR
 64 592E 4C2E0D
 65
 66 5931 54595045 SMESS1:        DB   'TYPE L,R,D OR K)'
 66 5935 204C2C52
 66 5939 2C44204F
 66 593D 52204B3E
 66 5941 0D
 67
 68 5942 CDB25C   LOAD:          CALL READOBJ
 69 5945 18AC                    JR   REPT
 70
 71 5947 CDC45C   RUN:           CALL NLV
 72 594A 2A0611                  LD   HL,(EXEC)
 73 594D E9                      JP   (HL)
 74
 75
 76 594E 11AC59   DECHEX:        LD   DE,N1MESS
 77 5951 CDE45C                  CALL LINEV
 78 5954 CD675E                  CALL READKEY
 79 5957 22D559                  LD   (N1),HL
 80 595A B7                      OR   A
 81 595B CD9A59                  CALL PRNT
 82 595E CDC45C                  CALL NLV
 83 5961 11B559                  LD   DE,N2MESS
 84 5964 CDF25C                  CALL COPYV
 85 5967 CD675E                  CALL READKEY
 86 596A 22D759                  LD   (N2),HL
 87 596D CD9A59                  CALL PRNT
 88 5970 CDC45C                  CALL NLV
 89 5973 11BE59                  LD   DE,SUM
 90 5976 CDF25C                  CALL COPYV
 91 5979 2AD559                  LD   HL,(N1)
 92 597C ED5BD759                LD   DE,(N2)
 93 5980 19                      ADD  HL,DE
 94 5981 CD9A59                  CALL PRNT
 95 5984 11C459                  LD   DE,DIFF
 96 5987 CDE45C                  CALL LINEV
 97 598A 2AD559                  LD   HL,(N1)
 98 598D ED5BD759                LD   DE,(N2)
 99 5991 B7                      OR   A
100 5992 ED52                    SBC  HL,DE
101 5994 CD9A59                  CALL PRNT
102 5997 C3F358                  JP   REPT
103
104 599A F5          PRNT:       PUSH AF
105 599B B7                      OR   A
106 599C CD125D                  CALL PRINTV
107 599F 37                      SCF
108 59A0 CD125D                  CALL PRINTV
109 59A3 F1                      POP  AF
110 59A4 D0                      RET  NC
```

```
111 59A5 11CA59              LD   DE,CARRY
112 59A8 CDF25C              CALL COPYV
113 59AB C9                  RET
114
115 59AC 4E554D42 N1MESS:    DB   'NUMBER1=',CR
115 59B0 4552313D
115 59B4 0D
116 59B5 4E554D42 N2MESS:    DB   'NUMBER2=',CR
116 59B9 4552323D
116 59BD 0D
117 59BE 53554D20 SUM:       DB   'SUM =',CR
117 59C2 3D0D
118 59C4 44494646 DIFF:      DB   'DIFF=',CR
118 59C8 3D0D
119 59CA 20434152 CARRY:     DB   ' CARRY SET',CR
119 59CE 52592053
119 59D2 45540D
120
121 59D5 0000     N1:        DW   0
122 59D7 0000     N2:        DW   0
123
124
125 59D9 87       ERROR:     ADD  A,A                ; ERROR NOS. 0-7
126 59DA 5F                  LD   E,A
127 59DB 1600                LD   D,0
128 59DD 21ED59              LD   HL,MES
129 59E0 19                  ADD  HL,DE
130 59E1 5E                  LD   E,(HL)
131 59E2 23                  INC  HL
132 59E3 56                  LD   D,(HL)
133 59E4 CDE45C              CALL LINEV
134 59E7 CDC45C              CALL NLV
135 59EA C3F358              JP   REPT
136
137 59ED FB59     MES:       DW   M0
138 59EF 0D5A                DW   M1
139 59F1 1E5A                DW   M2
140 59F3 2F5A                DW   M3
141 59F5 4F5A                DW   M4
142 59F7 5A5A                DW   M5
143 59F9 6B5A                DW   M6
144
145
146 59FB 54415045 M0:        DB   'TAPE WRITE ERROR.
146 59FF 20575249
146 5A03 54452045
146 5A07 52524F52
146 5A0B 2E0D
147 5A0D 54415045 M1:        DB   'TAPE READ ERROR.'
147 5A11 20524541
147 5A15 44204552
147 5A19 524F522E
147 5A1D 0D
148 5A1E 4E4F5420 M2:        DB   'NOT A HEX DIGIT.'
148 5A22 41204845
148 5A26 58204449
148 5A2A 4749542E
148 5A2E 0D
149 5A2F 494E5445 M3:        DB   'INTEGER IS ) 3276
149 5A33 47455220
```

```
149 5A37 4953203E
149 5A3B 20333237
149 5A3F 3637204F
149 5A43 5220
150 5A45 3C202D33              DB   '( -32768.',CR
150 5A49 32373638
150 5A4D 2E0D
151 5A4F 42524541 M4:          DB   'BREAK KEY.',CR
151 5A53 4B204B45
151 5A57 592E0D
152 5A5A 53554D20 M5:          DB   'SUM CHECK ERROR.'
152 5A5E 43484543
152 5A62 4B204552
152 5A66 524F522E
152 5A6A 0D
153 5A6B 48454144 M6:          DB   'HEADER WRITE ERRO
153 5A6F 45522057
153 5A73 52495445
153 5A77 20455252
153 5A7B 4F522E0D
154
155                            END
```

```
1
2
3       ;*********************************
4       ;*                               *
5       ;*          MZ80K                *
6       ;*       TAPE CASSETTE           *
7       ;*       INPUT/OUTPUT            *
8       ;*       SUBROUTINES.            *
9       ;*                               *
10      ;*     COPYRIGHT 1981            *
11      ;*    BATEMAN SOFTWARE           *
12      ;*                               *
13      ;*********************************
14
15                ORG  05A7FH
16                LOAD 05A7FH
17
18
19      ;*********************************
20      ;*                               *
21      ;*           REMEMBER!           *
22      ;*           ********            *
23      ;*                               *
24      ;*      SET STACK POINTER        *
25      ;* BEFORE CALLING ROUTINES!      *
26      ;*                               *
27      ;*           ********            *
28      ;*                               *
29      ;* SET ERROR REPORT ADDRESS IN   *
30      ;* LAST WORD OF STORE AND ADD    *
31      ;* AN ERROR ROUTINE IN YOUR      *
32      ;* MAIN PROGRAM.                 *
33      ;*                               *
34      ;*********************************
35
36
37
38      CR:           EQU  13
39      SPC:          EQU  32
40      MESSAGEBUFF:  EQU  11A3H
41      PHEAD:        EQU  436H
42      PDATA:        EQU  475H
43      LHEAD:        EQU  4D8H
44      LDATA:        EQU  4F8H
45      CHECK:        EQU  588H
46      TAPECONT:     EQU  10F1H
47      LENGTH:       EQU  1102H
48      START:        EQU  1104H
49      EXEC:         EQU  1106H
50      MATCH:        EQU  180H
51      BREAK:        EQU  1EH
52      STRINGKEY:    EQU  05E59H
53      LINEV:        EQU  05CE4H
54      COPYV:        EQU  05CF2H
55      READKEY:      EQU  05E67H
56      PRINT:        EQU  05D24H
57      ZMB:          EQU  05E74H
58      READIN:       EQU  05E82H
59      LASTWORD:     EQU  05FFEH
60
```

```
                ;*********************************
                ;*                               *
                ;* START OF SUBROUTINES.         *
                ;*                               *
                ;*********************************



                ;*********************************
                ;*                               *
                ;* OUTPUT ONE CHARACTER FROM 'A' *
                ;* TO TAPE. REGISTERS UNCHANGED  *
                ;* ON EXIT. THE TAPE BUFFER IS   *
                ;* OUTPUT IF 'END OF FILE'       *
                ;* CHARACTER(255) OR BUFFER FULL.*
                ;*                               *
                ;*********************************


5A7F D5         OUTT:       PUSH DE
5A80 E5                     PUSH HL
5A81 C5                     PUSH BC
5A82 F5                     PUSH AF
5A83 ED5BE55F               LD   DE,(OUTB)
5A87 12                     LD   (DE),A
5A88 13                     INC  DE
5A89 ED53E55F               LD   (OUTB),DE
5A8D B7                     OR   A
5A8E 2AE75F                 LD   HL,(MAXOUTB)
5A91 ED52                   SBC  HL,DE
5A93 2804                   JR   Z,BOUT
5A95 FEFF                   CP   255                ; EOF
5A97 2011                   JR   NZ,TOUT
5A99 ED5B0411 BOUT:         LD   DE,(START)
5A9D ED53E55F               LD   (OUTB),DE
5AA1 ED4B0211               LD   BC,(LENGTH)
5AA5 CD7504                 CALL PDATA
5AA8 3805                   JR   C,OTF
5AAA F1         TOUT:       POP  AF
5AAB C1                     POP  BC
5AAC E1                     POP  HL
5AAD D1                     POP  DE
5AAE C9                     RET

5AAF AF         OTF:        XOR  A                  ; ERROR 0
5AB0 2AFE5F     ERROR:      LD   HL,(LASTWORD)
5AB3 E9                     JP   (HL)



                ;*********************************
                ;*                               *
                ;* INPUT ONE CHARACTER FROM TAPE*
                ;* TO 'A'. THE BUFFER IS LOADED *
                ;* WHEN EMPTY. THE CARRY FLAG IS*
                ;* SET WHEN 'END OF FILE'(255)   *
                ;* IS FOUND. 'BC','DE', AND 'HL'*
                ;* ARE UNCHANGED ON EXIT.        *
```

```
121                     ;*                                  *
122                     ;************************************
123
124
125
126 5AB4 D5          INTAPE:     PUSH DE
127 5AB5 E5                     PUSH HL
128 5AB6 C5                     PUSH BC
129 5AB7 ED5BE95F               LD   DE,(INB)
130 5ABB 2AEB5F                 LD   HL,(MAXINB)
131 5ABE B7                     OR   A
132 5ABF ED52                   SBC  HL,DE
133 5AC1 2014                   JR   NZ,INT1
134 5AC3 ED5B0411               LD   DE,(START)
135 5AC7 ED53E95F               LD   (INB),DE
136 5ACB D5                     PUSH DE
137 5ACC ED4B0211               LD   BC,(LENGTH)
138 5AD0 CDF804                 CALL LDATA
139 5AD3 DAE95A                 JP   C,ITF
140 5AD6 D1                     POP  DE
141 5AD7 1A          INT1:      LD   A,(DE)
142 5AD8 13                     INC  DE
143 5AD9 ED53E95F               LD   (INB),DE
144 5ADD FEFF                   CP   255                    ; EOF
145 5ADF 2803                   JR   Z,REOF
146 5AE1 B7                     OR   A
147 5AE2 1801                   JR   REOF1
148
149 5AE4 37          REOF:      SCF
150 5AE5 C1          REOF1:     POP  BC
151 5AE6 E1                     POP  HL
152 5AE7 D1                     POP  DE
153 5AE8 C9                     RET
154
155 5AE9 3E01        ITF:       LD   A,1                    ; ERROR 1
156 5AEB C3B05A                 JP   ERROR
157
158
159
160
161                     ;************************************
162                     ;*                                  *
163                     ;* OUTPUT NEWLINE TO TAPE. THE      *
164                     ;* REGISTERS ARE UNCHANGED.         *
165                     ;*                                  *
166                     ;************************************
167
168
169
170 5AEE F5          NLT:        PUSH AF
171 5AEF 3E0D                   LD   A,CR
172 5AF1 CD7F5A                 CALL OUTT
173 5AF4 F1                     POP  AF
174 5AF5 C9                     RET
175
176
177                     ;************************************
178                     ;*                                  *
179                     ;* OUTPUT SPACE TO TAPE.            *
180                     ;* THE REGISTERS ARE UNCHANGED      *
```

```
181                     ;* ON EXIT.                          *
182                     ;*                                   *
183                     ;************************************
184
185
186
187 5AF6 F5          SPT:        PUSH AF
188 5AF7 3E20                    LD   A,SPC
189 5AF9 CD7F5A                  CALL OUTT
190 5AFC F1                      POP  AF
191 5AFD C9                      RET
192
193
194                     ;************************************
195                     ;*                                   *
196                     ;* OUTPUT 'END OF FILE' TO TAPE*
197                     ;* THE REGISTERS ARE UNCHANGED *
198                     ;* ON EXIT. ( 'EOF' IS 255 )    *
199                     ;*                                   *
200                     ;************************************
201
202
203
204 5AFE F5          EOF:        PUSH AF
205 5AFF 3EFF                    LD   A,255
206 5B01 CD7F5A                  CALL OUTT
207 5B04 F1                      POP  AF
208 5B05 C9                      RET
209
210
211                     ;************************************
212                     ;*                                   *
213                     ;* OUTPUT NEWLINE, FOLLOWED BY    *
214                     ;* THE STRING POINTED AT BY 'DE' *
215                     ;* TO TAPE.                          *
216                     ;*                                   *
217                     ;************************************
218
219
220 5B06 CDEE5A      LINET:      CALL NLT
221 5B09 CD0D5B                  CALL COPYT
222 5B0C C9                      RET
223
224
225
226
227                     ;************************************
228                     ;*                                   *
229                     ;* COPY STRING TO TAPE.              *
230                     ;* 'DE' POINTS AT STRING, WHICH   *
231                     ;* MUST BE TERMINATED WITH A 'CR'*
232                     ;* REGISTERS UNCHANGED ON EXIT.   *
233                     ;* THE CR IS OUTPUT TO TAPE.      *
234                     ;*                                   *
235                     ;************************************
236
237
238
239 5B0D F5          COPYT:      PUSH AF
240 5B0E D5                      PUSH DE
```

```
241 5B0F 1A       COPYT1:     LD   A,(DE)
242 5B10 13                   INC  DE
243 5B11 CD7F5A               CALL OUTT
244 5B14 FE0D                 CP   CR
245 5B16 20F7                 JR   NZ,COPYT1
246 5B18 D1                   POP  DE
247 5B19 F1                   POP  AF
248 5B1A C9                   RET
249
250
251               ;**********************************
252               ;*                                *
253               ;* PRINT INTEGER TO TAPE.         *
254               ;* ENTER WITH INTEGER IN HL.      *
255               ;* ENTER WITH CARRY CLEAR TO      *
256               ;* PRINT DECIMAL AND CARRY        *
257               ;* SET TO PRINT HEX.              *
258               ;* REGISTERS RESTORED ON EXIT.    *
259               ;*                                *
260               ;**********************************
261
262
263
264 5B1B D5        PRINTT:     PUSH DE
265 5B1C CD245D               CALL PRINT
266 5B1F CD0D5B               CALL COPYT
267 5B22 D1                   POP  DE
268 5B23 C9                   RET
269
270
271               ;**************************************
272               ;*                                    *
273               ;* INPUT STRING FROM TAPE.            *
274               ;* ON ENTRY, 'DE' POINTS AT           *
275               ;* STRING WHICH MUST END WITH A       *
276               ;* 'CR'. REGISTERS UNCHANGED ON       *
277               ;* EXIT. WHEN 'END OF FILE' IS        *
278               ;* READ IT IS REPLACED BY 'CR'        *
279               ;* AND THE CARRY FLAG IS SET.         *
280               ;*                                    *
281               ;**************************************
282
283
284
285 5B24 B7        STRINGTAPE: OR   A                    ; CLEAR CARRY
286 5B25 D5                   PUSH DE
287 5B26 F5                   PUSH AF
288 5B27 CDB45A   ST1:        CALL INTAPE
289 5B2A 3005                 JR   NC,ST2
290 5B2C F1                   POP  AF
291 5B2D 37                   SCF                     ; SET CARRY FOR E
292 5B2E F5                   PUSH AF
293 5B2F 3E0D                 LD   A,CR               ; REPLACE EOF WIT
294 5B31 12       ST2:        LD   (DE),A
295 5B32 13                   INC  DE
296 5B33 FE0D                 CP   CR
297 5B35 20F0                 JR   NZ,ST1
298 5B37 F1                   POP  AF
299 5B38 D1                   POP  DE
300 5B39 C9                   RET
```

```
301
302
303
304                     ;**********************************
305                     ;*                                *
306                     ;* READ INTEGER FROM TAPE.        *
307                     ;* (-32768(=INT(=32767 OR         *
308                     ;*  8000H(=INT(=8FFFH).           *
309                     ;* INTEGER IN 'HL' ON EXIT. ALL   *
310                     ;* OTHER REGISTERS REMAIN         *
311                     ;* UNCHANGED.                     *
312                     ;*                                *
313                     ;**********************************
314
315
316
317 5B3A 11A711    READTAPE:   LD   DE,MESSAGEBUFF+4
318 5B3D CD245B                CALL STRINGTAPE
319 5B40 CD745E                CALL ZMB
320 5B43 CD825E                CALL READIN
321 5B46 C9                    RET
322
323
324
325                     ;**********************************
326                     ;*                                *
327                     ;* OPEN FILE FOR READING          *
328                     ;*                                *
329                     ;* BUFFER ADDRESS,SIZE AND        *
330                     ;* EXECUTION ADDRESS ARE READ     *
331                     ;* FROM THE FILE HEADER.          *
332                     ;* THE USER MUST RESERVE A        *
333                     ;* SUITABLE BUFFER OF THE SAME    *
334                     ;* SIZE THAT WAS USED WHEN THE    *
335                     ;* FILE WAS WRITTEN.              *
336                     ;* READ NAME FROM KEYS            *
337                     ;* BUFFER SIZE READ FROM TAPE.    *
338                     ;* BUFFER CHAR. COUNT SET TO MAX  *
339                     ;* IN 'INB'                       *
340                     ;* END OF BUFFER+1 SET IN         *
341                     ;* 'MAXINB'                       *
342                     ;*                                *
343                     ;**********************************
344
345
346
347 5B47 E5        OPENR:      PUSH HL
348 5B48 C5                    PUSH BC
349 5B49 F5                    PUSH AF
350 5B4A D5                    PUSH DE
351 5B4B 11965B                LD   DE,MOF
352 5B4E CDE45C                CALL LINEV
353 5B51 11A311                LD   DE,MESSAGEBUFF
354 5B54 CD595E                CALL STRINGKEY          ; INPUT FILE NAME
355 5B57 1A                    LD   A,(DE)
356 5B58 FE0D                  CP   CR                 ; NO FILE NAME?
357 5B5A 2005                  JR   NZ,OR1
358 5B5C CDAB5B                CALL FHEAD
359 5B5F 180D                  JR   OR2
360 5B61 CDAB5B    OR1:        CALL FHEAD
```

```
361 5B64 21F110              LD   HL,TAPECONT
362 5B67 0610                LD   B,16
363 5B69 CD8001              CALL MATCH              ; CHECK FILE NAME
364 5B6C 20F3                JR   NZ,OR1
365 5B6E 2A0211    OR2:      LD   HL,(LENGTH)
366 5B71 118D5B              LD   DE,RLOAD
367 5B74 CDE45C              CALL LINEV
368 5B77 11F110              LD   DE,TAPECONT
369 5B7A CDF25C              CALL COPYV
370 5B7D ED5B0411            LD   DE,(START)
371 5B81 19                  ADD  HL,DE
372 5B82 22EB5F              LD   (MAXINB),HL
373 5B85 22E95F              LD   (INB),HL
374 5B88 D1                  POP  DE
375 5B89 F1                  POP  AF
376 5B8A C1                  POP  BC
377 5B8B E1                  POP  HL
378 5B8C C9                  RET
379
380 5B8D 4C4F4144 RLOAD:     DB   'LOADING ',CR
380 5B91 494E4720
380 5B95 0D
381 5B96 54595045 MOF:       DB   'TYPE INPUT FILE N
381 5B9A 20494E50
381 5B9E 55542046
381 5BA2 494C4520
381 5BA6 4E414D45
381 5BAA 0D
382
383
384                ;**********************************
385                ;*                                *
386                ;* INPUT AND CHECK FILE HEADER.   *
387                ;*                                *
388                ;**********************************
389
390
391
392 5BAB D5        FHEAD:    PUSH DE
393 5BAC CDD804              CALL LHEAD              ; LOAD HEADER
394 5BAF 3814                JR   C,FERR             ; SUM CHECK ERROR
395 5BB1 11D15B              LD   DE,FMESS
396 5BB4 CDE45C              CALL LINEV
397 5BB7 11F110              LD   DE,TAPECONT
398 5BBA 211000              LD   HL,16
399 5BBD 19                  ADD  HL,DE
400 5BBE 360D                LD   (HL),CR            ; LIMIT NAME TO 16
401 5BC0 CDF25C              CALL COPYV              ; PRINT FILE NAME
402 5BC3 D1                  POP  DE
403 5BC4 C9                  RET
404
405
406 5BC5 FE02      FERR:     CP   2                  ; KEY PRESS?
407 5BC7 3E04                LD   A,4                ; ERROR 4. BREAK KE
408 5BC9 CAB05A              JP   Z,ERROR
409 5BCC 3E05                LD   A,5                ; ERROR 5. SUM CHEC
410 5BCE C3B05A              JP   ERROR
411
412
413 5BD1 464F554E FMESS:     DB   'FOUND ',CR
```

```
413 5BD5 44200D
414
415
416
417                ;**********************************
418                ;*                                *
419                ;* OPEN FOR WRITING.              *
420                ;*                                *
421                ;* READ NAME FROM KEYS.           *
422                ;* ENTER WITH BUFFER ADDRES IN DE*
423                ;* AND LENGTH IN BC. REGISTERS    *
424                ;* UNCHANGED ON EXIT.             *
425                ;*                                *
426                ;**********************************
427
428
429
430 5BD8 E5        OPENW:      PUSH HL
431 5BD9 F5                    PUSH AF
432 5BDA D5                    PUSH DE
433 5BDB C5                    PUSH BC
434 5BDC 11195C                LD   DE,WM
435 5BDF CDE45C                CALL LINEV
436 5BE2 11A311                LD   DE,MESSAGEBUFF
437 5BE5 CD595E                CALL STRINGKEY
438 5BE8 211000                LD   HL,16
439 5BEB 19                    ADD  HL,DE
440 5BEC 360D                  LD   (HL),CR            ; LIMIT NAME TO 16
441 5BEE 21F110                LD   HL,TAPECONT
442 5BF1 EB                    EX   DE,HL
443 5BF2 011000                LD   BC,16
444 5BF5 EDB0                  LDIR                    ; TRANSFER FILE NAME
445 5BF7 E1                    POP  HL                 ; LENGTH
446 5BF8 220211                LD   (LENGTH),HL
447 5BFB D1                    POP  DE                 ; BUFFER ADDRESS
448 5BFC D5                    PUSH DE
449 5BFD E5                    PUSH HL
450 5BFE ED530411              LD   (START),DE
451 5C02 ED53E55F              LD   (OUTB),DE
452 5C06 19                    ADD  HL,DE
453 5C07 22E75F                LD   (MAXOUTB),HL
454 5C0A CD3604                CALL PHEAD              ; OUTPUT HEADER
455 5C0D 3805                  JR   C,WERR
456 5C0F C1                    POP  BC
457 5C10 D1                    POP  DE
458 5C11 F1                    POP  AF
459 5C12 E1                    POP  HL
460 5C13 C9                    RET
461
462
463
464 5C14 3E06      WERR:       LD   A,6                ; ERROR 6. WRITE ERROR
465 5C16 C3B05A                JP   ERROR
466
467
468 5C19 54595045 WM:          DB   'TYPE OUTPUT FILE
468 5C1D 204F5554
468 5C21 50555420
468 5C25 46494C45
468 5C29 204E414D
```

```
468 5C2D 452E0D
469
470
471
472                 ;**********************************
473                 ;*                                *
474                 ;* READ FILE, WRITE FILE,         *
475                 ;* VERIFY FILE, READOBJ,          *
476                 ;* WRITEOBJ, VERIFYOBJ.           *
477                 ;*                                *
478                 ;**********************************
479
480
481                 ;**********************************
482                 ;*                                *
483                 ;* READFILE FROM TAPE.            *
484                 ;* SEE OPENR FOR CONDITIONS.      *
485                 ;*                                *
486                 ;**********************************
487
488
489
490 5C30 CD475B    READFILE:   CALL OPENR
491 5C33 CDF804                CALL LDATA
492 5C36 DAC55B                JP   C,FERR
493 5C39 C9                    RET
494
495
496                 ;**********************************
497                 ;*                                *
498                 ;* VERIFYFILE.                    *
499                 ;* READS AND CHECKS FILE          *
500                 ;* WITHOUT STORING. SEE OPENR     *
501                 ;* FOR INPUT CONDITIONS.          *
502                 ;*                                *
503                 ;**********************************
504
505
506
507 5C3A CD475B    VERIFYFILE: CALL OPENR
508 5C3D CD8805                CALL CHECK
509 5C40 DAC55B                JP   C,FERR
510 5C43 C9                    RET
511
512
513                 ;**********************************
514                 ;*                                *
515                 ;* WRITEFILE.                     *
516                 ;* WRITES FILE TO TAPE. SEE       *
517                 ;* OPENW FOR INPUT CONDITIONS.    *
518                 ;*                                *
519                 ;**********************************
520
521
522
523 5C44 D5        WRITEFILE:  PUSH DE
524 5C45 C5                    PUSH BC
525 5C46 CDD85B                CALL OPENW
526 5C49 CD7504                CALL PDATA
527 5C4C DA145C                JP   C,WERR
```

```
528 5C4F C1                     POP   BC
529 5C50 D1                     POP   DE
530 5C51 C9                     RET
531
532
533                 ;*******************************
534                 ;*                             *
535                 ;* WRITE OBJECT FILE.          *
536                 ;* PROMPT OUTPUT:-             *
537                 ;*         FROM? FFFFH         *
538                 ;*         TO? FFFFH           *
539                 ;*         EXECUTE? FFFFH      *
540                 ;*                             *
541                 ;*******************************
542
543
544
545 5C52 D5        WRITEOBJ:   PUSH DE
546 5C53 E5                     PUSH HL
547 5C54 11825C                 LD   DE,FROM
548 5C57 CDE45C                 CALL LINEV
549 5C5A CD675E                 CALL READKEY
550 5C5D E5                     PUSH HL
551 5C5E E5                     PUSH HL
552 5C5F 119A5C                 LD   DE,TO
553 5C62 CDE45C                 CALL LINEV
554 5C65 CD675E                 CALL READKEY
555 5C68 D1                     POP  DE
556 5C69 B7                     OR   A
557 5C6A ED52                   SBC  HL,DE
558 5C6C 23                     INC  HL
559 5C6D E5                     PUSH HL
560 5C6E 11A65C                 LD   DE,EX
561 5C71 CDE45C                 CALL LINEV
562 5C74 CD675E                 CALL READKEY
563 5C77 220611                 LD   (EXEC),HL
564 5C7A C1                     POP  BC
565 5C7B D1                     POP  DE
566 5C7C CD445C                 CALL WRITEFILE
567 5C7F E1                     POP  HL
568 5C80 D1                     POP  DE
569 5C81 C9                     RET
570
571 5C82 41444452 FROM:         DB   'ADDRESS OF OBJECT
571 5C86 45535320
571 5C8A 4F46204F
571 5C8E 424A4543
571 5C92 54204649
571 5C96 4C453D
572 5C99 0D                     DB   CR
573 5C9A 544F2041 TO:           DB   'TO ADDRESS ',CR
573 5C9E 44445245
573 5CA2 5353200D
574 5CA6 45584543 EX:           DB   'EXECUTE AT ',CR
574 5CAA 55544520
574 5CAE 4154200D
575
576
577
578                 ;*******************************
```

```
579                     ;*                                  *
580                     ;* READ OBJECT FILE FROM TAPE.       *
581                     ;* PARAMETERS READ FROM HEADER.      *
582                     ;*                                  *
583                     ;************************************
584
585
586
587 5CB2 CD305C    READOBJ:    CALL READFILE
588 5CB5 C9                     RET
589
590
591                     ;************************************
592                     ;*                                  *
593                     ;* VERIFY OBJECT FILE.              *
594                     ;*                                  *
595                     ;************************************
596
597
598
599 5CB6 CD3A5C    VERIFYOBJ:  CALL VERIFYFILE
600 5CB9 C9                     RET
601
602
603                     ;************************************
604                     ;*                                  *
605                     ;* GLOBAL WORKSPACE FOR ROUTINES*
606                     ;*                                  *
607                     ;************************************
608
609                             ORG  05FE2H
610                             LOAD 05FE2H
611
612
613
614
615 5FE2 00          FLAGP:      DB   0
616 5FE3 0000        INT:        DW   0
617 5FE5 0000        OUTB:       DW   0
618 5FE7 0000        MAXOUTB:    DW   0
619 5FE9 0000        INB:        DW   0
620 5FEB 0000        MAXINB:     DW   0
621 5FED 00          INFLAG:     DB   0
622 5FEE 00          ZERO:       DB   0
623 5FEF 00          RFLAG:      DB   0
624 5FF0 000000      DDAR:       DB   0,0,0
625 5FF3 00000000    DAR:        DB   0,0,0,0,0,0,0,0
625 5FF7 00000000
626 5FFB 0000        DARRAY:     DW   0
627 5FFD 00          SI:         DB   0
628
629 5FFE B05A                    DW   ERROR
630
631
632
633                             END
```

```
;*********************************
;*                               *
;*          MZ80K                *
;*      VDU AND PRINTER          *
;*        INPUT/OUTPUT           *
;*        SUBROUTINES.           *
;*                               *
;*     COPYRIGHT 1981            *
;*    BATEMAN SOFTWARE           *
;*                               *
;*********************************

            ORG  05CBAH
            LOAD 05CBAH


;*********************************
;*                               *
;*           REMEMBER!           *
;*           ********            *
;*                               *
;*      SET STACK POINTER        *
;* BEFORE CALLING ROUTINES!      *
;*                               *
;*           ********            *
;*                               *
;* SET ERROR REPORT ADDRESS IN   *
;* LAST WORD OF STORE AND ADD    *
;* AN ERROR ROUTINE IN YOUR      *
;* MAIN PROGRAM.                 *
;*                               *
;*********************************


CR:          EQU  13
SPC:         EQU  32
OUTVV:       EQU  18
DELAY:       EQU  759H
INK:         EQU  1BH
CONHEX:      EQU  410H
ASCI:        EQU  3DAH
MESSAGEBUFF:EQU  11A3H
BREAK:       EQU  1EH
LASTWORD:    EQU  05FFEH
```

```
                ;*********************************
                ;*                               *
                ;* START OF SUBROUTINES.         *
                ;*                               *
                ;*********************************


                ;*********************************
                ;*                               *
                ;* OUTPUT ONE CHARACTER FROM     *
                ;* 'A' TO THE VDU.               *
                ;* REGISTERS UNCHANGED AT EXIT   *
                ;*                               *
                ;*********************************

5CBA F5         OUTV:       PUSH AF
5CBB CD1200                 CALL OUTVV
5CBE F1                     POP  AF
5CBF C9                     RET


                ; ERROR ROUTINE. LOCATION 05FFEH
                ; PRESET TO FORCE A DYNAMIC STOP
                ; AT LABEL 'ERROR'. THE USER
                ; SHOULD OVERWRITE LASTWORD WITH
                ; THE ADDRESS OF HIS OWN ERROR
                ; ROUTINE.

5CC0 2AFE5F     ERROR:      LD   HL,(LASTWORD)
5CC3 E9                     JP   (HL)


                ;*********************************
                ;*                               *
                ;* OUTPUT NEWLINE TO VDU AND     *
                ;* PRINTER. THE REGISTERS ARE    *
                ;* UNCHANGED ON EXIT.            *
                ;*                               *
                ;*********************************

5CC4 F5         NLV:        PUSH AF
5CC5 3E0D                   LD   A,CR
5CC7 CDBA5C                 CALL OUTV
5CCA F1                     POP  AF
5CCB C9                     RET

5CCC F5         NLP:        PUSH AF
5CCD 3E0D                   LD   A,CR
5CCF CDE35D                 CALL OUTP
5CD2 F1                     POP  AF
```

```
121 5CD3 C9                     RET
122
123
124                 ;**********************************
125                 ;*                                *
126                 ;* OUTPUT SPACE TO VDU AND        *
127                 ;* PRINTER. THE REGISTERS ARE     *
128                 ;* UNCHANGED ON EXIT.             *
129                 ;*                                *
130                 ;**********************************
131
132
133
134 5CD4 F5       SPV:        PUSH AF
135 5CD5 3E20                  LD   A,SPC
136 5CD7 CDBA5C                CALL OUTV
137 5CDA F1                    POP  AF
138 5CDB C9                    RET
139
140
141
142 5CDC F5       SPP:        PUSH AF
143 5CDD 3E20                  LD   A,SPC
144 5CDF CDE35D                CALL OUTP
145 5CE2 F1                    POP  AF
146 5CE3 C9                    RET
147
148
149                 ;**********************************
150                 ;*                                *
151                 ;* OUTPUT NEWLINE, FOLLOWED BY    *
152                 ;* THE STRING POINTED AT BY 'DE'  *
153                 ;* TO THE VDU AND PRINTER.        *
154                 ;* REGISTERS UNCHANGED ON EXIT.   *
155                 ;*                                *
156                 ;**********************************
157
158
159 5CE4 CDC45C    LINEV:      CALL NLV
160 5CE7 CDF25C                CALL COPYV
161 5CEA C9                    RET
162
163
164 5CEB CDCC5C    LINEP:      CALL NLP
165 5CEE CD025D                CALL COPYP
166 5CF1 C9                    RET
167
168
169                 ;**********************************
170                 ;*                                *
171                 ;* COPY TEXT TO VDU AND PRINTER.  *
172                 ;* ENTER WITH 'DE' POINTING       *
173                 ;* TO STRING. THE STRING MUST     *
174                 ;* BE TERMINATED WITH A 'CR'.     *
175                 ;* ('CR' IS NOT PRINTED)          *
176                 ;* REGISTERS UNCHANGED ON EXIT    *
177                 ;*                                *
178                 ;**********************************
179
180
```

```
181
182 5CF2 F5        COPYV:      PUSH AF
183 5CF3 D5                   PUSH DE
184 5CF4 1A        COPYV1:     LD   A,(DE)
185 5CF5 13                   INC  DE
186 5CF6 FE0D                 CP   CR
187 5CF8 2805                 JR   Z,CVRET
188 5CFA CDBA5C               CALL OUTV
189 5CFD 18F5                 JR   COPYV1
190
191 5CFF D1        CVRET:      POP  DE
192 5D00 F1                   POP  AF
193 5D01 C9                   RET
194
195
196 5D02 F5        COPYP:      PUSH AF
197 5D03 D5                   PUSH DE
198 5D04 1A        COPY1:      LD   A,(DE)
199 5D05 13                   INC  DE
200 5D06 FE0D                 CP   CR                    ; TERMINATE ON
201 5D08 2805                 JR   Z,CRET
202 5D0A CDE35D               CALL OUTP
203 5D0D 18F5                 JR   COPY1
204
205 5D0F D1        CRET:       POP  DE
206 5D10 F1                   POP  AF
207 5D11 C9                   RET
208
209
210                ;*********************************
211                ;*                               *
212                ;* PRINT INTEGER TO VDU AND      *
213                ;* PRINTER.                      *
214                ;* ENTER WITH INTEGER IN HL.     *
215                ;* ENTER WITH CARRY CLEARED TO   *
216                ;* PRINT DECIMAL AND CARRY       *
217                ;* SET TO PRINT HEX.             *
218                ;* REGISTERS RESTORED ON EXIT.   *
219                ;*                               *
220                ;*********************************
221
222
223
224 5D12 D5        PRINTV:     PUSH DE
225 5D13 CD245D               CALL PRINT
226 5D16 CDF25C               CALL COPYV
227 5D19 D1                   POP  DE
228 5D1A C9                   RET
229
230
231 5D1B D5        PRINTP:     PUSH DE
232 5D1C CD245D               CALL PRINT
233 5D1F CD025D               CALL COPYP
234 5D22 D1                   POP  DE
235 5D23 C9                   RET
236
237
238                ;*********************************
239                ;*                               *
240                ;* COMMON PRINT ROUTINE.         *
```

```
241                     ;*                                  *
242                     ;************************************
243
244
245
246 5D24 E5          PRINT:      PUSH HL
247 5D25 C5                     PUSH BC
248 5D26 F5                     PUSH AF
249 5D27 01F35F                 LD   BC,DAR
250 5D2A ED43FB5F               LD   (DARRAY),BC
251 5D2E DABB5D                 JP   C,PHEX
252 5D31 AF                     XOR  A
253 5D32 32EE5F                 LD   (ZERO),A          ; CLEAR FLAG
254 5D35 CB7C                   BIT  7,H               ; CHECK SIGN
255 5D37 2823                   JR   Z,POSINT          ; POSITIVE
256 5D39 7C                     LD   A,H
257 5D3A D680                   SUB  128               ; -32768?
258 5D3C 2013                   JR   NZ,NOTMAXN        ; NO
259 5D3E 7D                     LD   A,L
260 5D3F B7                     OR   A
261 5D40 200F                   JR   NZ,NOTMAXN        ; NO
262 5D42 11495D                 LD   DE,MAXNEG         ; PRINT -32768
263 5D45 F1                     POP  AF
264 5D46 C1                     POP  BC
265 5D47 E1                     POP  HL
266 5D48 C9                     RET
267
268 5D49 2D333237 MAXNEG:       DB   '-32768 ',CR
268 5D4D 3638200D
269
270 5D51 EB          NOTMAXN:   EX   DE,HL             ; NEG NUMBER
271 5D52 210000                 LD   HL,0
272 5D55 B7                     OR   A
273 5D56 ED52                   SBC  HL,DE             ; INVERT INTEGER
274 5D58 3E2D                   LD   A,'-'
275 5D5A 1802                   JR   POSINT1
276 5D5C 3E20        POSINT:    LD   A,SPC
277 5D5E CDB05D      POSINT1:   CALL PLANT             ; SPC OR '-'
278 5D61 011027                 LD   BC,10000
279 5D64 CD905D                 CALL EXTDIG            ; EXTRACT FIRST DIGIT
280 5D67 01E803                 LD   BC,1000
281 5D6A CD905D                 CALL EXTDIG            ; EXTRACT SEC. DIGIT
282 5D6D 016400                 LD   BC,100
283 5D70 CD905D                 CALL EXTDIG            ; EXTRACT THIRD DIGIT
284 5D73 010A00                 LD   BC,10
285 5D76 CD905D                 CALL EXTDIG            ; EXTRACT FOURTH DIGIT
286 5D79 7D                     LD   A,L
287 5D7A C630                   ADD  A,'0'
288 5D7C CDB05D                 CALL PLANT             ; PLANT LAST DIGIT
289 5D7F 3E20        PLANTTL:   LD   A,SPC
290 5D81 CDB05D                 CALL PLANT
291 5D84 3E0D                   LD   A,CR
292 5D86 CDB05D                 CALL PLANT
293 5D89 11F35F                 LD   DE,DAR            ; POINT TO INTEGER STRI
294 5D8C F1                     POP  AF
295 5D8D C1                     POP  BC
296 5D8E E1                     POP  HL
297 5D8F C9                     RET
298
299 5D90 AF          EXTDIG:    XOR  A                 ; EXTRACT DIGIT
```

```
300 5D91 B7          EDL:    OR   A                  ; BY REPEATED S
301 5D92 ED42                SBC  HL,BC
302 5D94 FA9A5D              JP   M,EXF
303 5D97 3C                  INC  A                  ; DIGIT IN 'A'
304 5D98 18F7                JR   EDL
305
306 5D9A 09          EXF:    ADD  HL,BC
307 5D9B B7                  OR   A
308 5D9C 2805                JR   Z,LZ
309 5D9E 32EE5F              LD   (ZERO),A           ; SET ZERO FLAG
310 5DA1 1807                JR   EXTOUT
311 5DA3 57          LZ:     LD   D,A
312 5DA4 3AEE5F              LD   A,(ZERO)           ; CHECK ZERO FL
313 5DA7 B7                  OR   A
314 5DA8 7A                  LD   A,D
315 5DA9 C8                  RET  Z
316 5DAA C630        EXTOUT: ADD  A,'0'
317 5DAC CDB05D              CALL PLANT              ; PRINT DIGIT
318 5DAF C9                  RET
319
320 5DB0 ED4BFB5F    PLANT:  LD   BC,(DARRAY)        ; FORM DIGIT
321 5DB4 02                  LD   (BC),A             ; STRING FOR CO
322 5DB5 03                  INC  BC
323 5DB6 ED43FB5F            LD   (DARRAY),BC
324 5DBA C9                  RET
325
326
327 5DBB 3E20        PHEX:   LD   A,SPC              ; PRINT HEX
328 5DBD CDB05D              CALL PLANT
329 5DC0 7C                  LD   A,H
330 5DC1 CDD05D              CALL CONH
331 5DC4 7D                  LD   A,L
332 5DC5 CDD05D              CALL CONH
333 5DC8 3E48                LD   A,'H'
334 5DCA CDB05D              CALL PLANT
335 5DCD C37F5D              JP   PLANTTL
336
337
338
339 5DD0 F5          CONH:   PUSH AF                 ; EXTRACT HEX D
340 5DD1 1F                  RRA
341 5DD2 1F                  RRA
342 5DD3 1F                  RRA
343 5DD4 1F                  RRA
344 5DD5 CDDA03              CALL ASCI               ; CONVERT TO AS
345 5DD8 CDB05D              CALL PLANT
346 5DDB F1                  POP  AF
347 5DDC CDDA03              CALL ASCI
348 5DDF CDB05D              CALL PLANT
349 5DE2 C9                  RET
350
351
352
353                 ;**********************************
354                 ;*                                *
355                 ;* TEST PRINTER STATUS AND        *
356                 ;* OUTPUT CHARACTER FROM 'A'.     *
357                 ;* REGISTERS UNCHANGED ON EXIT.   *
358                 ;*                                *
359                 ;**********************************
```

```
360
361
362
363 5DE3 D5          OUTP:       PUSH DE                 ; OUTPUT TO PRINTER
364 5DE4 C5                      PUSH BC
365 5DE5 F5                      PUSH AF
366
367 5DE6 06C8        OUTPP1:     LD   B,200              ; CHECK STATUS 200
368                  ; TIMES
369 5DE8 167F        STAT:       LD   D,127              ;  INNER LOOP
370 5DEA DBFE        STATUS:     IN   A,(254)            ; INPUT STATUS
371 5DEC E60F                    AND  15
372 5DEE FE02                    CP   2
373 5DF0 2813                    JR   Z,OUTPP2
374 5DF2 CD5907                  CALL DELAY
375 5DF5 15                      DEC  D
376 5DF6 20F2                    JR   NZ,STATUS
377 5DF8 10EE                    DJNZ STAT
378 5DFA 110B5E                  LD   DE,OER
379 5DFD CDE45C                  CALL LINEV
380 5E00 CD2F5E      OUTPP3:     CALL INKEY              ; WAIT FOR KEY
381 5E03 18E1                    JR   OUTPP1             ; RE-TRY STATUS
382
383 5E05 F1          OUTPP2:     POP  AF                 ; GET CHAR
384 5E06 D3FF                    OUT  (255),A            ; OUT TO PRINTER
385 5E08 C1                      POP  BC
386 5E09 D1                      POP  DE
387 5E0A C9                      RET
388
389 5E0B 43484543 OER:           DB   'CHECK PRINTER. '
389 5E0F 4B205052
389 5E13 494E5445
389 5E17 522E20
390 5E1A 54595045                DB   'TYPE CR TO CONTIN
390 5E1E 20435220
390 5E22 544F2043
390 5E26 4F4E5449
390 5E2A 4E55452E
390 5E2E 0D
391
392
393
394                  ;**********************************
395                  ;*                                *
396                  ;* INPUT ONE CHARACTER FROM       *
397                  ;* KEYS TO 'A'. CARRY SET IF      *
398                  ;* 'BREAK' KEY PRESSED.           *
399                  ;* 'BC','DE' AND 'HL' UNCHANGED*
400                  ;* ON EXIT.                       *
401                  ;*                                *
402                  ;**********************************
403
404
405
406 5E2F C5          INKEY:      PUSH BC
407 5E30 CD1B00      INKEY0:     CALL INK
408 5E33 B7                      OR   A
409 5E34 2005                    JR   NZ,INKEY1
410 5E36 32ED5F                  LD   (INFLAG),A
411 5E39 18F5                    JR   INKEY0
```

```
412 5E3B 47       INKEY1:     LD   B,A
413 5E3C 3AED5F               LD   A,(INFLAG)
414 5E3F B7                   OR   A
415 5E40 78                   LD   A,B
416 5E41 20ED                 JR   NZ,INKEY0
417 5E43 32ED5F               LD   (INFLAG),A
418 5E46 FE64                 CP   64H                ; BREAK
419 5E48 280C                 JR   Z,INKEY3
420 5E4A FE66                 CP   66H                ; CR
421 5E4C 2002                 JR   NZ,INKEY2
422 5E4E 3E0D                 LD   A,CR
423 5E50 CDBA5C   INKEY2:     CALL OUTV
424 5E53 B7                   OR   A
425 5E54 C1                   POP  BC
426 5E55 C9                   RET
427
428 5E56 37       INKEY3:     SCF
429 5E57 C1                   POP  BC
430 5E58 C9                   RET
431
432
433
434               ;*************************************
435               ;*                                   *
436               ;* INPUT STRING FROM KEYS INTO       *
437               ;* BUFFER POINTED AT BY 'DE'.        *
438               ;* THE STRING IS TERMINATED BY A     *
439               ;* 'CR'. ON EXIT, 'DE' POINTS TO     *
440               ;* THE START OF THE STRING. ALL      *
441               ;* THE OTHER REGISTERS ARE           *
442               ;* UNCHANGED.                        *
443               ;*                                   *
444               ;*************************************
445
446
447
448 5E59 F5         STRINGKEY:  PUSH AF
449 5E5A D5                   PUSH DE
450 5E5B CD2F5E   SK1:        CALL INKEY
451 5E5E 12                   LD   (DE),A
452 5E5F 13                   INC  DE
453 5E60 FE0D                 CP   CR                 ; END OF STRI
454 5E62 20F7                 JR   NZ,SK1
455 5E64 D1                   POP  DE
456 5E65 F1                   POP  AF
457 5E66 C9                   RET
458
459
460               ;*************************************
461               ;*                                   *
462               ;* READ INTEGER FROM KEYS.           *
463               ;*    ( -32768 (= INT (= 32767 OR    *
464               ;*         8000H (= INT (= 32767 )   *
465               ;* INTEGER IN 'HL' ON EXIT. ALL      *
466               ;* OTHER REGISTERS UNCHANGED.        *
467               ;*                                   *
468               ;*************************************
469
470
471
```

```
5E67 11A711   READKEY:   LD   DE,MESSAGEBUFF+4
5E6A CD595E              CALL STRINGKEY
5E6D CD745E              CALL ZMB
5E70 CD825E              CALL READIN
5E73 C9                  RET


                ;**********************************
                ;*                                *
                ;* ZERO BUFFER FOR LEADING ZEROS  *
                ;*                                *
                ;**********************************


5E74 F5       ZMB:       PUSH AF
5E75 C5                  PUSH BC
5E76 D5                  PUSH DE
5E77 E1                  POP  HL
5E78 013004              LD   BC,430H
5E7B 2B       ZMB1:      DEC  HL
5E7C 71                  LD   (HL),C
5E7D 10FC                DJNZ ZMB1
5E7F C1                  POP  BC
5E80 F1                  POP  AF
5E81 C9                  RET


                ;**********************************
                ;*                                *
                ;* COMMON READ ROUTINE.           *
                ;*                                *
                ;**********************************


5E82 D5       READIN:    PUSH DE
5E83 C5                  PUSH BC
5E84 F5                  PUSH AF
5E85 0605                LD   B,5
5E87 D5                  PUSH DE
5E88 1A       RI1:       LD   A,(DE)          ; LOOK FOR HEX
5E89 FE48                CP   'H'
5E8B 2810                JR   Z,RI2
5E8D FE0D                CP   CR
5E8F 2804                JR   Z,RI3
5E91 13                  INC  DE
5E92 23                  INC  HL
5E93 10F3                DJNZ RI1
5E95 D1       RI3:       POP  DE
5E96 CDAD5E              CALL READ            ; DECIMAL
5E99 F1                  POP  AF
5E9A C1                  POP  BC
5E9B D1                  POP  DE
5E9C C9                  RET

5E9D D1       RI2:       POP  DE              ; HEX STARTS FROM HL
5E9E EB                  EX   DE,HL
5E9F CD1004              CALL CONHEX
```

```
 532 5EA2 3005                  JR   NC,RIOUT
 533 5EA4 3E02                  LD   A,2                ; ERROR 2
 534 5EA6 C3C05C                JP   ERROR
 535 5EA9 F1            RIOUT:  POP  AF
 536 5EAA C1                    POP  BC
 537 5EAB D1                    POP  DE
 538 5EAC C9                    RET
 539
 540
 541
 542                    ;**********************************
 543                    ;*                                *
 544                    ;* COMMON DECIMAL READ ROUTINE.   *
 545                    ;*                                *
 546                    ;**********************************
 547
 548
 549
 550 5EAD AF            READ:   XOR  A
 551 5EAE 210000                LD   HL,0
 552 5EB1 22E35F                LD   (INT),HL           ; CLEAR INTEGER
 553 5EB4 32FD5F                LD   (SI),A             ; CLEAR SIGN FLAG
 554 5EB7 3D                    DEC  A
 555 5EB8 32EF5F                LD   (RFLAG),A          ; SET TERM FLAG
 556 5EBB D5                    PUSH DE
 557 5EBC 1A                    LD   A,(DE)
 558 5EBD FE2D                  CP   '-'
 559 5EBF 2009                  JR   NZ,FDIG
 560 5EC1 3E01                  LD   A,1
 561 5EC3 32FD5F                LD   (SI),A             ; SIGN = -
 562 5EC6 D1            READ1:  POP  DE
 563 5EC7 13                    INC  DE
 564 5EC8 1A                    LD   A,(DE)
 565 5EC9 D5                    PUSH DE
 566 5ECA D630          FDIG:   SUB  '0'                ; DIGIT 0-9?
 567 5ECC FA085F                JP   M,CHECKF
 568 5ECF FE0A                  CP   10
 569 5ED1 F2085F                JP   P,CHECKF
 570 5ED4 32EF5F                LD   (RFLAG),A          ; END ON NEXT NON-
 571 5ED7 B7                    OR   A
 572 5ED8 2AE35F                LD   HL,(INT)           ; FORM HL=HL*10+A
 573 5EDB E5                    PUSH HL
 574 5EDC D1                    POP  DE
 575 5EDD 0609                  LD   B,9                ; *10
 576 5EDF 19            READ3:  ADD  HL,DE
 577 5EE0 10FD                  DJNZ READ3
 578 5EE2 010000                LD   BC,0
 579 5EE5 4F                    LD   C,A
 580 5EE6 09                    ADD  HL,BC
 581 5EE7 22E35F                LD   (INT),HL
 582 5EEA 30DA                  JR   NC,READ1
 583 5EEC 21CC0C                LD   HL,3276            ; POSSIBLE OVERFL
 584 5EEF 3F                    CCF
 585 5EF0 ED52                  SBC  HL,DE
 586 5EF2 200F                  JR   NZ,READ2
 587 5EF4 FE08                  CP   8                  ; LAST DIGIT=8?
 588 5EF6 200B                  JR   NZ,READ2
 589 5EF8 3AFD5F                LD   A,(SI)             ;MUST BE -VE FOR
 590 5EFB B7                    OR   A
 591 5EFC 2805                  JR   Z,READ2
```

```
5EFE 210080             LD   HL,8000H          ; SET INT TO -32768
5F01 181B               JR   CH1

5F03 3E03     READ2:    LD   A,3               ; ERROR 3
5F05 C3C05C             JP   ERROR

5F08 3AEF5F   CHECKF:   LD   A,(RFLAG)         ;IGNORE
              ;LEADING NON-DIGITS
5F0B B7                 OR   A
5F0C FAC65E             JP   M,READ1
5F0F 2AE35F             LD   HL,(INT)          ;END ON TRAILING
              ;NON-DIGIT
5F12 3AFD5F             LD   A,(SI)            ;ADJUST INT FOR SIGN
5F15 B7                 OR   A
5F16 2806               JR   Z,CH1
5F18 EB                 EX   DE,HL
5F19 210000             LD   HL,0
5F1C ED52               SBC  HL,DE
5F1E D1       CH1:      POP  DE
5F1F C9                 RET


              ;**********************************
              ;*                                *
              ;* DEBUG ROUTINE.                 *
              ;* EDIT INTO ASSEMBLER CODE THE*
              ;* CALL:-    CALL DEBUG           *
              ;* THE OUTPUT WILL BE:-           *
              ;* PC AF  BC  DE  HL  IX IY=      *
              ;* SP AF' BC' DE' HL' IR  =       *
              ;*       ON THE PRINTER.          *
              ;*                                *
              ;* THIS MAY BE CONVERTED TO USE*
              ;* THE VDU BY REPLACING ALL       *
              ;* PRINTER CALLS WITH THE         *
              ;* EQUIVALENT VDU CALLS. IN       *
              ;* ADDITION, INSERT A 'CALL       *
              ;* INKEY' TO PREVENT THE SCREEN*
              ;* FROM SCROLLING.                *
              ;**********************************


5F20 22E05F   DEBUG:    LD   (THL),HL
5F23 F5                 PUSH AF
5F24 E1                 POP  HL
5F25 22DE5F             LD   (TAF),HL
5F28 E1                 POP  HL
5F29 E5                 PUSH HL                ; RET
5F2A FDE5               PUSH IY
5F2C DDE5               PUSH IX
5F2E D5                 PUSH DE
5F2F 11AA5F             LD   DE,DG1
5F32 CDEB5C             CALL LINEP
5F35 2B                 DEC  HL
5F36 2B                 DEC  HL
5F37 2B                 DEC  HL
5F38 37                 SCF
5F39 CD1B5D             CALL PRINTP            ; PC OF DEBUG
```

```
652 5F3C 2ADE5F          LD    HL,(TAF)
653 5F3F CD1B5D          CALL  PRINTP
654 5F42 C5              PUSH  BC
655 5F43 E1              POP   HL
656 5F44 CD1B5D          CALL  PRINTP
657 5F47 D1              POP   DE
658 5F48 EB              EX    DE,HL
659 5F49 CD1B5D          CALL  PRINTP
660 5F4C EB              EX    DE,HL
661 5F4D 2AE05F          LD    HL,(THL)
662 5F50 CD1B5D          CALL  PRINTP
663 5F53 E1              POP   HL                ; IX
664 5F54 CD1B5D          CALL  PRINTP
665 5F57 E1              POP   HL                ; IY
666 5F58 CD1B5D          CALL  PRINTP
667 5F5B D5              PUSH  DE
668 5F5C 11C45F          LD    DE,DG2
669 5F5F CDEB5C          CALL  LINEP
670 5F62 D1              POP   DE
671 5F63 2ADE5F          LD    HL,(TAF)
672 5F66 E5              PUSH  HL
673 5F67 F1              POP   AF
674 5F68 2AE05F          LD    HL,(THL)
675 5F6B 08              EX    AF,AF
676 5F6C D9              EXX
677 5F6D 22E05F          LD    (THL),HL
678 5F70 F5              PUSH  AF
679 5F71 E1              POP   HL
680 5F72 22DE5F          LD    (TAF),HL
681 5F75 210000          LD    HL,0
682 5F78 39              ADD   HL,SP
683 5F79 37              SCF
684 5F7A CD1B5D          CALL  PRINTP            ; SP
685 5F7D 2ADE5F          LD    HL,(TAF)
686 5F80 CD1B5D          CALL  PRINTP
687 5F83 C5              PUSH  BC
688 5F84 E1              POP   HL
689 5F85 CD1B5D          CALL  PRINTP
690 5F88 D5              PUSH  DE
691 5F89 E1              POP   HL
692 5F8A CD1B5D          CALL  PRINTP
693 5F8D 2AE05F          LD    HL,(THL)
694 5F90 CD1B5D          CALL  PRINTP
695 5F93 ED57            LD    A,I
696 5F95 67              LD    H,A
697 5F96 ED5F            LD    A,R
698 5F98 6F              LD    L,A
699 5F99 CD1B5D          CALL  PRINTP
700 5F9C 2ADE5F          LD    HL,(TAF)
701 5F9F E5              PUSH  HL
702 5FA0 F1              POP   AF
703 5FA1 2AE05F          LD    HL,(THL)
704 5FA4 08              EX    AF,AF
705 5FA5 D9              EXX
706 5FA6 CDCC5C          CALL  NLP
707 5FA9 C9              RET
708
709 5FAA 50432041 DG1:   DB    'PC AF  BC  DE  HL
709 5FAE 46204243
709 5FB2 43204445
```

```
709 5FB6 45202048
709 5FBA 4C202049
709 5FBE 58204959
709 5FC2 3D
710 5FC3 0D                     DB    CR
711 5FC4 53502041 DG2:          DB    "SP AF' BC' DE' HL
711 5FC8 46272042
711 5FCC 43272044
711 5FD0 45272048
711 5FD4 4C272049
711 5FD8 52
712 5FD9 2020203D               DB    '    =',CR
712 5FDD 0D
713
714 5FDE 0000       TAF:        DW    0
715 5FE0 0000       THL:        DW    0
716
717
718
719                 ;***********************************
720                 ;*                                 *
721                 ;* GOBAL WORKSPACE FOR ROUTINES.   *
722                 ;*                                 *
723                 ;***********************************
724
725
726                             ORG   05FE2H
727                             LOAD  05FE2H
728
729
730 5FE2 00          FLAGP:      DB    0
731 5FE3 0000        INT:        DW    0
732 5FE5 0000        OUTB:       DW    0
733 5FE7 0000        MAXOUTB:    DW    0
734 5FE9 0000        INB:        DW    0
735 5FEB 0000        MAXINB:     DW    0
736 5FED 00          INFLAG:     DB    0
737 5FEE 00          ZERO:       DB    0
738 5FEF 00          RFLAG:      DB    0
739 5FF0 000000      DDAR:       DB    0,0,0
740 5FF3 00000000    DAR:        DB    0,0,0,0,0,0,0,0
740 5FF7 00000000
741 5FFB 0000        DARRAY:     DW    0
742 5FFD 00          SI:         DB    0
743
744 5FFE C05C                    DW    ERROR
745
746
747                             END
```

Dear S.U.N. and SHARPSOFT,

Ref: ZEN TOOLKIT and DISASSEMBLEBA00

A year ago I purchased 'ZEN-DOS' Assembler from you which, after chasing you for missing documentation, has been extremely useful. I am finding the DOS to be a significant improvement on the SHARP SP-6015 DOS and the ZEN assembler is both fast and relatively easy to use.

I found Mr. M.J. Wiechowski's letter about ZEN-DOS (SUN 16) very interesting and have successfully modified my copy of SHARPSOFT Forth as suggested so that it will run under ZEN-DOS, but still have a lot to do to write the disk-filing routines (can Mr. Wiechowski offer any advice?) However, the routine he gives for 'TSAVE' does not appear to work and assemble one byte short of BFFH. Is it an error in typesetting SUN?

The entry and editing of source code with ZEN is cumbersome and so I was delighted to see a 'ZEN TOOLKIT' advertised by yourselves, which claimed to provide 'on-screen' editing and various other 'bells-and-whistles'.

I purchased the toolkit together with a cassette-based disassembler ('DISASSEMBLEBA00') and set about adding the cassette-based toolkit routines to my DOS-based ZEN. That's when my problems started!

Firstly, the toolkit-modified ZEN just crashed each time it was tried. This turned out to be due to the toolkit overwriting part of the DOS-based ZEN. The toolkit had been written to add code to the end of cassette-based ZEN (from 23B0H to 3820H), and to modify some addresses and code within ZEN.

So, by studying a source code listing of cassette-based ZEN and a disassembly of DOS-based ZEN, I managed to produce a version of cassette-based ZEN which could be run with the toolkit. This I called 'SUPAZENCAS'.

However, the special printer routines included in the toolkit, one of which ('Modified EPSON') was claimed to "..print the full set of SHARP characters plus high resolution graphics.", have proved to be most unsatisfactory. This 'modified EPSON' is, in face, the same routine as that used for the 'SHARP' option and does not operate as claimed (at least not with my set-up of EPSON FX-80 + Betan printer interface card). So I am now trying to unravel the relevant code and insert suitable routines for driving my printer.

The disassembler has been extremely useful (as you can see above) although it too has caused me some headaches. It is supplied (according to the documentation provided) in three versions: 1) Loading at BA00H, 2) Loading at 8A00H, 3) A simplified version loading at 4E00H. The tape which arrived had only the BA00H version which over-wrote the DOS v1.1 at C000H-CFFH used by ZEN-DOS.

Rather than recreate a 8A00H version which would waste valuable ZEN-source storage space, I re-located the disassembler at AA00H. It is now possible to have ZEN, ZEN-DOS and the disassembler resident at the same time, making machine code 'bashing' a great deal easier.

This is still not the end of the story. As with SUPAZENCAS, the disassembler output to the printer caused form-feeds, shift in/out and other horrors whenever a non-ASCII character was output to the printer ins traight HEX. I now have to re-write the printer routine to make it useable with my printer set-up.

Despite the hair-tearing frustrations experienced and the countless hours spent getting this far, I am now a confirmed 'machine-code junkie' and would like to get in touch with any other MZ-80K machine-code or ZEN-DOS users who would ike to pool ideas (and more importantly, solutions!).

Whilst on the 'scrounge', has any reader information on the MZ-80K ports (e.g. what do all the bits in the priter status port - FED - do)?

I understand that the MZ-80K is to be dropped by S.U.N. What a disaster! If enough machine-code freaks contact me after this letter is published, a separate MZ-80K newsletter could be established by us, the users, to maintain the flow of ideas and informtion bout this fine machine (shame about the keyboard - anybody done any mods in this direction?) It would be a great pity for MZ-80K owners to be left high and dry without a vehicle for communication.

The mention of communications reminds me....
I will be acquiring a 300/300 baud modem shortly and would like to contact any MZ-80K owner who has experience with using modems and '80Ks and who can advise on software for disk file transmission.

Keep up the good work S.U.N.

Doug Edworthy
5 Framfield Road
UCKFIELD
East Sussex,
TN22 5AG.

Dear Sirs,

Please find enclosed one software and one hardware mod for use with a Sharp MZ-80B. Hopefully this should help your "Bumper Edition".

I currently use an MZ-80B with a three and a half inch disk system, home brew RS232 interface, home brew eprom programmer (which will handle most types of eproms), two character sets and an RS232 printer.

Any reader interested in building any of the above interfaces or the exchange of ideas/ information should send a S.A.E.

NOTE The above interfaces require a reasonable knowledge of electronics and are **NOT SUITABLE FOR THE INEXPERIENCED.**

SOFTWARE MODIFICATION

This patch should be carried out to the Sharp MZ-80B Disc Basic 'Utility' program, it will then permit bootable copies of submaster discs to be produced (can also be used for F-DOS submaster copies), master disks will never wear out!

1. Run 'Utility' program.

2. Press monitor button at rear of machine.

3. Modify the following locations using the monitor "M" command

$13FB 3A 3E (CR)

$13FC 1D FF (CR)

$13FD 40 32 (CR)

$13FE FE 1D (CR)

$13FF FF 40 (CR)

$1400 C2 00 (CR)

$1401 E3 00 (CR)

$1402 14 00 (CR)

(shift break)

Place tape in recorder

S (CR)

FILE NAME SUPERCOPY (CR)

S-ADR$1220 (CR)

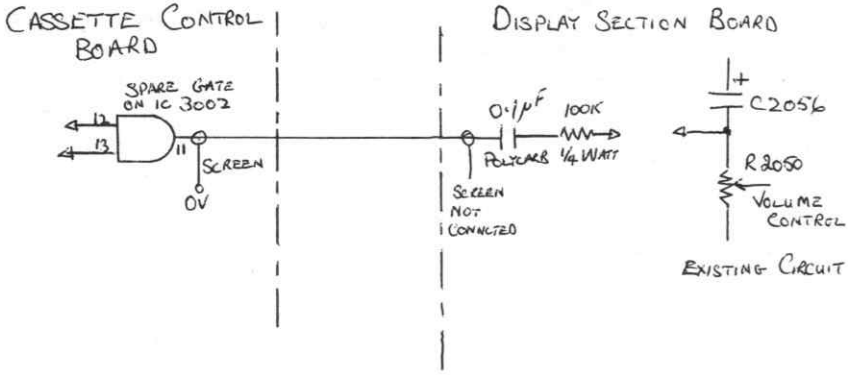
E-ADR$1CD3 (CR)

J-ADR$1220 (CR)

When the data has been saved re-boot and use the Sharp 'Filing CMT' program to transfer to disc, ensuring that the filename is not already used. 'Supercopy' can be used in the same way as 'Utility' but the dreaded 'CAN NOT COPY' will not appear.

HARDWARE MODIFICATION
*** SEE WARNING NOTE ABOVE ***

This mod will inject data to/from the tape recorder into the sound amplifier, this makes it easy to find program starts and ocnfirm data transfers to/from tape, data will also be heard in the fast forward/rev mode. The following circuit should be installed with reference to the circuit diagrams supplied in the MZ-80B Owner's manual.




```
10 REM --- SEND PRINTER DRIVER CODES
20 REM --- Sharp Basic does not allow codes < 32d to be
30 REM --- sent to a printers whereas many printers have
40 REM --- their control codes in this range (standard
50 REM --- ASCII Escape codes).
60 REM --- To use; POKE codes into address 65535; then
70 REM --- enter USR($FF44) to send to printer.
80 REM --- Ex 'ESC A 5';  POKE65535,27:USR($FF44)
90 REM ---                POKE65535,65:USR($FF44)
100 REM --                POKE65535,5 :USR($FF44)
110 REM
120 REM     ARENS; 5 HEIDEPARK; WAGENINGEN HOOG; NL-6705 AB HOLLAND
130 REM
60000 REM ---- Code from $FF40 to $FFFF (protected from BASIC)
60010 FORI=65344TO65535:READX:POKEI,X:NEXT
60020 DATA 0,0,0,0,58,255,255,245
60030 DATA 62,0,205,93,255,241,211,255
60040 DATA 62,128,211,254,62,1,205,93
60050 DATA 255,175,211,254,201,197,213,87
60060 DATA 30,12,1,0,0,219,254,230
60070 DATA 13,186,32,3,209,193,201,11
60080 DATA 120,177,32,241,29,32,238,209
60090 DATA 193,195,161,255,71,62,27,205
60100 DATA 71,255,120,205,71,255,62,0
60110 DATA 205,93,255,219,254,15,15,201
60120 DATA 62,3,205,124,255,210,161,255
60130 DATA 62,4,205,124,255,210,161,255
60140 DATA 201,205,166,12,17,189,255,205
60150 DATA 219,8,205,176,8,17,245,255
60160 DATA 205,233,14,205,113,8,254,0
60170 DATA 40,249,195,128,18,72,65,82
60180 DATA 68,87,65,82,69,32,69,82
60190 DATA 82,79,82,32,79,78,32,84
60200 DATA 72,69,32,80,82,73,78,84
60210 DATA 69,82,46,32,72,73,84,32
60220 DATA 65,78,89,32,75,69,89,32
60230 DATA 84,79,32,67,79,78,84,73
60240 DATA 78,85,69,46,13,65,48,43
60250 DATA 65,82,65,43,65,82,42,83
```

```
 10 REM -- THE EARTH
 20 CONSOLEC80,S0,24,N:GRAPHI1,C,O1
 30 PRINT"  THE EARTH"
 40 RESTORE
 50 FORI=1TO2046:READX,Y:SETX,Y:NEXT
 60 FORN=1TO10000:NEXT
 70 GRAPHC,I1,O1:PRINTCHR$(6):CLR:END
 80 DATA6,47,8,46,10,45,12,44,12,45,13,43,13,45,14,43,14,45,15,42,15,45,16,42,1
,44,16,45,17,41,17,44,18,39,18,40
 90 DATA18,41,18,43,19,38,19,43,20,37,20,43,21,36,21,43,22,35,22,43,22,73,22,74
22,75,22,76,22,77,22,78,22,79,22,80
 100 DATA23,32,23,33,23,35,23,42,23,69,23,70,23,71,23,72,23,81,23,82,24,31,24,3
,24,35,24,42,24,64,24,65,24,66,24,67
 110 DATA24,68,24,79,24,80,24,83,24,84,25,31,25,42,25,63,25,79,25,81,25,82,25,8
,25,86,26,31,26,42,26,61,26,62,26,79
 120 DATA26,83,26,84,26,87,27,31,27,42,27,60,27,80,27,85,27,86,28,31,28,41,28,5
,28,81,28,82,29,31,29,41,29,58,29,83
 130 DATA29,85,29,86,29,87,29,88,30,30,30,41,30,57,30,84,30,89,31,29,31,42,31,5
,31,55,31,56,31,90,32,29,32,43,32,44
 140 DATA32,52,32,53,32,91,32,92,33,29,33,45,33,46,33,47,33,48,33,49,33,50,33,5
,33,93,34,29,34,94,35,29,35,95,36,29
 150 DATA36,95,37,29,37,88,37,89,37,90,37,96,38,29,38,85,38,86,38,87,38,91,38,9
,39,30,39,82,39,83,39,84,39,92,39,97
 160 DATA40,31,40,81,40,93,40,97,40,98,40,99,41,32,41,80,41,92,41,100,42,33,42,
0,42,91,42,101,43,33,43,80,43,90,43,94
 170 DATA43,95,43,96,43,102,44,32,44,81,44,90,44,93,44,97,44,102,45,33,45,82,45
91,45,92,45,98,45,103,46,34,46,83,46,98
 180 DATA46,104,46,105,46,118,46,119,46,120,47,35,47,82
 190 DATA47,98,47,100,47,101,47,102,47,106,47,116,47,117,47,121,47,122,47,123,4
,35,48,81,48,99,48,103,48,107,48,114
 200 DATA48,115,48,124,48,125,49,34,49,80,49,90,49,104,49,107,49,113,49,126,50,
0,50,33,50,80,50,90,50,91,50,105,50,108
 210 DATA50,111,50,112,50,127,51,29,51,31,51,33,51,81,51,90,51,91,51,105,51,10
51,110,51,128,52,28,52,31,52,32,52,33
 220 DATA52,82,52,83,52,84,52,85,52,86,52,91,52,95,52,104,52,105,52,129,52,130
2,131,53,28,53,31,53,34,53,87,53,91
 230 DATA53,104,53,132,54,29,54,30,54,34,54,35,54,82,54,83,54,84,54,85,54,86,54
92,54,104,54,133,55,34,55,36,55,80
 240 DATA55,81,55,92,55,105,55,134,56,35,56,79,56,88,56,92,56,95,56,106,56,135
6,136,57,35,57,79,57,105,57,137,57,138
 250 DATA57,139,57,146,57,147,58,30,58,36,58,78,58,87,58,95,58,104,58,140,58,14
,58,142,58,143,58,144,58,145,58,148,58,149
 260 DATA58,150,58,151,58,152,58,159,58,161,59,37,59,77,59,89,59,104,59,153,59,
54,59,155,59,156,59,164,59,166,60,28
 270 DATA60,37,60,76,60,91,60,94,60,104,60,157,60,158,60,159,60,160,60,161,60,1
2,60,168,61,37,61,48,61,49,61,73,61,74
 280 DATA61,75,61,93,61,95,61,104,61,163,61,164,61,170,62,31,62,32,62,37,62,48,
2,47,62,50,62,72,62,91,62,93,62,95
 290 DATA62,104,62,165,62,166,62,167,63,25,63,26,63,31
 300 DATA63,33,63,38,63,45,63,51,63,71,63,93,63,95,63,104,63,168,63,169,63,171,
4,25,64,26,64,30,64,33,64,39,64,44
 310 DATA64,52,64,70,64,93,64,95,64,105,64,170,65,24,65,25,65,30,65,33,65,39,65
44,65,53,65,54,65,55,65,56,65,69,65,94
 320 DATA65,106,65,171,66,30,66,33,66,39,66,43,66,57,66,69,66,107,66,172,67,27,
7,31,67,32,67,39,67,43,67,56,67,69
 330 DATA67,107,67,172,68,28,68,38,68,43,68,51,68,56,68,68,68,94,68,108,68,162,
8,161,68,162,68,165,68,166,68,167,68,168
 340 DATA68,169,68,172,69,25,69,36,69,37,69,42,69,50,69,55,69,68,69,108,69,159,
9,163,69,164,69,170,69,172,70,24,70,26
 350 DATA70,27,70,35,70,42,70,54,70,68,70,95,70,108,70,158,70,171,71,24,71,27,7
,31,71,35,71,42,71,52,71,53,71,67,71,109
 360 DATA71,154,71,155,71,157,72,25,72,26,72,31,72,35,72,41,72,50,72,51,72,66,7
,97,72,109,72,153,72,156,73,32,73,35
 370 DATA73,41,73,43,73,49,73,66,73,110,73,152,74,32,74,36,74,40,74,42,74,44,74
47,74,48,74,62,74,66,74,100,74,104
 380 DATA74,111,74,153,75,29,75,32,75,37,75,39,75,42,75,44,75,46,75,61,75,63,75
66,75,102,75,112,75,113,75,114,75,115
 390 DATA75,153,75,167,76,28,76,30,76,38,76,43,76,44,76,46,76,61,76,64,76,65,76
67,76,100,76,116,76,152,76,167,76,168
 400 DATA77,25,77,28,77,30,77,47,77,60,77,66,77,67,77,117
 410 DATA77,150,77,151,78,24,78,25,78,28,78,29,78,47,78,60,78,117,78,149,79,23,
9,25,79,48,79,60,79,65,79,118,79,145
 420 DATA79,146,79,147,79,148,80,23,80,25,80,33,80,34,80,41,80,42,80,49,80,50,8
,51,80,60,80,64,80,68,80,69,80,118
 430 DATA80,144,81,22,81,24,81,33,81,35,81,40,81,43,81,50,81,60,81,118,81,143,1
,22,82,23,82,28,82,29,82,34,82,36,82,39
```

```
 440 DATA82,44,82,45,82,49,82,60,82,119,82,142,83,21,83,23,83,35,83,37,83,38,83,
46,83,49,83,52,83,53,83,54,83,55,83,59
 450 DATA83,120,83,141,84,21,84,22,84,31,84,35,84,46,84,50,84,51,84,56,84,57,84,
58,84,61,84,62,84,120,84,140,85,26
 460 DATA85,32,85,35,85,46,85,60,85,63,85,120,85,137,85,138,85,139,86,32,86,36,8
6,44,86,45,86,46,86,59,86,63,86,120
 470 DATA86,135,86,136,87,36,87,43,87,47,87,60,87,63,87,121,87,132,87,133,87,134
,88,25,88,26,88,30,88,31,88,36,88,37
 480 DATA88,38,88,39,88,43,88,61,88,62,88,122,88,131,89,24,89,26,89,29,89,31,89,
40,89,42,89,61,89,123,89,130,89,168
 490 DATA89,170,90,24,90,26,90,28,90,30,90,35,90,36,90,41,90,124,90,125,90,126,9
0,127,90,128,90,129,90,168,91,23,91,27
 500 DATA91,29,91,30,91,34,91,37,91,169,92,23,92,29,92,34,92,37,93,23,93,29,93,3
5,93,36,94,24,94,28,94,31,94,32,94,47
 510 DATA94,48,95,18,95,19,95,24,95,27,95,30,95,33,95,44
 520 DATA95,45,95,46,95,49,96,17,96,20,96,24,96,26,96,30,96,34,96,43,96,50,97,17
,97,21,97,24,97,26,97,30,97,35,97,39
 530 DATA97,40,97,42,97,50,98,17,98,21,98,24,98,25,98,29,98,36,98,37,98,38,98,41
,98,50,99,18,99,21,99,29,99,49,100,19
 540 DATA100,22,100,28,100,48,101,19,101,22,101,27,101,48,101,49,102,19,102,23,1
02,26,102,47,103,20,103,23,103,26,103,46
 550 DATA104,21,104,22,104,26,104,46,105,26,105,46,106,26,106,45,107,26,107,45,1
08,25,108,44,109,24,109,43,110,24,110,43
 560 DATA111,25,111,43,111,102,112,25,112,42,112,99,112,100,112,101,112,103,113,
25,113,42,113,96,113,97,113,98,113,104
 570 DATA113,105,114,25,114,41,114,47,114,48,114,91,114,92,114,93,114,94,114,95,
114,106,115,24,115,40,115,46,115,49,115,90
 580 DATA115,107,116,24,116,39,116,45,116,49,116,89,116,108,116,109,117,25,117,3
9,117,45,117,49,117,88,117,110,117,111
 590 DATA118,26,118,34,118,35,118,38,118,45,118,49,118,88,118,112,119,27,119,33,
119,36,119,37,119,44,119,45,119,48,119,87
 600 DATA119,113,120,27,120,32,120,42,120,43,120,48,120,86,120,113,121,27,121,31
,121,42,121,47,121,84,121,85,121,112,122,27
 610 DATA122,30,122,42,122,45,122,46,122,83,122,111,123,28,123,29,123,43,123,44,
123,71,123,72,123,73,123,74,123,82,123,111
 620 DATA124,70,124,75,124,80,124,81,124,111,125,58,125,70
 630 DATA125,76,125,78,125,79,125,112,126,57,126,59,126,70,126,76,126,77,126,112
,127,57,127,59,127,70,127,77,127,111,128,58
 640 DATA128,64,128,65,128,70,128,76,128,78,128,111,129,55,129,56,129,59,129,63,
129,66,129,69,129,70,129,76,129,78,129,112
 650 DATA130,54,130,57,130,58,130,60,130,62,130,67,130,68,130,69,130,75,130,78,1
30,112,131,53,131,55,131,56,131,60,131,62
 660 DATA131,71,131,72,131,73,131,74,131,79,131,112,132,57,132,60,132,62,132,71,
132,79,132,113,133,58,133,59,133,61,133,70
 670 DATA133,74,133,80,133,113,134,61,134,69,134,80,134,114,134,117,134,118,134,
119,135,60,135,69,135,80,135,115,135,116
 680 DATA135,120,135,136,135,137,135,138,136,51,136,52,136,53,136,59,136,69,136,
80,136,121,136,122,136,132,136,133,136,134
 690 DATA136,135,136,139,136,140,137,50,137,54,137,59,137,69,137,79,137,123,137,
124,137,125,137,130,137,131,137,141,137,142
 700 DATA138,48,138,49,138,54,138,57,138,58,138,70,138,72,138,79,138,126,138,127
,138,128,138,129,138,143,138,144,138,145
 710 DATA139,47,139,53,139,56,139,70,139,73,139,79,139,80,139,81,139,146,139,147
,139,148,140,46,140,54,140,57,140,58,140,59
 720 DATA140,71,140,82,140,149,140,150,140,151,140,152,141,45,141,55,141,60,141,
72,141,75,141,82,141,153,142,44,142,56
 730 DATA142,60,142,69,142,70,142,73,142,74,142,76,142,82
 740 DATA142,154,143,43,143,56,143,60,143,69,143,71,143,76,143,83,143,154,144,32
,144,33,144,43,144,55,144,59,144,69,144,72
 750 DATA144,74,144,75,144,84,144,154,145,31,145,34,145,42,145,51,145,52,145,53,
145,54,145,59,145,70,145,72,145,73,145,85
 760 DATA145,154,146,29,146,31,146,34,146,42,146,50,146,58,146,71,146,85,146,154
,147,31,147,33,147,41,147,49,147,56,147,57
 770 DATA147,72,147,73,147,74,147,75,147,85,147,153,148,31,148,32,148,40,148,49,
148,51,148,52,148,53,148,55,148,76,148,84
 780 DATA148,152,149,29,149,40,149,50,149,54,149,77,149,83,149,152,150,31,150,41
,150,77,150,89,150,83,150,152,151,41,151,75
 790 DATA151,76,151,83,151,151,152,30,152,42,152,74,152,75,152,80,152,84,152,151
,153,42,153,73,153,76,153,77,153,84,153,150
 800 DATA154,42,154,72,154,75,154,78,154,85,154,147,154,148,154,149,155,43,155,7
1,155,75,155,78,155,86,155,146,156,43,156,71
 810 DATA156,75,156,79,156,86,156,143,156,144,156,145,157,44,157,72,157,76,157,7
9,157,85,157,87,157,141,157,142,158,45
 820 DATA158,46,158,73,158,77,158,79,158,84,158,87,158,88,158,89,158,140,159,47,
159,72,159,77,159,80,159,83,159,87,159,90
 830 DATA159,91,159,92,159,139,160,47,160,71,160,76,160,81,160,82,160,87,160,93,
160,139,161,46,161,71,161,77,161,88,161,89
 840 DATA161,90,161,94,161,95,161,96,161,97,161,139,162,31
 850 DATA162,45,162,72,162,77,162,91,162,92,162,98,162,99,162,100,162,129,162,13
0,162,131,162,138,163,32,163,45,163,73
```

```
 860 DATA163,74,163,76,163,93,163,101,163,126,163,127,163,128,163,132,163,133,16
3,134,163,135,163,136,163,137,164,32,164,45
 870 DATA164,75,164,94,164,95,164,96,164,97,164,102,164,103,164,125,164,146,164,
147,165,32,165,45,165,98,165,99,165,104
 880 DATA165,124,165,144,165,145,165,148,166,31,166,44,166,100,166,105,166,106,1
66,121,166,122,166,123,166,140,166,141,166,142
 890 DATA166,143,166,148,167,31,167,44,167,101,167,102,167,103,167,107,167,120,1
67,138,167,139,167,148,168,30,168,37,168,38
 900 DATA168,39,168,40,168,44,168,104,168,108,168,119,168,137,168,147,169,36,169
,40,169,41,169,45,169,105,169,109,169,118
 910 DATA169,136,169,145,169,146,170,35,170,38,170,39,170,45,170,106,170,109,170
,117,170,136,170,143,170,144,171,35,171,37
 920 DATA171,45,171,90,171,107,171,109,171,116,171,135,171,141,171,142,172,35,17
2,36,172,44,172,89,172,91,172,106,172,108
 930 DATA172,115,172,135,172,140,173,44,173,74,173,75,173,76,173,89,173,92,173,1
06,173,108,173,113,173,114,173,135,173,139
 940 DATA174,44,174,74,174,77,174,78,174,89,174,92,174,105,174,108,174,112,174,1
36,174,137,174,138,175,40,175,41,175,45
 950 DATA175,73,175,79,175,80,175,90,175,93,175,104,175,109
 960 DATA175,110,175,111,176,40,176,42,176,43,176,44,176,73,176,75,176,76,176,81
,176,82,176,83,176,91,176,94,176,103,177,41
 970 DATA177,74,177,77,177,83,177,91,177,95,177,103,178,42,178,78,178,83,178,92,
178,95,178,103,179,43,179,44,179,45,179,79
 980 DATA179,82,179,93,179,103,180,46,180,80,180,81,180,93,180,95,180,103,181,41
,181,42,181,46,181,93,181,96,181,97,181,102
 990 DATA182,40,182,43,182,44,182,45,182,93,182,98,182,99,182,100,182,101,183,39
,183,94,184,39,184,94,185,38,185,94,186,30
1000 DATA186,37,186,95,187,31,187,37,187,96,187,97,188,32,188,38,188,98,189,32,
189,38,189,99,190,38,190,99,191,33,191,37
1010 DATA191,100,192,37,192,101,193,36,193,102,193,103,193,104,193,105,194,35,1
94,106,194,107,195,35,195,108,195,109,195,110
1020 DATA195,111,196,35,196,112,197,36,197,113,198,37,198,112,199,38,199,111,20
0,39,200,109,200,110,201,40,201,108,201,112
1030 DATA201,113,201,114,202,40,202,106,202,107,202,112,202,115,203,40,203,104,
203,105,203,113,203,114,203,115,204,40,204,103
1040 DATA205,39,205,101,205,102,206,39,206,100,207,39,207,100,208,39,208,99,209
,40,209,98,210,41,210,99,211,41,211,100
1050 DATA212,41,212,101,212,102,212,103,213,40,213,104,213,105,213,106,214,40,2
14,107,215,41,215,108,215,119,215,120,215,121
1060 DATA216,37,216,40,216,109,216,110,216,119,216,122,217,37
1070 DATA217,40,217,111,217,112,217,119,217,123,217,124,218,35,218,37,218,39,21
8,113,218,114,218,115,218,120,218,125,218,126
1080 DATA219,37,219,39,219,109,219,110,219,111,219,116,219,117,219,120,219,127,
220,37,220,40,220,108,220,112,220,113,220,118
1090 DATA220,121,220,127,221,34,221,36,221,41,221,109,221,114,221,115,221,116,2
21,119,221,122,221,123,221,127,222,40,222,110
1100 DATA222,117,222,118,222,124,222,125,222,126,223,40,223,111,223,112,224,40,
224,101,224,102,224,113,224,128,224,129,224,155
1110 DATA224,156,224,157,224,158,224,159,225,40,225,100,225,103,225,104,225,113
,225,129,225,150,225,151,225,154,225,160
1120 DATA225,161,225,162,226,40,226,100,226,105,226,106,226,107,226,108,226,113
,226,122,226,123,226,130,226,149,226,152
1130 DATA226,153,226,163,227,41,227,99,227,102,227,109,227,112,227,121,227,124,
227,130,227,148,227,163,228,41,228,99,228,101
1140 DATA228,103,228,110,228,111,228,120,228,125,228,130,228,131,228,148,228,16
3,229,42,229,79,229,80,229,81,229,98,229,102
1150 DATA229,119,229,126,229,131,229,147,229,162,230,42,230,57,230,58,230,78,23
0,82,230,98,230,117,230,118,230,127,230,131
1160 DATA230,147,230,162,231,42,231,56,231,59,231,78,231,83,231,84,231,97,231,1
16,231,127,231,146,231,161,232,42,232,56
1170 DATA232,60,232,78,232,85,232,86,232,96,232,112,232,113
1180 DATA232,115,232,125,232,126,232,127,232,146,232,161,233,41,233,55,233,61,2
33,79,233,87,233,88,233,89,233,90,233,95
1190 DATA233,111,233,115,233,124,233,131,233,145,233,160,234,41,234,55,234,61,2
34,80,234,81,234,82,234,91,234,92,234,93
1200 DATA234,94,234,116,234,117,234,118,234,123,234,144,234,159,235,41,235,55,2
35,62,235,83,235,84,235,106,235,107,235,108
1210 DATA235,113,235,119,235,120,235,121,235,122,235,132,235,144,235,159,236,41
,236,55,236,62,236,77,236,78,236,85,236,95
1220 DATA236,96,236,97,236,105,236,109,236,111,236,113,236,143,236,159,237,42,2
37,56,237,63,237,76,237,79,237,80,237,84
1230 DATA237,94,237,96,237,105,237,106,237,107,237,108,237,131,237,142,237,160,
238,42,238,56,238,64,238,65,238,66,238,67
1240 DATA238,74,238,75,238,81,238,82,238,83,238,94,238,95,238,111,238,112,238,1
41,238,160,239,42,239,52,239,53,239,54
1250 DATA239,55,239,68,239,69,239,72,239,73,239,111,239,113,239,141,239,161,240
,42,240,51,240,70,240,71,240,111,240,114
1260 DATA240,132,240,141,240,162,240,163,241,41,241,51,241,112,241,113,241,133,
241,140,241,164,241,165,242,41,242,52,242,53
```

```
1270 DATA242,54,242,55,242,56,242,83,242,84,242,133,242,139,242,166,242,167,243
,41,243,57,243,72,243,73,243,74,243,75
1280 DATA243,76,243,81,243,82,243,85,243,139,243,168,244,42
1290 DATA244,58,244,59,244,71,244,77,244,80,244,84,244,87,244,138,244,169,244,1
73,245,43,245,60,245,70,245,78,245,79
1300 DATA245,83,245,137,245,170,245,173,245,174,246,43,246,54,246,55,246,61,246
,70,246,71,246,72,246,73,246,82,246,85
1310 DATA246,137,246,171,246,173,246,174,247,43,247,53,247,56,247,62,247,69,247
,74,247,82,247,137,247,171,247,173,248,43
1320 DATA248,48,248,52,248,57,248,58,248,59,248,62,248,68,248,75,248,80,248,81,
248,138,248,139,248,140,248,141,248,171
1330 DATA249,44,249,47,249,49,249,50,249,51,249,60,249,61,249,76,249,77,249,78,
249,79,249,80,249,127,249,142,249,143,249,144
1340 DATA249,171,250,45,250,47,250,126,250,128,250,145,250,170,251,46,251,47,25
1,126,251,129,251,146,251,168,251,169,252,126
1350 DATA252,130,252,131,252,132,252,146,252,167,253,126,253,133,253,145,253,16
8,254,127,254,133,254,141,254,142,254,143
1360 DATA254,144,254,165,255,127,255,133,255,140,255,163,255,164,256,127,256,13
9,256,139,256,162,257,127,257,132,257,138
1370 DATA257,139,257,140,257,141,257,142,257,161,258,127,258,133,258,143,258,14
4,258,145,258,146,258,159,258,160,258,178
1380 DATA259,128,259,134,259,147,259,148,259,149,259,150,259,158,259,177,259,17
9,260,129,260,130,260,135,260,151,260,152
1390 DATA260,155,260,156,260,157,260,176,260,177,260,178
1400 DATA261,131,261,136,261,153,261,154,261,175,261,176,261,177,262,132,262,13
3,262,175,262,176,263,133,263,136,263,174
1410 DATA263,175,264,134,264,135,264,173,264,174,265,172,265,173,266,131,266,17
2,267,131,268,128,268,130,269,129,269,171
1420 DATA270,170,270,172,271,131,271,169,271,171,272,168,272,170,273,132,273,16
9,273,170.274,169,274,170,275,134,277,135
```

```
10 REM -- SINUSITIS
20 REM -- ARENS; 5 HEIDEPARK; WAGENINGEN-HOOG; NL-6705 AB HOLLAND
30 DIMY(250):CONSOLEC80,S0,24,N:GRAPHI1,C,O1
40 FORI=-12.5TO12.5STEP0.1
50 Y(125+(I*10))=SIN(I)*40
60 NEXT
70 Q=1:T=1
80 FORI=TTO250STEPQ
90 LINE I-1,Y(I-1)+100,I+19,Y(I-1)+100
100 BLINE I,Y(I)+100,I+20,Y(I)+100
110 NEXT
120 Q=INT(RND(2)*7)+1
130 T=INT(RND(1)*10)+1
140 GOTO80
```

```
10 REM -- KNIGHT'S TOUR ~A classic problem~
20 REM -- CC82; ARENS; 5 HEIDEPARK; WAGENINGEN-HOOG; NL-6705 AB HOLLAND
30 REM
40 REM INITIALIZATION
50 CONSOLEC80,S0,24,N:GRAPHC,I1,O1
60 DIMH(25,25),A(8),B(8),P0(8),P1(8)
70 A(1)=1:A(2)=2:A(3)=2:A(4)=1
80 A(5)=-1:A(6)=-2:A(7)=-2:A(8)=-1
90 B(1)=2:B(2)=1:B(3)=-1:B(4)=-2
100 B(5)=-2:B(6)=-1:B(7)=1:B(8)=2
110 PRINTCHR$(6):PRINT"                    KNIGHT'S TOUR PROBLEM "
120 PRINT:PRINT:INPUT" Enter size of board (max. 25)          : ";N
130 PRINT:INPUT" Enter starting location, X-coordinate: ";X1
140 PRINT:INPUT" Enter starting location, Y-coordinate: ";Y1
150 PRINT:INPUT" Printer ( Y / N )                     : ";SS$
160 IFSS$="Y"THENLPF=1:PRINT/PCHR$(20)
170 IFSS$="N"THENLPF=0
180 NMOS=1:A1=1:N1=N*N
190 REM
200 REM CLEAR BOARD
210 REM
220 FORI=1TON
230 FORJ=1TON
240 H(I,J)=0
250 NEXTJ:NEXTI
260 FORI=1TON
270 FORJ=1TON
280 X=I:Y=J:GOSUB880
290 NEXTJ:NEXTI
300 REM
```

```
310 REM MAKE FIRST MOVE
320 REM
330 X=X1:Y=Y1
340 H(X,Y)=1:L=2
350 GOSUB980
360 Q1=1:Q2=1
370 REM
380 REM ROUTINE THAT FINDS NEXT MOVE
390 REM
400 IF(Q1=0)+(Q2=0)THENGOTO750
410 Q1=0:Z1=X:Z2=Y:GOSUB1140
420 IFR0>1THENGOTO460
430 C1=T1:C2=T2
440 GOTO660
450 REM
460 REM EXAMINE ALL TIES
470 REM
480 FORK=1TOR0
490 P1(K)=P0(K)
500 NEXTK
510 R1=R0
520 IFA1=1THENGOTO540
530 M2=0:GOTO550
540 M2=-9
550 FORK=1TOR1
560 F1=P1(K):Z1=X+A(F1):Z2=Y+B(F1)
570 IFA1=1THENGOTO610
580 GOSUB1140
590 IFM2>=M1THENGOTO640
600 GOTO630
610 GOSUB1320
620 IFM1<=M2THENGOTO640
630 M2=M1:C1=Z1:C2=Z2
640 NEXTK
650 REM
660 REM NEXT MOVE FOUND; UPDATE BOARD
670 REM
680 IFQ1=0THENGOTO730
690 X=C1:Y=C2
700 H(X,Y)=L
710 GOSUB980
720 IFL=N1THENQ2=0
730 L=L+1
740 GOSUB1500:NMOS=NMOS+1:GOTO400
750 REM
760 REM KNIGHT'S TOUR OVER; PRINT RESULTS
770 REM
780 IFQ2=1THENGOTO830
790 PRINT:PRINT:PRINT" Solution found:":PRINT
800 IFLPF=1THENPRINT/P:PRINT/P:PRINT/P"SOLUTION FOUND:"
810 GOSUB1550:IFLPF=1THENPRINT/PCHR$(21)
820 PRINT:GOTO1680
830 IFA1=0THENGOTO850
840 A1=0:GOTO190
850 PRINT:PRINT" No solution.":IFLPF=1THENPRINT/PCHR$(21)
860 PRINT:GOTO1680
870 REM
880 REM BOARD INITIALIZATION ROUTINE
890 REM
900 FORI1=1TO8
910 U=X+A(I1):V=Y+B(I1)
920 IF(U>0)*(U<=N)THENGOTO940
930 GOTO970
940 IF(V>0)*(V<=N)THENGOTO960
950 GOTO970
960 H(U,V)=H(U,V)-1
970 NEXTI1:RETURN
980 REM
990 REM BOARD UPDATING ROUTINE
1000 REM
1010 B=1
1020 FORI1=1TO8
1030 U=X+A(I1):V=Y+B(I1)
1040 IF(U>0)*(U<=N)THENGOTO1060
1050 GOTO1110
1060 IF(V>0)*(V<=N)THENGOTO1080
1070 GOTO1110
1080 IFH(U,V)<0THENGOTO1100
1090 GOTO1110
1100 H(U,V)=H(U,V)+1:B=0
1110 NEXTI1
1120 IFB1=1THENQ1=0
```

```
1130 RETURN
1140 REM
1150 REM ROUTINE THAT SEARCHES FOR NEXT MOVE WITH MINIMUM WEIGHT
1160 REM
1170 M1=-9:R0=1
1180 FORW=1TO8
1190 U=Z1+A(W):V=Z2+B(W)
1200 IF((U>0)*(U<=N))*((V>0)*(V<=N))THENGOTO1220
1210 GOTO1300
1220 IF(H(U,V)<=0)*(H(U,V)>=M1)THENGOTO1240
1230 GOTO1300
1240 IFH(U,V)=M1THENGOTO1290
1250 M1=H(U,V):Q1=1
1260 T1=U:T2=V
1270 R0=1:P0(R0)=W
1280 GOTO1300
1290 R0=R0+1:P0(R0)=W
1300 NEXTW
1310 RETURN
1320 REM
1330 REM ROUTINE THAT SEARCHES FOR NEXT MOVE WITH MAXIMUM WEIGHT
1340 REM
1350 M1=0:R0=1
1360 FORW=1TO8
1370 U=Z1+A(W):V=Z2+B(W)
1380 IF((U>0)*(U<=N))*((V>0)*(V<=N))THENGOTO1400
1390 GOTO1480
1400 IF(H(U,V)<0)*(H(U,V)<=M1)THENGOTO1420
1410 GOTO1480
1420 IFH(U,V)=M1THENGOTO1470
1430 M1=H(U,V):Q1=1
1440 T1=U:T2=V
1450 R0=1:P0(1)=W
1460 GOTO1480
1470 R0=R0+1:P0(R0)=W
1480 NEXTW
1490 RETURN
1500 REM
1510 REM ROUTINE THATS PRINTS-OUT BOARD
1520 REM
1530 IFNMOS=1THENPRINTCHR$(6);" Move";NMOS:GOTO1550
1540 PPINT" Move";NMOS:IFLPF=1THENPRINT/P:PRINT/P:PRINT/P"MOVE";NMOS
1550 L2=N
1560 TAB=4:IFN>20THENTAB=3
1570 FORI2=1TON
1580 PRINT
1590 IFLPF=1THENPRINT/P
1600 FORL1=1TON
1610 PRINTH(L1,L2),
1620 IFLPF=1THENPRINT/PH(L1,L2),
1630 NEXTL1
1640 L2=L2-1
1650 NEXTI2
1660 PRINT:PRINT
1670 RETURN
1680 PRINT"Another run (Y/N) ?"
1690 GETA$:IFA$="Y"THENRUN
1700 IFA$<>"N"THEN1690
1710 PRINTCHR$(6):CLR:END
```

DSPR (Issue 13) Update from Mervyn Fry

In Issue 13 of the S.U.N. you printed my article about 'DSPR', a utility I had written for the MZ-80B. I received suggestions from several readers who tried this program. The result is a new utility confined to renumbering and appending BASIC programs. Though entirely in machine code it needs no knowledge of code, being called by a BASIC USR call and returning the user to BASIC always. Since its OPTIONS are Renumber, Append, Merge, Basic, it is called RAMB(U), the U denoting compatibility with versions v1.0 and v1.1 of Sharp BASIC 5510. The only inputs are 1st line number and Interval (in decimal), and unlike DSPR, GOTO (etc.) numbers can be in almost any (decimal) form up to 65535 (e.g. 5,05,00005,12345). Also RESTORE, which I forgot when writing DSPR, is included. It is a super RENUMBER/MERGE program for anyone not wishing to dabble in code so I offer it as being of wider interest than DSPR.

One puzzle has arisen which I have not yet solved, and I would welcome any advice readers can offer. After renumbering and/or appending, BASIC programs must be saved and re-loaded before they can be run. Otherwise the system may crash. I shall find the answer someday no doubt, but if it is obvious to someone will they please tell me. It does not affect the usefulness of the program.

I have updated the Disassembler/Search/Renumber program DSPR, by adding these improvements and a Zero option for clearing areeas of memory, so that it is no DSPRZBM(U). If anyone would like a tape of both programs I will supply one with instructions (and printout of the code if requested) for £4. Readers who got DSPR from me need ony send £2 (If they cannot afford that I will accept £1 to cover the costs of tape, jiffy bag, postage).

I enclose screen printouts illustrating the operation of RAMB(U). Is this too much for S.U.N?

Program 1 as loaded.

RAMB(U) in Operation

```
1 PRINTCHR$(6);"SILLY DEMO1"
20 PRINT"INPUT A,B or C"
31 INPUTX$:IF X$="A" THEN 77
40 IF1$="B" GOTO 9080
50 IF1$="C" THEN100
60 GOTO   00001
77 GOSUB 56210:GOTO20
80 PRINT"B for BANANA"
95 GOTO020
100 PRINT"INPUT A NUMBER 1-3"
110 GET$:IF(G(1)+(G)3) THEN  110
120 ON G GOSUB  1301,01402,11503:GOTO20
1301 PRINT"THE NUMBER WAS 1":RETURN
1402 PRINT"THE NUMBER WAS 2":RETURN
11503 PRINT"THE NUMBER WAS 3":RETURN
56210 PRINT"A for APPLE":RETURN
```

Program 1 renumbered.

```
20000 PRINTCHR$(6);"SILLY DEMO1"
20100 PRINT"INPUT A,B or C"
20200 INPUTX$:IF X$="A" THEN 20600
20300 IF1$="B" GOTO 20700
20400 IF1$="C" THEN20900
20500 GOTO   20000
20600 GOSUB 21500:GOTO20100
20700 PRINT"B for BANANA"
20800 GOTO20100
20900 PRINT"INPUT A NUMBER 1-3"
21000 GET$:IF(G(1)+(G)3) THEN  21000
21100 ON G GOSUB  21200,21300,21400:GOTO20100
21200 PRINT"THE NUMBER WAS 1":RETURN
21300 PRINT"THE NUMBER WAS 2":RETURN
21400 PRINT"THE NUMBER WAS 3":RETURN
21500 PRINT"A for APPLE":RETURN
```

Program 2 as loaded

after selecting Option A (Append)

```
 1 PRINTCHR$(6);"DEMO2"
 11 DIMA(3)
 120 DATA 1,2,3
 130 DATA 4,5,6
 145 PRINT"RESTORE DATA 1 or 2 ?"
 152 GETA:IFIA<1)+(A>2) THEN152
 160 ON A GOTO 0170,180
 170 RESTORE:GOTO  190
 180 RESTORE 130
 190 FOR J=1 TO 3:READ A(J):NEXT
 200 PRINT:PRINT"INPUT A,B orC":PRINT
 321 INPUTX$:IF X$="A" THEN 0700
 400 IFX$="B" GOTO  00800
 500 IFX$="C" THEN1000
 607 GOTO 200
 700 GOSUB 18000:GOTO200
 800 PRINT"B for BANANA":GOTO200
 1000 PRINT"INPUT A NUMBER 1-3"
 1100 GETG:IF(G<1)+(G>3) THEN1100
 1200 ON G GOSUB  1300,01400,12345:PRINT:PRINT:GOTO145
 1300 PRINT"A(1)= ";A(G):RETURN
 1400 PRINT"A(2)= ";A(G):RETURN
 12345 PRINT"A(3)= ";A(G):RETURN
 18000 PRINT"A for APPLE":RETURN
 65535 END
```

Programs merged after renumbering 2.

```
 20000 PRINTCHR$(6);"SILLY DEMO1"
 20100 PRINT"INPUT A,B orC"
 20200 INPUTX$:IF X$="A" THEN 20600
 20300 IFX$="B" GOTO 20700
 20400 IFX$="C" THEN20900
 20500 GOTO  20000
 20600 GOSUB 21500:GOTO20100
 20700 PRINT"B for BANANA"
 20800 GOTO20100
 20900 PRINT"INPUT A NUMBER 1-3"
 21000 GETG:IF(G<1)+(G>3) THEN  21000
 21100 ON G GOSUB  21200,21300,21400:GOTO20100
 21200 PRINT"THE NUMBER WAS 1":RETURN
 21300 PRINT"THE NUMBER WAS 2":RETURN
 21400 PRINT"THE NUMBER WAS 3":RETURN
 21500 PRINT"A for APPLE":RETURN
 21600 PRINTCHR$(6);"DEMO2"
 21700 DIMA(3)
 21800 DATA 1,2,3
 21900 DATA 4,5,6
 22000 PRINT"RESTORE DATA 1 or 2 ?"
 22100 GETA:IFIA<1)+(A>2) THEN22100
 22200 ON A GOTO 22300,22400
 22300 RESTORE:GOTO  22500
 22400 RESTORE 21900
 22500 FOR J=1 TO 3:READ A(J):NEXT
 22600 PRINT:PRINT"INPUT A,B orC":PRINT
 22700 INPUTX$:IF X$="A" THEN 23100
 22800 IFX$="B" GOTO  23200
 22900 IFX$="C" THEN23300
 23000 GOTO 22600
 23100 GOSUB 23900:GOTO22600
 23200 PRINT"B for BANANA":GOTO22600
 23300 PRINT"INPUT A NUMBER 1-3"
 23400 GETG:IF(G<1)+(G>3) THEN23400
 23500 ON G GOSUB  23600,23700,23800:PRINT:PRINT:GOTO22000
 23600 PRINT"A(1)= ";A(G):RETURN
 23700 PRINT"A(2)= ";A(G):RETURN
 23800 PRINT"A(3)= ";A(G):RETURN
 23900 PRINT"A for APPLE":RETURN
 24000 END
```

SCREEN DISPLAYS (RAMB(W))

```
OPTIONS
```

Fig 1. (Title page)

```
        R  RENUMBER
                A  APPEND
                        M MERGE. Use only after APPEND.

                                                        B BASIC

 [illegible] OPTION LETTER
```

```
 RENUMBER
```

Fig 2. (After chosing option R and responding to prompts).

```
FIRST LINE No. [illegible] ?
 Dec. 10
        OK ? (Y/N)

INTERVAL [illegible] ?
 Dec. 10
        OK ? (Y/N)

CONTINUE

OK ? (Y/N)
Ready
```

```
   APPEND
```

Fig3. (After choosing option A)

```
          (1) LOAD PROG.2

                          (2) [illegible] PROG.2

                                                (3) If needed RENUMBER PROG.

(4) FINALLY CALL USR($FF77) and select MERGE.
Ready
```

INSTRUCTIONS for RAMB(U) by M.Fry (MZ 80B)

1. Load Sharp Basic SB-5510 VERSION 1.0 or 1.1. Limit #LFFF
2. DEFKEY(n)=CONSOLECBO!USR($FF77)↵↵(GRPHandSFTLOCK together).
3. Enter MON and L to load the machine code RAMB(U). Basic sign 'Ready' will appear.
4. LOAD a Basic program (PROG1). LIST it right to the end before calling RAMB.
5. Use KEY(n) to call RAMB. Select option required:
RENUMBER asks for 1st Line No. and Interval, both decimal. GOTO (etc) line numbers can be up to 65000plus (<$FFF) in any format (ex 00005,05,5) up to 5 digits. Leading 0's will be deleted.

APPEND gives instructions. It 'forgets' PROG1 by setting the start of the Basic program area. Load PROG2 normally. Edit it if you wish. LIST it to the end before calling RAMB.
MERGE resets program area, merging PROG1 and PROG2.

At every stage RAMB returns to Basic command level.

6. After renumbering etc SAVE the Basic program BEFORE ATTEMPTING TO RUN IT or it may crash. When re-loaded it will RUN normally.

To make a back-up copy of RAMB(U):-Enter MON and S. Enter S-ADR $FB10, E-ADR $FFDF, J-ADR $FB10, once you have a tape ready.

Please let me know if you find any 'bugs', or difficulty.

Mervyn Fry.

INSTRUCTIONS for DSPRZBM(u)■ by M.Fry. April 1985. (MZ-80B)

1. Load Sharp Basic SB-5510; version 1.0 or 1.1.
2. DEFKEY(n1)=USR($FCAB)+↓.(+↓isGRPH+SFTLOCK together).
   If reduced size printout required:-
   DEFKEY(n2)=PRINT/PCHR$(17);CHR$(20)!COPY/P1
3. Enter MON and L to load DSPRZBM(U) machine code. Direct command mode is entered and Options displayed. Touch initial key of option required.

D. Requests a $START ADDRESS. Displays the 20 Op.Instructions, with choice of C (printer),Y (another 20 Op codes), or N (Options). C(opy) jumps to Basic, prints the command COPY/P1 below the list. Move cursor to it and press CR to print the 20 instructions. Use Key n2 to overprint the command COPY/P1 for reduced size. Recall Options with Key n1.

S. Requests a $ADR. (2bytes eg $FCAB), a $START and $FINISH. If the 2bytes are NOT an address (eg FE0D, code for CR 0D) enter them in reverse (0DFE). The address of every occurrence of those bytes (ABFC or FE0D) between START and FINISH will be listed.

P. Sets the start of Basic program area. Enables another program to be loaded and edited normally, to be appended when compatible. To change POINTERS you need to know the address of first byte after the end of your Basic program. Use M. (Monitor mode) to examine the program area for this. Or,if program has been listed to the end, examine $ADR 4A0E,F (v1.0) or 4ACE,F (v1.1).

R. LIST a program right to the end before calling DSPRZBM(U) to RENUMBER it. R. requests only 1st Line No. and Interval both decimal. To merge the two programs use P. and enter the original program start ($505C for v1.0 or $511C for v1.1).
SAVE renumbered programs and RELOAD before attempting to RUN them.

Z. Enter $START and $FINISH. Be CAREFUL. Touching Y in response to the next prompt instantly zeros the whole memory between those addresses. You could wipe out a large chunk of DSPR or Basic.

B. Transfers command to Basic.

M. Transfers command to Monitor. Equivalent to MON in Basic.

To SAVE DSPRZBM(U)4/85:-S-Adr$FZ49. E-Adr$FFCF. J-Adr$FFB7.

LITTLE BEN

A. O. Lucas

This program runs on SHARP MZ80B with Basic SB5510, on Disk Basic 6510, and with little modification on other standard Basic programs.

After loading, the progaram demands:-

"Border....". You should enter a character - anyone will do - it will be used to frame the clock.

The clock face is drawn (lines 60 to 330). The time shown on the clock can be reset at any time by pressing the the key "R". **RESET TIME** appears to the right of the clock. You should enter the time remebering the format of TI$:-

HHMMSS

For example:-

(a) 5.42am 054200

(b) 10.54pm 225400

(c) 10.45(7seconds)am .....104507.

Other time zones can be obtained by pressing the initial letter for the town e.g.

LONDON	L
BANGKOK	B
ANCHORAGE	A

The time zones in this program are based on Geneva time i.e. G.M.T. +1. By changing names of the town and the difference in hours from GMT or some other reference time zone, you can introduce your own list of towns.

```
10 REM  LITTLE BEN A. D. LUCAS GENEVA
20 PRINTCHR$(6):CONSOLEN:PRINT :PRINT"CHOOSE ANY CHARACTER AS THE BORDER"
30 PRINT"FOR THE CLOCK":PRINT:PRINT:PRINT
40 INPUT "BORDER....";B$
50 PRINTCHR$(6):CONSOLE C40
60 GRAPHC,01,I1,01
70 AN$="SAME DAY"
80 FOR R=2 TO 23:CURSOR R,1:PRINT B$:CURSOR R,23:PRINT B$:NEXT
90 FOR C=1 TO 23:CURSOR 1,C:PRINT B$:CURSOR 23,C:PRINT B$:NEXT
100 IF((VAL(TI$)<60000)+(VAL(TI$)>180000))THEN CONSOLE N: GOTO 0120
110 CONSOLE R
120 CURSOR3,3:PRINT"♣":CURSOR 3,21:PRINT"♦":CURSOR21,3:PRINT"♥":CURSOR 21,21:PR
INT"♠"
130 FOR W=16 TO 184 STEP 0.5
140 W1=ABS(100-W)
150 W2=SQR(ABS(84-W1)*(84+W1))
160 W3=100+W2
170 W4=100-W2
180 SET W,W3
190 SET W,W4
200 NEXT W
210 K=π/180
220 CURSOR 17,4  :PRINT"1"
230 CURSOR 20,8  :PRINT"2"
240 CURSOR 21,12 :PRINT"3"
250 CURSOR 20,16 :PRINT"4"
260 CURSOR 17,20 :PRINT"5"
270 CURSOR 12,21 :PRINT"6"
280 CURSOR 7,20  :PRINT"7"
290 CURSOR 4,16  :PRINT"8"
300 CURSOR 3,12  :PRINT"9"
310 CURSOR 4,8   :PRINT"10"
320 CURSOR 7,4   :PRINT"11"
330 CURSOR 12,3  :PRINT"12"
340 REM - CHIMES
350 DIM A$(4)
360 A$(1)="B4AGD7"
370 A$(2)="D4ABG7"
380 A$(3)="B4GAD7"
390 A$(4)="D4ABG7"
400 TEMPO6
410 X0=100:Y0=100:X=0:Y=0
420 REM - POSITION OF HR,MIN,SEC ARMS
430 K=π/180
440 S=VAL(RIGHT$(TI$,2))*6
450 IFABS(S-S1)> 1 THEN GOSUB 0660
460 X=65*SIN(S*K):Y=65* COS(S*K)
470 X1=58*SIN(S*K):Y1=58* COS(S*K)
480 LINE X0+X1,Y0-Y1,X +X0,Y0-Y
490 S1=S
500 GET G$:IFG$<>"" THEN GOSUB 0850
510 M=VAL(MID$(TI$,3,2))*6
520 IFABS(M-M1)>1 THEN GOSUB 0690
530 XM=55*SIN(M*K):YM=55* COS(M*K)
540 LINE X0,Y0,XM+X0,Y0-YM
550 M1=M
560 IF Z<0  THEN 0580
570 IF  (M-(INT(M/90))*90)<.00001 THEN IF M<>0 THEN  GOSUB 0810
580 H=(VAL(LEFT$(TI$,2))+M/360)*30
590 IFABS(H-H1)>.25THEN GOSUB 0720
600 XH=40*SIN(H*K):YH=40* COS(H*K)
```

```
 610 LINE XO,YO,XH+XO,YO-YH
 620 H1=H
 630 IF Z<0  THEN 0650
 640 IFM=0THENIFS=0 THEN  GOSUB 0740
 650 GOTO 0440
 660 BLINE XO+X1,YO-Y1,X +XO,YO-Y
 670 IF((VAL(TI$)<60000)+(VAL(TI$)>180000))THEN CONSOLE N:RETURN
 680 CONSOLE R: RETURN
 690 BLINE XO,YO,XM+XO,YO-YM
 700 Z=Z+1
 710 RETURN
 720 BLINE XO,YO,XH+XO,YO-YH
 730 RETURN
 740 MUSIC A$(1),A$(2),A$(3),A$(4)
 750 HR=(VAL(LEFT$(TI$,2)))
 760 IF HR>12 THEN HR=HR-12
 770 FOR P=1 TO HR:MUSIC" R5G5" :NEXT P
 780 Z=-5
 790 IF((VAL(TI$)<60000)+(VAL(TI$)>180000))THEN CONSOLE N:RETURN
 800 CONSOLE R:RETURN
 810 REM  SOUNDING CHIMES
 820 FOR MU=1 TO M/90:MUSIC A$(MU):NEXT
 830 Z=-5
 840 RETURN
 850 CURSOR25,10:PRINTSPACE$(12):CURSOR25,12:PRINTSPACE$(8):CURSOR 25,14:PRINTSP
ACE$(12):IFG$<>"R" THEN 0890
 860 CURSOR 25,10: PRINT"RESET TIME "
 870 CURSOR25,12:INPUTTI$:Z=0
 880 RETURN
 890 RESTORE 980
 900 READ A$,C:IFA$="END" THEN RETURN
 910 IF LEFT$(A$,1)<> G$ THEN 0900
 920 ZO=VAL(LEFT$(TI$,2))+C:IFZO<0 THEN ZO=ZO+24
 930 AN$="SAME DAY": IFZO<0 THEN ZO=ZO+24:AN$="PREVIOUS DAY"
 940 IF ZO>23 THEN ZO=ZO-24:AN$="NEXT DAY"
 950 CURSOR 25,10: PRINTA$
 960 CURSOR25,12:PRINT ZO;":";MID$(TI$,3,2):CURSOR 25,14:PRINTAN$
 970 RETURN
 980 DATA "LONDON ",-1
 990 DATA "NEW YORK ",-6
1000 DATA "IBADAN ",0
1010 DATA "KARACHI ",4
1020 DATA "MOSCOW ",2
1030 DATA "SYDNEY ",10
1040 DATA "TOKYO ",8
1050 DATA "BANGKOK",6
1060 DATA "HONGKONG ",7
1070 DATA "ZURICH ",0
1080 DATA "JAKARTA ",6
1090 DATA "TAIPEI ",9
1100 DATA "PARIS ",0
1110 DATA "DUBAI ",3
1120 DATA "CAIRO ",1
1130 DATA "ANCHORAGE ",-11
1140 DATA "GDANSK ",0
1150 DATA "HELSINKI ",1
1160 DATA "WASHINGTON ",-6
1170 DATA "END",-999
```

A SERIOUS DATABASE FOR THE MZ-80A

From P.J. RAWSON

Thank you for starting to include program listings for more serious applications in your recent issues of SUN.

Whilst I would not venture to suggest that the SHARO MZ-80A/K is a machine ideally suited for full business applications, it does show that with a little ingenuity it can be coaxed into doing a bit more than zapping aliens.

With this in mind I enclose some notes and a program listing for the implementation of a small database on the MZ-80A.

AND FINALLY...

It can be a daunting task to key in a complex program such as this. I would never have the courage to sit down and keep bashing at the keys until it is finished as there would almost certainly be multiple mistakes and frequent unexplained crashes especially in the machine code sections.

My plan would be to use a modular approach. Enter and test one section at a time, creating dummy records and indexes to test the progress of the program.

As presented the USR routines are loaded from disc at the start of the program.

These were written and tested using PAMON, the finished code being tranferred to disc using tape to disc utility provided by SA-6510.

An alternative approach would be to incorporate the HEX code into DATA statements within the BASIC program and POKE this separately into memory at the start of the program.

I have used this program on my MZ-80A for several months now without encountering any major problems, although I am not sure whether the index will fill up the available memory or whether the disc space will fill up first.

There is little doubt that the program could be improved in a variety of ways but, as a simple do it yourself Database Manager, it should prove more than adequate for most needs.

P.J. RAWSON
MANCHESTER

INFORMATION RETRIEVAL SYSTEM

A Micro Database for the SHARP MZ-80A

By P.J.Rawson July 1985

The compilation and processing of data records is a subject lacking in glamour and apart from its use by business and commerce remains a mainly neglected area.

There are of course many excellent commercially produced programs which cater for business users but for the small system user they tend to suffer from high cost and invariably offer features far in excess of the users requirements.

Information Retrieval System is a small database management program for the Sharp MZ-80A running Sharp Basic SA-6510.

The program specification provides the following features:

a) Create a data file by specifying the number of fields per record and entering the data from the keyboard

b) Save the file to disc under an appropriate Filename

c) Retrieve and display a selected record or records on inputting a searchkey

d) Reference several data records with one keyword

e) Allow several keywords to reference one data record

f) Provide the facility to make additions, deletions and changes to any record

The program uses an Index to keep track of the main Data File, this index being loaded into memory at the start of program execution. Any changes to the Data File are also noted in the Index and this is written back to disc when the program ends.

The index is an unordered list of keywords and record numbers, a linear search being made by a machine code routine for the required keyword.

If the specified keyword is found, the associated record numbers are passed back to the BASIC program to allow a standard Random File access routine to retrieve and display the requested data record.

The data file and the index must always correspond exactly and to maintain this relationship the program must do a lot of housekeeping.

If this were all to be done in BASIC it could take several minutes to search and modify an index for a database of a few hundred records. Using machine code, the index is updated almost instantaneously and the search routine is able to return a record number back to BASIC in less than a second.

THE BASIC PROGRAM

The BASIC program is written as a series of modules and with the REM statements removed (wihich must be done to avoid Insufficient Memory errors) runs in about 6K.

The enclosed program listing will be largely self explanatory so the following notes are restricted to the more obscure points especially where BASIC has been bent a little.

The machine code routines are held on disc as a seperate OBJ. file under the filename INFO/USR's.
Line 40 loads this file into reserved memory from address $9000 onwards.
The USR routine in Line 50 relocates the code to the normally wasted space at the end of the Basic Interpreter.

Following initialisation, the user is given the option of either interrogating an exsiting data file or creating a totally new one.

In either case the user is asked to enter a Filename and specify the active disc drive number.
Line 180 provides a means of specifically selecting the required disc drive without having to alter the program.

If a new file is being created the number of fields per record is also requested and Line 400 sets the initial length of the in memory Index file to zero bytes.

When the requirement is to interrogate an existing file, the program firstly checks that the specified file exists and if not, offers the opportunity to create it.

For reasons which will be explained later the Index file is not stored on disc as a conventional (Mode 3) data file.
It is written as a (Mode 1) object file which is loaded into memory in Line 520.

Lines 530-600 display the main menu and from here on the remainder of the program merely passes values between the machine code routines and the user input responses.

The data retrieval module, Lines 790-920) provides a sub menu of options to enable (A)lterations to be made to individual records, for (D)eletions and to (P)rint hardcopy of the requested record.
The option (N)ext is used to continue searching the index when more than one data record is referrenced by the same keyword.

In order to maintain the relationship between the Index and the Data file it is essential that the program terminates via the routine starting in Line 1120.
The USR routine in Line 1130 writes the Index back to disc and the series of POKES in lines 1140 and 1150 restore BASIC to its original state.

THE MACHINE CODE ROUTINES

As mentioned earlier the program needs to do a lot of housekeeping on the Index file and there is no way that BASIC alone can be made to work fast enough.

A HEX dump of the various machine code routines is enclosed and the following attempts to describe the function of the main routines.

USR($5895) : This routine takes characters from the keyboard and places them in an input buffer created at address $5F00.
Unlike the input routines in BASIC null strings are rejected.

USR($58AC),HL : This searches the Index from a specified location, passed from the variable HL, for a match with the keyword entered via the routine USR(5895).
On return to BASIC a value is passed back to HL, either Zero if no match is found or the memory address of the first record number associated with the desired keyword.
It will also locate sub strings of keywords if present.
For example, if the target keyword entered were PET then the following index keywords would be found if present: PETER PETROL PET ...etc.

USR($58E1) : This moves a keyword from the input buffer to the end of the Index and adjusts the two bytes at $9000 & $9001 indicating the length of the index.
It is called when a record is added to the main data file and the keyword entered does not already exist in the index.

USR($58FC),FF : This is used to obtain the logical record numbers from the Index and return them to BASIC in the variable FF.

USR($5914),HL,RN : This is used by the Add a Record module and inserts the logical record number in RN into the Index at location HL.

USR($597D),CC : When a data record is deleted all references to it in the index must also be deleted.
This routine searches the index for all occurrences of record number CC, counts and deletes them.

USR($5A14),FF : Used by the Display Keywords module this simply prints out the ASCII characters from the index to the VDU.
A return to BASIC is made with the address of the first record number after the keyword and this is used as the argument for USR($58FC) to obtain the logical record numbers associated with this keyword.

USR($593A),FF$: This routine writes the Index back to disc at the end of the program giving it the filename as specified by FF$.
Early attempts to write the index to disc using the normal PRINT# command proved somewhat unsatisfactory.
Extra control characters kept appearing in the file which had the effect of chopping the index into pieces.
The problem was solved by borrowing various of the I/O routines from the Basic Interpreter and dumping an entire section of memory to disc in the form of an object file.

```
1 REM ***   A MICRO DATABASE PROGRAM FOR THE SHARP MZ-80A WITH
2 REM ***    TWIN DISC DRIVES,PRINTER AND SHARP BASIC SA-6510
3 REM
4 REM ***       WRITTEN BY P.J.RAWSON          JULY 1985
5 REM
6 REM
7 REM ***  DELETE ALL REM STATEMENTS BEFORE RUNNING THE PROGRAM
8 REM
9 REM
10 LIMIT$8FFF:PRINT"        INFORMATION RETRIEVAL SYSTEM"
20 PRINT"         ______________________________________"
30 CURSOR6,10:PRINT"INITIALISING.....Please Wait"
39 REM *** LOAD M/C ROUTINES FROM OBJ. FILE INFO/USR's
40 LOAD"INFO/USR's"
48 REM *** BLOCK MOVE USR's TO BASIC WORK AREA DISABLE BREAKKEY
49 REM *** AND MODIFY BASIC TO ALLOW INPUT OF NULLS AND COMMAS.
50 USR($9225):POKE$313C,$00:POKE$3266,$00:POKE$1BBA,$00
60 POKE$1D2C,$18:POKE$3130,$18:NF=0
70 ON ERROR GOTO1310
77 REM
78 REM *** THE PRIMARY OPTIONS
79 REM
80 CURSOR4,10:PRINT"TYPE [1] TO CREATE A NEW FILE "
90 PRINT"            [2] TO READ EXISTING FILE"
100 Z$="TYPE":MX=3:CH=4:CV=10:GOSUB110:PRINT"":ONYGOTO390,440
106 REM
107 REM *** SUBROUTINES AT FRONT OF PROGRAM TO SPEED UP EXECUTION
108 REM
109 REM *** SUBROUTINE TO FLASH INPUT PROMPT
110 CURSORCH,CV:PRINTZ$:FORI=1TO40:GETA$:Y=VAL(A$)
120 IF(Y>0)*(Y<MX)THENRETURN
130 NEXT:IFZ$="TYPE"THENZ$="    ":GOTO110
140 IFZ$="    "THENZ$="TYPE":GOTO110
149 REM *** SUBROUTINE FOR ENTRY OF FILENAME & ACTIVE DISC DRIVE
150 INPUT"ENTER FILE NAME ";F$:IFF$=""THEN150
160 FF$=F$+"/IND"
170 INPUT"ENTER DISC DRIVE # ";FD$
180 FD=VAL(FD$):IF(FD>0)*(FD<3)THENPOKE$3819,0:POKE$381A,FD-1:RETURN
190 GOTO170
198 REM *** SUBROUTINE TO ADD A RECORD TO DATA FILE AND TO
199 REM *** FILL UP ANY SPACE CAUSED BY PREVIOUS DELETIONS
200 XOPEN#1,F$:IFZB=0THENRN=ZE:ZE=ZE+1:GOTO220
210 RN=ZB:DR=DR-1:INPUT#1(2+(RN-1)*FL),DM$
220 GOSUB270:RR=RR+1:IFZB=0THENRETURN
230 ZB=VAL(MID$(DM$,2,4)):RETURN
239 REM *** SUBROUTINE TO GET RECORD FROM DISC & OUTPUT TO VDU
240 XOPEN#1,F$:FORI=1TOFL:INPUT#1(2+(RN-1)*FL+I-1),A$(I)
250 PRINTA$(I):NEXT:CLOSE#1:RETURN
259 REM *** SUBROUTINE FOR OPTIONAL OUTPUT TO PRINTER
260 FORI=1TOFL:PRINT/PA$(I):NEXT:PRINT/P:PRINT/P:RETURN
270 FORI=1TOFL:PRINT#1(2+(RN-1)*FL+I-1),A$(I):NEXT:CLOSE#1:RETURN
278 REM *** SUBROUTINE TO DELETE A RECORD FROM DATA FILE AND
279 REM *** IDENTIFY FREED SPACE FOR LATER RE-USE
280 XOPEN#1,F$:IFZB=0THEN300
290 PRINT#1(2+(RN-1)*FL),"*"+STR$(ZB):GOTO310
300 PRINT#1(2+(RN-1)*FL),"*0000"
310 ZB=RN:DR=DR+1:RR=RR-1:CLOSE#1:RETURN
```

```
319 REM *** SUBROUTINE TO ACCEPT DATA ENTRY FROM KEYBOARD
320 PRINT"■■** MAX 32 Characters Per Field■":FORI=1TOFL
330 PRINT"FIELD #";I;:INPUTA$(I)
340 IFLEN(A$(I))<32THEN360
350 PRINT"FIELD TOO LONG , REDO":GOTO330
360 NEXT:PRINT"■■IS THIS RECORD OK? (Y/N)
370 GETA$:IF(A$<>"Y")*(A$<>"N")THEN370
380 RETURN
387 REM
388 REM *** PRIMARY OPTION [1]
389 REM *** BUILD FILESPEC & WRITE INITIAL PARAMETERS TO DISC
390 PRINT"CREATE A NEW FILE"
400 POKE$9000,$01:POKE$9001,$90:GOSUB150
410 INPUT"■ENTER # OF FIELDS PER RECORD ";FL$
420 FL=VAL(FL$):IF FL<1THEN410
430 RR=0:ZB=0:ZE=1:DR=0:GOSUB1170:DIMA$(FL):GOTO530
437 REM
438 REM *** PRIMARY OPTION [2]
439 REM *** VERIFY THAT FILE EXISTS & IF NOT OFFER TO CREATE IT
440 PRINT"READ FILE FROM DISC"
450 GOSUB150:PRINT"■■SEARCHING FOR FILE...."
460 GOSUB1230:IFRR>0THENDIMA$(FL):GOTO510
470 PRINT"■■FILE DOES NOT EXIST...■"
480 PRINT"     DO YOU WISH TO CREATE IT? (Y/N)"
490 GOSUB370:IFA$="Y"THENPRINT"■":GOTO390
499 REM *** IF OFFER DECLINED TERMINATE PROGRAM
500 DELETEF$:GOTO1140
510 PRINT"■■■■LOADING FILE INDEX...."
519 REM *** INDEX FILE IS LOADED AT BASE ADDRESS $9000
520 NF=1:LOADFF$
527 REM
528 REM *** THE MAIN PROGRAM MENU & OPTIONS
529 REM
530 PRINT"■■■■■      INFORMATION RETRIEVAL SYSTEM"
540 PRINT"     ──────────────────────────────"
550 PRINT"■■■■   FILENAME : ";F$
560 CURSOR3,12:PRINT"TYPE [1] TO ADD A RECORD"
570 PRINTTAB(8);"■[2] TO LOOKUP/MODIFY A RECORD"
580 PRINTTAB(8);"■[3] TO CHECK SYSTEM STATUS"
590 PRINTTAB(8);"■[4] TO LIST THE KEYWORDS"
600 PRINTTAB(8);"■[5] TO EXIT THE PROGRAM"
609 REM *** GET DESIRED OPTION AND BRANCH TO IT
610 CH=3:CV=12:MX=6:GOSUB110:PRINT"■":ONYGOTO620,790,980,1060,1120
615 REM
616 REM *** ADD A RECORD OPTION
617 REM
618 REM *** THE RECORD TO BE ADDED IS EITHER PUT AT THE HIGHEST
619 REM *** RECORD # +1 OR IN THE SPACE FROM PREVIOUS DELETIONS
620 IFZB=0THENRN=ZE:GOTO640
630 RN=ZB
640 PRINT"■DATA ENTRY FOR RECORD #";RN
650 GOSUB320:IFA$<>"Y"THEN620
660 GOSUB200
```

```
669 REM *** KEYWORD INPUT & CROSSREFERENCES SET UP IN THE INDEX
670 PRINT"■ENTER YOUR KEYWORD      ■"
680 USR($5895)
690 HL=36866:USR($58AC),HL
700 IFHL=0THENUSR($58E1):GOTO690
710 FF=HL
718 REM *** IF KEYWORD ENTERED IS A SUBSTRING OF ANOTHER KEYWORD
719 REM *** FOR THIS RECORD IT IS NOT ENTERED INTO THE INDEX
720 USR($58FC),FF:IFFF=RNTHEN530
730 IFFF<>0THENFF=FF+2:GOTO720
740 USR($5914),HL,RN
750 PRINT"■■■";SPACE$(119)
760 PRINT"■■■ANOTHER KEYWORD (Y/N)
770 GOSUB370:IFA$="Y"THENPRINT"■";:GOTO670
780 GOTO530
784 REM
785 REM *** LOOKUP/MODIFY A RECORD OPTION
786 REM
787 REM *** THIS OPTION SEARCHES THE INDEX FOR A GIVEN KEYWORD
788 REM *** OBTAINS THE RECORD NUMBER ASSCOCIATED WITH IT AND
789 REM *** GETS THE RECORD FROM DISC AND DISPLAYS IT ON THE VDU
790 PRINT"KEY OF RECORD REQUIRED■"
800 USR($5895):HL=36866
809 REM *** GET LOCATION OF FIRST RECORD # OR 0 IF NOT FOUND
810 USR($58AC),HL
820 IFHL=0THENPRINT"■SEARCH COMPLETED....":GOSUB1270:GOTO530
829 REM *** EXTRACT RECORD NUMBER OR 0 IF NO MORE ENTRIES
830 FF=HL:USR($58FC),FF:IFFF=0THENGOTO810
839 REM *** GET AND DISPLAY RECORD
840 RN=FF:PRINT"■":GOSUB240
849 REM *** SUB-MENU OPTIONS
850 PRINT"■■■ (N)ext  (F)inish  (D)elete  (A)lter"
860 PRINT"■                    (P)rint"
870 GETA$:IFA$=""THEN870
880 IFA$="N"THENHL=HL+2:GOTO830
890 IFA$="F"THEN530
900 IFA$="A"THEN950
910 IFA$="P"THENGOSUB260:GOTO870
920 IFA$<>"D"THEN870
928 REM *** IF A RECORD IS DELETED EXTENSIVE HOUSEKEEPING IS
929 REM *** NEEDED THIS IS THE FUNCTION OF USR($597D)
930 GOSUB280:POKE$597B,$00:POKE$597C,$00:CC=RN:USR($597D),CC
940 PRINT"■RECORD DELETED....":GOSUB1270:GOTO530
948 REM *** ALTERATION OF RECORDS IS ALLOWED BUT KEYWORDS CANNOT
949 REM *** BE CHANGED AS THEY MAY REFERENCE OTHER RECORDS
950 PRINT"■TYPE AMMENDED DATA":GOSUB320
960 IFA$="N"THEN950
970 XOPEN#1,F$:GOSUB270:GOTO530
976 REM
977 REM *** FILE STATUS OPTION
978 REM
979 REM *** THIS PROVIDES A SIMPLE DISPLAY OF SELECTED VARIABLES
980 EI=PEEK($9000)+PEEK($9001)*256:FM=53247-EI
990 PRINT"FILE STATUS■■"
1000 PRINT"FILE NAME  :";F$
1010 PRINT"■■NUMBER OF RECORDS REGISTERED:";RR
1020 PRINT"■FIELDS USED PER RECORD:";FL
1030 PRINT"■NUMBER OF DELETED RECORDS:";DR
1040 PRINT"■ROOM LEFT FOR INDEX:";FM;" BYTES■■■"
1050 GOTO1280
```

```
1056 REM
1057 REM *** LIST KEYWORDS OPTION
1058 REM
1059 REM *** THE LISTING CAN BE HALTED BY PRESSING THE SPACEBAR
1060 PRINT"■":EI=PEEK($9000)+PEEK($9001)*256:FF=36868
1070 IFFF>EITHENPRINT"■■■ISTING FINISHED";:GOTO1280
1080 USR($5A14),FF:FF=FF+1:PRINTTAB(20);
1090 HL=FF:USR($58FC),HL
1100 FF=FF+2:IFHL<>0THENPRINTHL;:GOTO1090
1110 PRINT:GOTO1070
1113 REM
1114 REM *** TERMINATE PROGRAM ROUTINE
1115 REM
1116 REM *** IT IS VITAL THAT THE PROGRAM TERMINATES VIA THIS
1117 REM *** SECTION IF THE INDEX/DATABASE IS TO BE MAINTAINED
1118 REM *** THE UPDATED INDEX & FILE HEADER ARE WRITTEN BACK TO
1119 REM *** DISC & THE ORIGINAL BASIC AND MEMORY SIZE RESTORED
1120 PRINT"■■■REWRITING FILE HEADER & INDEX":IFNF=1THENDELETEFF$
1130 GOSUB1170:USR($593A)FF$
1140 POKE$313C,$0D:POKE$3266,$0D:POKE$1BBA,$2C
1150 POKE$1D2C,$20:POKE$3130,$20
1160 PRINT"■":CURSOR8,10:PRINT"    PROGRAM TERMINATED":LIMITMAX:END
1167 REM
1168 REM *** CONSTRUCT AND WRITE FILE HEADER TO DISC
1169 REM
1170 D$="0000":RR$=STR$(RR):RR$=LEFT$(D$,4-LEN(RR$))+RR$
1180 ZB$=STR$(ZB):ZB$=LEFT$(D$,4-LEN(ZB$))+ZB$
1190 ZE$=STR$(ZE):ZE$=LEFT$(D$,4-LEN(ZE$))+ZE$
1200 FL$=STR$(FL):FL$=LEFT$(D$,4-LEN(FL$))+FL$
1210 DR$=STR$(DR):DR$=LEFT$(D$,4-LEN(DR$))+DR$:D$=RR$+ZB$+ZE$+FL$+DR$
1220 XOPEN#1,F$:PRINT#1(1),D$:CLOSE#1:RETURN
1227 REM
1228 REM *** READ HEADER FILE AND INITIALISE SYSTEM VARIABLES
1229 REM
1230 XOPEN#1,F$:INPUT#1(1),D$:CLOSE#1
1240 RR=VAL(LEFT$(D$,4)):ZB=VAL(MID$(D$,5,4))
1250 ZE=VAL(MID$(D$,9,4)):FL=VAL(MID$(D$,13,4))
1260 DR=VAL(MID$(D$,17,4)):RETURN
1267 REM
1268 REM *** SHORT DELAY FOR MESSAGE DISPLAY
1269 REM
1270 MUSIC"R7":RETURN
1277 REM
1278 REM *** WAIT FOR A SINGLE KEYPRESS
1279 REM
1280 PRINT" PRESS ANY KEY FOR MENU"
1290 GETA$:IFA$=""THEN1290
1300 GOTO530
1307 REM
1308 REM *** ERROR DISPOSAL ROUTINES
1309 REM
1310 PRINT"■■ERROR CONDITION.....■"
1315 IFERN=6THENPRINT"INSUFFICIENT MEMORY CAPACITY":END
1320 IFERN=55THENPRINT"NO MORE SPACE ON DISC":RESUME1120
1330 IFERN=50THENPRINT"DISC DRIVE #";FD;" NOT READY":GOTO1360
1340 IFERN=65THENPRINT"■PRINTER NOT ON LINE":RESUME870
1350 IFERN=67THENPRINT"■PRINTER OUT OF PAPER"
1360 PRINT"■CORRECT ERROR & PRESS (CR) TO CONTINUE"
1370 GETA$:IFA$=""THEN1370
1380 RESUME
```

INFORMATION RETRIEVAL SYSTEM

MACHINE CODE ROUTINES

The following is a HEX dump of the machine code routines used by the Basic program. The code is stored on disc as an OBJ. file under the name INFO/USR's. The load address is $9000 and a call to $9225 relocates the code to within the Basic Interpreter.

```
9000  CD 55 19 2C 2B 58 CD 48 20 D5 CD 58 19 2C 29 58
9010  CD 48 20 D5 CD 58 19 2C 2A 58 CD 48 20 D5 18 03
9020  E5 E5 E5 C1 D1 00 00 00 E1 C9 00 00 00 00 00 00
9030  00 00 00 00 00 00 00 00 22 32 58 ED 53 34 58 ED
9040  43 36 58 CD 55 19 2C 66 58 22 51 67 2A 32 58 CD
9050  67 58 2A 34 58 CD 67 58 2A 36 58 CD 67 58 C9 11
9060  38 58 D5 CD BF 19 E1 CD FB 24 CD 74 1F 2A 51 67
9070  CD E7 28 CD 1A 27 22 51 67 CD 79 1F CD 84 1F A5
9080  14 CD 55 19 2C 93 58 22 51 67 C9 F1 C9 11 00 5F
9090  CD 03 00 1A FE 1B 28 05 FE 0D 28 01 C9 3E 12 CD
90A0  12 00 18 E9 CD 08 58 ED 5B 00 90 CD 36 19 30 24
90B0  AF BE 23 20 F6 BE 20 F3 E5 23 11 00 5F 1A FE 0D
90C0  28 0A BE 28 03 E1 18 DF 13 23 18 F1 E1 BE 23 20
90D0  FC C3 40 58 21 00 00 18 F8 2A 00 90 23 36 00 23
90E0  36 00 23 EB 21 00 5F 3E 0D BE ED A0 20 FB 1B ED
90F0  53 00 90 C9 CD 08 58 ED 5B 00 90 CD 36 19 38 05
9100  21 00 00 18 04 5E 23 56 EB C3 40 58 CD 08 58 D5
9110  EB 2A 00 90 23 23 22 00 90 B7 ED 52 44 4D 2A 00
9120  90 54 5D 2B 2B ED B8 EB D1 73 23 72 C3 40 58 00
9130  00 00 2A 51 67 CD B6 3F 3E 01 32 96 3F CD 0B 1E
9140  22 51 67 11 00 90 2A 00 90 AF ED 52 23 22 AA 3F
9150  ED 53 AC 3F CD 26 3A CD 37 3E ED 43 B4 3F C5 CD
9160  4A 3A C1 21 00 90 CD 1C 3D CD B2 1C CD C9 37 CD
9170  D3 46 C9 00 00 CD 08 58 FD 21 7B 59 DD 21 00 00
9180  E5 21 00 90 23 23 3E 0D BE 20 FA 23 AF BE 20 05
9190  23 BE 28 F1 2B ED 5B 00 90 CD 36 19 30 16 D1 D5
91A0  7B BE 23 20 0C 7A BE 20 08 DD 23 AF FD BE 00 28
91B0  0A 23 18 D8 DD E5 E1 D1 C3 40 58 ED 5B 00 90 2B
91C0  E5 EB ED 52 44 4D 62 6B 23 23 ED B0 1B 1B ED 53
91D0  00 90 E1 E5 2B 3E 0D BE 20 0F 23 CD 36 19 3E 00
91E0  30 0A BE 20 04 23 BE 28 03 E1 18 C5 E1 E5 2B BE
91F0  2B 20 FC D1 E5 B7 2A 00 90 ED 52 44 4D 03 03 E1
9200  E5 EB ED B0 1B 1B ED 53 00 90 18 DD CD 1B 00 FE
9210  20 28 F9 CD 08 58 7E FE 0D CA 40 58 CD 12 00 23
9220  18 F4 FF FF FF 21 00 90 11 08 58 01 24 02 ED B0
9230  C9 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

```
0 REM **CITY BOMBER by D.P. Humphrey **
10 REM**        for MZ-80A            **
20 REM**  running on BASIC SA-5510.   **
30 REM
40 REM
50 POKE10407,0
60 PRINT""
70 PRINT"
80 PRINT"
90 PRINT"
100 PRINT"
110 PRINT"
120 PRINT"
130 PRINT""
140 PRINT""
150 PRINT"
160 PRINT"
170 PRINT"
180 PRINT"
190 PRINT"
200 PRINT"
210 FORT=1TO1000:NEXTT:PRINT"By David Humphrey. @ 1984."
220 PRINT""
230 FORT=1TO5000:NEXTT
240 FORT=1TO26:PRINTCHR$(4);:FORS=1TO10:NEXTS,T
250 A$="":BU=0:BO=0:A=0
260 H1=10:XE=0:SC=0:TEM=5:ER=00
270 PRINT"                    CITY BOMBER"
280 PRINT"                               "
290 PRINT"You are the pilot of a night bomber"
300 PRINT"who has to raze the city to the ground."
310 PRINT"To release your bomb, push any "
320 PRINT"key you find comfortable."
330 PRINT"If your 'plane hits a building, you    will explode";
340 PRINT", crash and die.If you level the city, ";
350 PRINT"you progress onto a larger one."
360 PRINT"        PUSH ANY KEY TO CONTINUE"
370 GETMK$:IFMK$=""THEN370
380 TEM=5:ER=00
390 PRINT"":A$="":BU=0:BO=0:A=0:XE=0:CR=0
400 PRINT"You can choose the sort of wind"
410 PRINT"you are to fly into. "
420 PRINT"1) NO WIND AT ALL."
430 PRINT"2) A LIGHT BREEZE."
440 PRINT"3) A STRONG BREEZE."
450 PRINT"Input your choice."
460 GETQ:IFQ=0THEN460
470 IFQ>3THEN44
480 IFQ=1THENWIN=40
490 IFQ=2THENWIN=39.925
500 IFQ=3THENWIN=39.7
510 H1=10
520 PRINT""
530 TEMPOTEM
540 T=54207
550 T=T+1:IFT=54248THEN580
560 POKET,208
570 GOTO550
580 FORT=54173TO54202
590 H=INT(RND(1)*H1)*40
600 FORI=TTOT-HSTEP-40
610 POKEI,74
```

```
620 NEXTI,T
630 A=A+1:Z=53252+A
640 POKE4514,SOU:USR(68)
650 GETB$:IFB$<>""THEN690
660 GOTO880
670 IFZ>54205THEN1150
690 GOTO630
690 DF=INT(RND(7)*H1)+1
700 POKEZ-5,0:FORB=Z+38TO54207STEPWIN
710 BO=1
720 ER=ER+8
730 POKE4514,4:POKE4513,ER:USR(68)
740 IFPEEK(Z+2)=74THEN960
750 IFPEEK(B)=74THENBU=BU+1:SC=SC+1:MUSIC"CO"
760 IFBU=DFTHENBU=0:B$="":ER=00:GOTO820
770 POKEB,83
780 A=A+1:Z=53252+A:GOTO880
790 POKEZ-6,0
800 POKEB,0
810 NEXTB:ER=00
820 BO=0:BU=0
830 IFSC>HSTHENHS=SC
840 PRINT"H→→SCORE - ";SC
850 PRINT"H→→→→→→→→→→→→→→→→→→→→→→→→→HI SCORE - ";HS
860 IFPEEK(B)<>208THEN POKEB,0
870 GOTO630
880 POKEZ-4,59:POKEZ-5,0:POKEZ-2,58:POKEZ-1,161:POKEZ,58:POKEZ+1,94
890 POKEZ-3,58:POKEZ-5,0
900 SOU=INT(100+(A/17))
910 POKE4514,SOU:USR(68)
920 IFBO=2THEN1020
930 IFPEEK(Z+2)=74THEN960
940 IFBO=1THEN800
950 GOTO670
960 C=Z:BO=2
970 Z=0
980 FORZ=CTO54207STEP40:CR=CR+10
990 USR(68)
1000 POKE4513,2:POKE4514,CR
1010 GOTO880
1020 POKEZ-4,0:POKEZ-5,0:POKEZ-2,0:POKEZ-1,0:POKEZ,0:POKEZ+1,0
1030 POKEZ-3,0:POKEZ-5,0
1040 NEXTZ
1050 USR(71)
1060 Z=Z-40
1070 POKEZ-4,228:POKEZ-3,224:POKEZ-2,109:POKEZ-1,160:POKEZ,203:POKEZ+1,22
1080 FOREX=1TO51:XE=XE+2
1090 USR(68):POKE4514,100+XE:PRINTCHR$(0);"H":NEXTEX:USR(71)
1100 IFEX/2=(INT(EX/2))THENPRINTCHR$(0)
1110 PRINT"H→→→→→→→→→→→→→→→→→↓↓↓↓↓↓ANOTHER GO ? "::USR(2483)
1120 GETR$:IFR$="Y"THENSC=0:GOTO380
1130 IFR$="N"THENPRINT"■":END
1140 GOTO1120
1150 MN=54205-4
1160 FORHL=MNTOMN-20STEP-1
1170 POKEHL,202
1180 MUSIC"RO"
1190 POKEHL,0
1200 NEXTHL
1210 MUSIC"RO"
1220 POKEHL,202
1230 MUSIC"RO"
```

```
1240 FORT=1TO03:MUSIC"R0"
1250 POKEHL,202:USR(62)
1260 POKEHL,203:USR(62):MUSIC"R1"
1270 POKEHL,205:USR(62):MUSIC"R1"
1280 POKEHL,204:USR(62):MUSIC"R1"
1290 POKEHL,202:USR(62):MUSIC"R1"
1300 NEXTT
1310 USR(71)
1320 TEM=TEM+1:IFTEM>7THENTEM=7
1330 PRINT"■■■■■■■■■■■■■■■■■■■■NEXT SCREEN COMING.":H1=H1+2
1340 FORT=1TO3
1350 POKEHL,202:MUSIC"+A0":POKEHL,0
1360 POKEHL-40,202:MUSIC"+B0":POKEHL-40,0
1370 POKEHL-80,202:MUSIC"+#B1":POKEHL-80,0
1380 POKEHL-40,202:MUSIC"+B0":POKEHL-40,0
1390 POKEHL,202:MUSIC"R2"
1400 NEXTT:POKEHL,0
1410 FORT=1TO3
1420 POKEHL,202:MUSIC"+A0":POKEHL,0
1430 POKEHL-40,202:MUSIC"+B0":POKEHL-40,0
1440 POKEHL-80,202:MUSIC"+#B1":POKEHL-80,0
1450 POKEHL-40,202:MUSIC"+B0":POKEHL-40,0
1460 POKEHL,202:MUSIC"R2"
1470 NEXTT:POKEHL,0
1480 FORHL=MN-20TOMN
1490 POKEHL,202
1500 MUSIC"R0"
1510 POKEHL,0
1520 NEXTHL
1530 POKEHL+1-40,202:MUSIC"R0":POKEHL+1-40,0
1540 POKEHL+1-80,202:MUSIC"R0":POKEHL+1-80,0
1550 POKEHL+2-80,202:MUSIC"R0":POKEHL+2-80,0
1560 POKEHL+3-40,202:MUSIC"R0":POKEHL+3-40,0
1570 A=0:BU=0:B0=0:GOTO520
1580 END
```

```
 1 REM*********************
 2 REM*                   *
 3 REM*  LABYRINTH        *
 4 REM*                   *
 5 REM*  Copyright by     *
 6 REM*                   *
 7 REM* Letizia Bizzarri  *
 8 REM*                   *
 9 GOSUB2000
 10 PRINT"■":NC=10:S=-1:M=19
 15 TEMPO7
 20 DIMN(255):N(0)=53248+500
 30 FORA=1TONC:N(A)=N(0)+A:NEXT
 40 GOSUB1070:M=M+1
 50 GETA$
 60 IFA$="Y"THENS=-40
 70 IFA$="N"THENS=40
 80 IFA$="G"THENS=-1
 90 IFA$="J"THENS=1
 100 N(0)=N(0)+S
 105 IFPEEK(N(0))=67THENPOKEN(0),0:MUSIC"▔C0▔D▔E▔F":GOTO600
 110 IFPEEK(N(0))<>0THEN700
```

```
200 POKEN(0),207
210 FORA=NCTO1STEP-1:N(A)=N(A-1):POKEN(A),68:NEXT
220 POKEN(NC),0:IFRND(1)>0.8THENGOSUB500
230 GOTO50
500 REM*Sceglie casualmente la posi-*
505 REM**zione della nuova casella***
506 X=INT(38*RND(1)):Y=INT(24*RND(1))
507 IFPEEK(53248+X+Y*40)<>0THEN506
508 POKE53248+X+Y*40,67:RETURN
510 X=INT(919*RND(1))+53288:IFPEEK(X)<>0THEN510
520 POKEX,67:RETURN
599 REM*Sceglie casualmente i punti*
600 J=INT(INT(4*RND(1))+1)*100
610 P=P+J:PRINT"■Points= ";P
620 GOTO200
700 REM*Fine gioco*
710 FORA=1TO5
720 POKEN(0),0:MUSIC"C3":POKEN(0),207
730 MUSIC"-B":NEXT
740 GOSUB2500
750 PRINT"■■■■Another go?   (Y/N)"
751 GETA$
752 IFA$="Y"THENGOTO10
753 IFA$="N"THEN755
754 GOTO751
755 PRINT"■■■OK.Bye from the snake!":END
999 REM*Disegna il labirinto*
1070 PRINT"■
1080 PRINT"■
1090 PRINT"■
1100 PRINT"■
1110 PRINT"■
1120 PRINT"■
1130 PRINT"■
1140 PRINT"■
1150 PRINT"■
1160 PRINT"■
1170 PRINT"■
1180 PRINT"■
1190 PRINT"■
1200 PRINT"■
1210 PRINT"■
1220 PRINT"■
1230 PRINT"■
1240 PRINT"■
1250 PRINT"■
1260 PRINT"■
1270 PRINT"■
1280 PRINT"■
1290 PRINT"■
1300 PRINT"■
1310 PRINT"■
1320 PRINT"■■■■■■":RETURN
2000 PRINT"■■■■■■■■■■
2010 PRINT"■■■■■
2020 PRINT"■■■■■
2030 PRINT"■■■■■          LABYRINTH
2040 PRINT"■■■■■
2050 PRINT"■■■■■
2060 PRINT"■■■■■
```

```
 2070 MUSIC"R7RR
 2080 PRINT"■"
 2090 PRINT"■■■■■Do you need instruction? Y/N"
 2100 GETA$:IFA$=""THEN2100
 2110 IFA$="N"THENRETURN
 2120 PRINT"■"
 2130 PRINT"■■■There is a little snake in the
 2140 PRINT"labyrinth. You move it with the
 2150 PRINT"following keys:
 2155 PRINT"■■■              (up)
 2160 PRINT"■                 Y
 2170 PRINT"■     (left) G + J (right)
 2180 PRINT"■                 N
 2190 PRINT"■              (down)
 2200 PRINT"■■■PRESS ANY KEY "
 2210 GETA$:IFA$=""THEN2210
 2220 PRINT"■"
 2230 PRINT"If you eat the ■ that appear you'll
 2240 PRINT"have a lot of point.
 2270 PRINT"■■■Be careful!!!
 2320 PRINT"■Any key to go
 2325 PRINT"-----------------------------------------"
 2330 GETA$:IFA$=""THEN2330
 2340 RETURN
 2500 INPUT"■■■■■What's your name? ";NOME$
 2505 IFP>1000 GOTO2515
 2510 BRAVO$=" you are not too expert.":GOTO2550
 2515 IFP>10000GOTO2525
 2520 BRAVO$=" you are learning something!":GOTO2550
 2525 IFP>50000GOTO2535
 2530 BRAVO$=" you have fantastic responses!":GOTO2550
 2535 IFP>50000THEN2540
 2540 BRAVO$=" you are the champion I was looking for!":GOTO2550
 2550 PRINT"■■■■■OK, ";NOME$;"..."
 2560 PRINT:PRINTBRAVO$
 2570 P=0:RETURN


3 GOSUB2000
4 INPUT"■How good you are?      (1-5) ";L
5 IF(L<1)+(L>5)THENPRINT"■                                              ■■■":GOTO4
6 PRINT"■":P=15:L=5-L
10 FOR N=0TO21
20 CURSOR0,N:PRINT"■";:CURSOR31,N:PRINT"■";:NEXT
30 CURSOR0,0:PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"
40 CURSOR0,21:PRINT"■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■"
50 FORN=0TO41:READ A,B,C:FORM=ATOB
60 CURSORM,C:PRINT"■";
65 POKE4514,M:USR(68):NEXTM,N
70 FORN=0TO27:READ A,B:CURSORB,A:PRINT"■";:POKE4514,N:USR(68):NEXT:USR(71)
80 CURSOR5,4:PRINT" ";:CURSOR1,1:PRINT"S";:CURSOR1,2:PRINT"T";
90 CURSOR1,3:PRINT"A";:CURSOR1,4:PRINT"R";:CURSOR1,5:PRINT"T";
100 CURSOR30,15:PRINT"F■■I■■■■N■■■■I■■■■S■■■1"
120 E$=">"
130 X=3:Y=3:A=0:B=0
140 GETA$
150 IFA$=""THENGOTO200
160 IFA$="G"THENB=-1:A=0:E$="<"
170 IFA$="J"THENB=1:A=0:E$=">"
180 IFA$="N"THENB=0:A=1
190 IFA$="Y"THENB=0:A=-1
200 CURSOR Y,X:PRINT" "
```

```
205 X=X+A:Y=Y+B
210 C$=CHARACTER$(Y,X)
220 IFC$="♠"THEN300
230 IFC$="♣"THEN400
240 IFC$="▗"THEN900
250 POKE4514,X+Y:USR(68):CURSORY,X:PRINTE$
260 FORK=1TOL*10:NEXT:GOTO140
300 FORN=50TO-50STEP5:POKE4514,N:USR(68):NEXTN
310 CURSORY,X:PRINT"♠":CURSOR8,21:PRINT"You crashed"
320 FORN=45TO0STEP-2:POKE4514,N:USR(68):NEXT
330 CURSOR8,21:PRINT"You crashed
340 FORN=100TO0STEP-4:POKE4514,N:USR(68):NEXTN
350 P=P-1:IFP=0THENGOTO500
360 GOTO130
400 FORN=0TO50STEP2:POKE4514,60-N:USR(68):NEXTN
410 CURSORY,X:PRINT"♣":CURSOR8,21:PRINT"You crashed
420 FORN=0TO45STEP1.75:POKE4514,N:NEXTN:CURSOR8,21:PRINT"You crashed
430 FORN=0TO100STEP4:POKE4514,N:USR(68):NEXT N
440 P=P-1:IFP=0THEN500
450 GOTO130
500 FORN=0TO21:PRINTCHR$(0):MUSIC"R0"
505 NEXT:PRINT"▣"
506 FORA=0TO255:POKE4514,A:USR(68):NEXT:USR(71)
510 CURSOR8,7:PRINT"▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀"
520 CURSOR8,13:PRINT"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄"
530 FORN=7TO13:CURSOR8,N:PRINT"▌";:CURSOR31,N:PRINT"▐":NEXT N
540 CURSOR10,9:PRINT"Pluck up courage...":CURSOR15,11:PRINT"try again!!!"
545 PRINT"▚▞▚▞▚▞"
550 PRINT"Another go (Y/N)"
560 GETR$:IFR$=""THEN560
561 IFR$="Y"THENRUN4
570 END
900 FORN=0TO100:POKE4514,N:USR(68):NEXTN
910 PRINT"▣":USR(71):FORN=7TO13
920 CURSOR8,N:PRINT"▌";:CURSOR24,N:PRINT"▐";:NEX.
930 CURSOR8,7:PRINT"▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀▀";:CURSOR8,13:PRINT"▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄▄
940 CURSOR10,9:PRINT"    We have";:CURSOR10,11:PRINT"a winner!!!!!!"
950 INPUT"▚▞Another go (Y/N)";T$:IFT$="Y"THENRUN4
960 END
1000 DATA5,30,1,6,8,2,13,30,2,25,30,3,26,30,4,27,30,5,1,3,6,9,10
1010 DATA6,22,22,6,29,30,6,1,23,7,30,30,7,1,7,8,12,23,8,30,30,8,1,6,9,17
1020 DATA22,9,1,4,9,30,30,9,1,1,10,30,30,10,1,1,11,29,30,11,1,1,12,10,11
1030 DATA 12,29,30,12,1,1,13,9,14,13,22,30,13,1,1,14,5,15,14,21,30,14,1
1040 DATA 1,14,1,1,15,5,30,15,1,1,16,6,30,16,1,1,17,1,1,18,19,24,18,1,6,1
1050 DATA1,30,20
1500 DATA 3,8,3,5,5,5,5,8,2,11,4,11,4,17,6,17,3,22,5,22,7,24,7,28
1510 DATA9,28,9,25,12,23,10,23,11,19,13,19,10,15,12,15,8,10,10,10,10,7
1520 DATA12,7,13,2,13,4,16,4,16,2,17,9,19,9,19,12
2000 PRINT"▣"
2010 DIMT$(13)
2020 T$(1)="▀▀▀▀  ▀  ▀  ▀▀▀▀  ▀▀▀▀  ▀▀▀▀"
2030 T$(2)="▀     ▀  ▀  ▀  ▀  ▀  ▀  ▀  ▀"
2040 T$(3)="▀▀▀   ▀  ▀  ▀  ▀  ▀  ▀  ▀  ▀"
2050 T$(4)="▀▀▀▀  ▀▀▀▀  ▀▀▀▀  ▀▀▀▀  ▀▀▀▀"
2060 T$(5)="   ▀  ▀  ▀  ▀  ▀  ▀  ▀  ▀  "
2080 T$(6)="▀▀▀▀  ▀  ▀  ▀  ▀  ▀  ▀  ▀  "
2090 REM
```

```
2100 T$(7)="
2110 T$(8)="
2120 T$(9)="
2130 T$(10)="
2140 T$(11)="
2150 T$(12)="
2160 PRINT"":FORX=1TO6:PRINTTAB(5);T$(X):NEXT
2165 PRINT:PRINT
2170 FORX=7TO12:PRINTTAB(5);T$(X):NEXT
2180 PRINT"":PRINTTAB(2);"
2190 PRINT"":FORX=1TO18:PRINTTAB(2);"|  ":NEXT
2200 PRINT"":FORX=1TO18:PRINTTAB(34);"  |":NEXT
2210 PRINT"
2220 PRINTTAB(2);"
2225 PRINT""
2230 FORX=1TO16
2233 TEMPO7
2235 FORS=1TO29:PRINTTAB(4);CHR$(45);:NEXT:PRINT:PRINT""
2240 MUSIC"+A0A":NEXT
2250 A$="Copyright by LETIZIA BIZZARRI"
2260 B$="     VIA LAGO ISOLFTTA,31      "
2270 C$="     65100 PESCARA - ITALY     "
2280 D$="        tel.085/60672          "
2290 PRINT""
2300 PRINTTAB(4);A$:PRINT:PRINTTAB(4);B$:PRINT
2310 PRINTTAB(4);C$:PRINT:PRINT:PRINTTAB(4);D$
2320 TEMPO1: MUSIC"R7R
2330 PRINT"Do you need instruction (Y/N)?"
2340 GETR$:IFR$=""THEN2340
2350 IFR$="N"THENRETURN
2360 PRINT""
2370 PRINT"Are you ready for slalom?
2380 PRINT"You don't have to crash against the
2390 PRINT"trees.
2400 PRINT"Keys to move are:
2410 PRINT" G - left      * right  - J"
2420 PRINT" Y - up        * down   - N"
2430 PRINT"If you'll crash, you'll start again.
2440 PRINT"You have 15 attempts "
2450 PRINT"Go? (Y/N)"
2460 GETR$:IFR$=""THEN2460
2470 IFR$="Y"THENRETURN
2480 END


   1 GOTO 3000
   2 USR(62):CLR
   5 PU=0:TI$="000000": PRINT""
  10 A$="
  20 B$="                                    "
  30 C$="                                    "
  40 N$=MID$(L$,1,35)
  41 CURSOR0,22:PRINTC$:CURSOR0,21:PRINTB$:CURSOR0,20:PRINTA$
  45 CURSOR0,23:PRINT"                Base alpha                "
  50 CLR:A$="                                                  +":B$="*  =  =="
  55 S=53248:P1=4514
  60 X1=0:Y1=19:CURSORX1,Y1:PRINTA$
  70 PRINT"":DIMX(5),Y(5)
  80 FORA=1TO5
  90 X=INT(20*RND(1))+19
 100 Y=INT(10*RND(1))
```

```
110 X(A)=X:Y(A)=Y:CURSORX(A),Y(A):PRINT"$":NEXT
120 GOSUB150
130 GOSUB500
140 GOTO120
150 GETR$:IFR$=""THENRETURN
160 IFR$="Y"THENS=-1:IFY1+S<0THENS=0
170 IFR$="N"THENS=1:IFY1+S>19THENS=0
180 IFR$="H"THENCURSOR3,Y1:PRINTB$:MUSIC"+A0":GOSUB1000:RETURN
190 CURSOR0,Y1:PRINT"    ":CURSOR38,Y1:PRINT" ":Y1=Y1+S
195 IFY1<=0THENY1=Y1+1
200 CURSOR0,Y1:PRINTA$:RETURN
500 REM
505 K=K+1:IFK>5THENK=0:GOTO505
510 A=INT(3*RND(1))
520 IFY(K)+A>19THENCURSORX(K)-1,Y(K)+A:PRINT"\|/■■■■-*-":GOTO2000
530 CURSORX(K),Y(K):PRINT" "
540 Y(K)=Y(K)+A:CURSORX(K),Y(K):PRINT"$"
550 RETURN
1000 E=0:FORN=1TO5:IFY1=Y(N)THENCURSORX(N),Y(N):N1=N:N=5:E=1
1010 NEXT:IFE=1THEN1016
1015 CURSOR3,Y1:PRINT"       ":RETURN
1016 FORX=3TOX(N1)-1:CURSORX,Y1:PRINT" =";:POKEP1,X*2:USR(68):NEXT
1017 CURSORX(N1),Y(N1):PRINT"*":MUSIC"-BC":CURSORX(N1),Y(N1):PRINT" "
1018 PU=PU+1:PRINT"■Destroyed: ";PU:PRINT"■■Time: ";TI$
1019 IFPU=50THEN7000
1020 X(N1)=INT(20*RND(1))+19:Y(N1)=0
1030 MUSIC"+A0"
1040 CURSOR3,Y1:PRINT"       ":RETURN
2000 FORA=0TO255:POKEP1,A:PRINT"■";CHR$(0)
2010 USR(68):NEXT:USR(71)
2020 GOTO6500
3000 PRINT"■":CLR
3010 M$="#":O1$="=":FORX=1TO33:O$=O$+O1$:NEXT:O$=M$+O$+M$
3020 V1$="||":FORX=1TO15:V$=V$+V1$+"■■":NEXT
3030 CURSOR2,5:PRINTO$
3040 CURSOR2,6:PRINTV$
3060 CURSOR2,21:PRINTO$
3065 CURSOR36,6:PRINTV$
3070 PRINT"■"
3080 DIMTT$(5)
3090 TT$(1)="■■■ ■■■ ■■■ ■■■ ■ ■ ■■■ ■■■ ■■■"
3100 TT$(2)="■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■"
3110 TT$(3)="■■■ ■■■ ■■■ ■ ■■■ ■ ■ ■■■ ■■■"
3120 TT$(4)="■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■ ■"
3130 TT$(5)="■■■ ■■■ ■ ■■■ ■ ■ ■■■ ■ ■"
3140 FORZ=1TO5:CURSOR4,10+Z:PRINTTT$(Z):NEXT
3145 TEMPO7
3150 FORZ=1TO21:PRINT"■";:MUSIC"-A0A+A":PRINTCHR$(0);:NEXT
3160 CLR
3170 A$="Copyright by "
3180 B$=" LETIZIA BIZZARRI - via lago isoletta,31 - 65100 PESCARA "
3185 C$=" Italy ":S$="                "
4000 L$=" "+S$+A$+B$+C$+" "
4010 A=1:B=31
4020 X=4
4030 N$=MID$(L$,A,B)
4050 CURSORX,9:PRINTN$
4060 MUSIC"+A2+B"
4070 A=A+1
4075 IFA=LEN(L$)THENPRINTCHR$(0):GOTO5000
4080 GOTO4020
5000 PRINT"■"
```

```
5010 PRINT"Do you need instruction? (Y/N)
5020 GETR$:IFR$=""THEN5020
5030 IFR$="N"THEN2
5040 USR(62):PRINT"You have to defend
5050 PRINT"BASE ALPHA, the last emplacement
5060 PRINT"for EARTH!!!
5070 PRINT"Be very carefull!!!"
5080 PRINT"KEYS TO MOVE: 'Y' to go up /
5090 PRINT"'N' to go down  / 'H' to shoot
5100 PRINT"The sight '+' will help you to see
5110 PRINT"when your cannon is in position.
5120 PRINT
5130 PRINT"Aliens are 50.
5140 PRINT"ANY KEY TO GO
5160 GETR$:IFR$=""THEN5160
5170 GOTO2
6000 USR(62):PRINT"YOU DID IT!!!
6010 PRINT"YOU ARE GREAT!
6020 GOTO6510
6030 PRINT"Another go? (Y/N)
6040 GETR$:IFR$=""THEN6040
6045 IFR$="Y"THEN2
6050 USR(62):PRINT"SEE YOU AGAING!
6060 PRINT"BYE!!!!":END
6500 USR(62):PRINT"Aliens invaded earth!!!
6510 PRINT"You destroyed ";PU;" aliens
6520 PRINT"in ";MID$(TI$,3,2);" minuts and ";RIGHT$(TI$,2);" seconds"
6525 M=VAL(MID$(TI$,3,2)):S=VAL(RIGHT$(TI$,2)):MS=M+S:PP=INT((MS*PU)/25
6526 PRINT"Your score is: ";PP
6530 PRINT"Another go? (Y/N)
6540 GETR$:IFR$=""THEN6540
6550 IFR$="Y"THEN2
6560 IFR$="N"THEN6050
6570 END
7000 PRINT""
7010 FORX=1TO20:PRINTCHR$(0):MUSIC"-A0A+A":NEXT
7020 GOTO6000


5 PRINT"";TAB(10);"Digital Clock"
6 PRINT"1 Insert time
7 PRINT"2 Insert alarm
8 PRINT"3 to see digital clock
9 INPUT"Comand ";N
10 ON N GOTO15,25,35,45
11 GOTO9
15 INPUT"Insert time (HHMMSS) ";W$
16 L=LEN(W$):IFL=6THEN19
17 GOTO15
19 TI$=W$:GOTO5
25 INPUT"Insert time for alarm (HHMMSS)";S$
26 IFLEN(S$)<>6THEN25
28 GOTO5
35 GOTO160
160 PRINT""
170 PRINTTAB(26);"":PRINTTAB(12);""
180 F$=MID$(TI$,1,1)
190 E$=MID$(TI$,2,1)
200 D$=MID$(TI$,3,1)
210 C$=MID$(TI$,4,1)
220 B$=MID$(TI$,5,1)
230 A$=RIGHT$(TI$,1)
```

```
235 IFVAL(TI$)=VAL(S$)THENMUSIC"CODEFGABCDEFGABCDEFGBCDEFGABCDEFGAB
240 X=34:IFA$="0"THENGOSUB860:GOTO340
250 IFA$="1"THENGOSUB940:GOTO340
260 IFA$="2"THENGOSUB1020:GOTO340
270 IFA$="3"THENGOSUB1100:GOTO340
280 IFA$="4"THENGOSUB1180:GOTO340
290 IFA$="5"THENGOSUB1260:GOTO340
300 IFA$="6"THENGOSUB1340:GOTO340
310 IFA$="7"THENGOSUB1420:GOTO340
320 IFA$="8"THENGOSUB1500:GOTO340
330 IFA$="9"THENGOSUB1580
340 IFA$=RIGHT$(TI$,1)THEN340
350 X=28:IFB$="0"THENGOSUB860:GOTO450
360 IFB$="1"THENGOSUB940:GOTO450
370 IFB$="2"THENGOSUB1020:GOTO450
380 IFB$="3"THENGOSUB1100:GOTO450
390 IFB$="4"THENGOSUB1180:GOTO450
400 IFB$="5"THENGOSUB1260:GOTO450
410 IFB$="6"THENGOSUB1340:GOTO450
420 IFB$="7"THENGOSUB1420:GOTO450
430 IFB$="8"THENGOSUB1500:GOTO450
440 IFB$="9"THENGOSUB1580:GOTO450
450 X=20:IFC$="0"THENGOSUB860:GOTO550
460 IFC$="1"THENGOSUB940:GOTO550
470 IFC$="2"THENGOSUB1020:GOTO550
480 IFC$="3"THENGOSUB1100:GOTO550
490 IFC$="4"THENGOSUB1180:GOTO550
500 IFC$="5"THENGOSUB1260:GOTO550
510 IFC$="6"THENGOSUB1340:GOTO550
520 IFC$="7"THENGOSUB1420:GOTO550
530 IFC$="8"THENGOSUB1500:GOTO550
540 IFC$="9"THENGOSUB1580:GOTO550
550 X=14:IFD$="0"THENGOSUB860:GOTO650
560 IFD$="1"THENGOSUB940:GOTO650
570 IFD$="2"THENGOSUB1020:GOTO650
580 IFD$="3"THENGOSUB1100:GOTO650
590 IFD$="4"THENGOSUB1180:GOTO650
600 IFD$="5"THENGOSUB1260:GOTO650
610 IFD$="6"THENGOSUB1340:GOTO650
620 IFD$="7"THENGOSUB1420:GOTO650
630 IFD$="8"THENGOSUB1500:GOTO650
640 IFD$="9"THENGOSUB1580:GOTO650
650 X=6:IFE$="0"THENGOSUB860:GOTO750
660 IFE$="1"THENGOSUB940:GOTO750
670 IFE$="2"THENGOSUB1020:GOTO750
680 IFE$="3"THENGOSUB1100:GOTO750
690 IFE$="4"THENGOSUB1180:GOTO750
700 IFE$="5"THENGOSUB1260:GOTO750
710 IFE$="6"THENGOSUB1340:GOTO750
720 IFE$="7"THENGOSUB1420:GOTO750
730 IFE$="8"THENGOSUB1500:GOTO750
740 IFE$="9"THENGOSUB1580:GOTO750
750 X=0:IFF$="0"THENGOSUB860:GOTO850
760 IFF$="1"THENGOSUB940:GOTO850
770 IFF$="2"THENGOSUB1020:GOTO850
780 IFF$="3"THENGOSUB1100:GOTO850
790 IFF$="4"THENGOSUB1180:GOTO850
800 IFF$="5"THENGOSUB1260:GOTO850
810 IFF$="6"THENGOSUB1340:GOTO850
820 IFF$="7"THENGOSUB1420:GOTO850
830 IFF$="8"THENGOSUB1500:GOTO850
840 IFF$="9"THENGOSUB1580:GOTO850
```

```
850 GOTO180
860 PRINT"■":PRINTTAB(X);"■■■■"
870 PRINTTAB(X);"■    ■"
880 PRINTTAB(X);"■    ■"
890 PRINTTAB(X);"■    ■"
900 PRINTTAB(X);"■    ■"
910 PRINTTAB(X);"■    ■"
920 PRINTTAB(X);"■■■■"
930 RETURN
940 PRINT"■":PRINTTAB(X);" ■■  "
950 PRINTTAB(X);"■■   "
960 PRINTTAB(X);"  ■   "
970 PRINTTAB(X);"  ■   "
980 PRINTTAB(X);"  ■   "
990 PRINTTAB(X);"  ■   "
1000 PRINTTAB(X);" ■■■  "
1010 RETURN
1020 PRINT"■":PRINTTAB(X);"■■■■■"
1030 PRINTTAB(X);"     ■"
1040 PRINTTAB(X);"     ■"
1050 PRINTTAB(X);"■■■■■"
1060 PRINTTAB(X);"■     "
1070 PRINTTAB(X);"■     "
1080 PRINTTAB(X);"■■■■■"
1090 RETURN
1100 PRINT"■":PRINTTAB(X);"■■■■■"
1110 PRINTTAB(X);"     ■"
1120 PRINTTAB(X);"     ■"
1130 PRINTTAB(X);"  ■■■■"
1140 PRINTTAB(X);"     ■"
1150 PRINTTAB(X);"     ■"
1160 PRINTTAB(X);"■■■■■"
1170 RETURN
1180 PRINT"■":PRINTTAB(X);"■    ■"
1190 PRINTTAB(X);"■    ■"
1200 PRINTTAB(X);"■    ■"
1210 PRINTTAB(X);"■■■■■"
1220 PRINTTAB(X);"     ■"
1230 PRINTTAB(X);"     ■"
1240 PRINTTAB(X);"     ■"
1250 RETURN
1260 PRINT"■":PRINTTAB(X);"■■■■■"
1270 PRINTTAB(X);"■     "
1280 PRINTTAB(X);"■     "
1290 PRINTTAB(X);"■■■■■"
1300 PRINTTAB(X);"     ■"
1310 PRINTTAB(X);"     ■"
1320 PRINTTAB(X);"■■■■■"
1330 RETURN
1340 PRINT"■":PRINTTAB(X);"■■■■■"
1350 PRINTTAB(X);"■     "
1360 PRINTTAB(X);"■     "
1370 PRINTTAB(X);"■■■■■"
1380 PRINTTAB(X);"■    ■"
1390 PRINTTAB(X);"■    ■"
1400 PRINTTAB(X);"■■■■■"
1410 RETURN
1420 PRINT"■":PRINTTAB(X);"■■■■■"
1430 PRINTTAB(X);"     ■"
1440 PRINTTAB(X);"     ■"
1450 PRINTTAB(X);"     ■"
1460 PRINTTAB(X);"     ■"
1470 PRINTTAB(X);"     ■"
1480 PRINTTAB(X);"     ■"
1490 RETURN
```

```
1500 PRINT"■":PRINTTAB(X);"■■■■■■"
1510 PRINTTAB(X);"■    ■"
1520 PRINTTAB(X);"■    ■"
1530 PRINTTAB(X);"■■■■■■"
1540 PRINTTAB(X);"■    ■"
1550 PRINTTAB(X);"■    ■"
1560 PRINTTAB(X);"■■■■■■"
1570 RETURN
1580 PRINT"■":PRINTTAB(X);"■■■■■■"
1590 PRINTTAB(X);"■    ■"
1600 PRINTTAB(X);"■    ■"
1610 PRINTTAB(X);"■■■■■■"
1620 PRINTTAB(X);"     ■"
1630 PRINTTAB(X);"     ■"
1640 PRINTTAB(X);"■■■■■■"
1650 RETURN
```

```
4 PRINT"■"
5 PRINT TAB(10);"  AMAZING"
6 PRINT TAB(10);"  ■■■■■■■"
7 PRINT "The idea is to get to the square"
8 PRINT"pointed to by the arrow [←]."
9 PRINT"On the way you will face various tasks."
10 PRINT"If you don't succeed you'll start again."
11 PRINT"move using W=WEST,N=NORTH,S=SOUTH,E=EAST":PRINT" GOOD LUCK!"
12 PRINT"MAKE SURE SOUND IS TURNED ON"
14 PRINT"PRESS A KEY WHEN READY"
15 GET A$:IF A$="" THEN 15
90 B=5344B
100 PRINT"■"
110 REM DEATH MAZE
120 PRINTTAB(5);"  AMAZING    "
130 PRINTTAB(5);"  ■■■■■■■    "
140 CURSOR 0,4
150 PRINT"  ■■■■■■ ■■■■■■■■  "
160 PRINT"■■■  ■ ■ ■  ■   "
170 PRINT"  ■■■■ ■■■  ■■■■■■ "
180 PRINT"  ■  ■ ■  ■■■■  ■ "
190 PRINT"  ■■■■■■  ■ ■  ■■■■ "
200 PRINT"  ■  ■ ■  ■ ■ ■  ■"
210 PRINT"  ■■■■ ■■■■ ■■■■■■ "
220 PRINT"     ■■■ ■  ■  ■  ■ "
230 PRINT"       ■■■  ■■■■■■ "
231 PRINT"         ■  ■  ■  ■ "
232 PRINT"         ■  ■■■■■  ■ "
233 PRINT"        ■■ ■ ■ ■   ■ "
234 PRINT"        ■ ■■ ■■■  ■■ "
235 PRINT"        ■■■■  ■   ■  "
236 PRINT"              ■■■■■■■←"
237 MUSIC"AA"
240 POKE B,(206)
260 CURSOR 1,20
270 INPUT"W,N,S,E?";A$
280 A=ASC(A$)
282 IF A=69 THEN A=1:GOTO 290
284 IF A=78 THEN A=2:GOTO 290
286 IF A=83 THEN A=3:GOTO 290
287 IF A=87 THEN A=4:GOTO 290
288 GOTO 240
290 ON A GOTO 300,330,400,450
300 IF PEEK(B+1)<>208 THEN 240
```

```
310 B=B+1
320 GOTO 1000
330 IF PEEK(B-40)<>208 THEN 240
340 B=B-40
350 GOTO 1000
400 IF PEEK(B+40)<>208 THEN 240
410 B=B+40
420 GOTO 1000
450 IF PEEK(B-1)<>208 THEN 240
460 B=B-1
470 GOTO 1000
1000 C=INT(12*RND(1))+1
1010 IF B=53988 THEN 20000
1020 ON C GOTO 1000,2000,3000,4000,5000,6000,7000,8000,9000,10000,11000,12000
2000 X=53980:PRINT"■"
2005 PRINT"JUST SIT & HOPE THEY MISS YOU"
2006 FOR I=1 TO 900:NEXT
2007 PRINT"■"
2010 POKE X,(206)
2020 K=53248
2030 POKE K+INT(5*RND(1))+1,(207)
2040 POKE K+INT(10*RND(1))+1,(207)
2050 POKE K+INT(15*RND(1))+1,(207)
2060 POKE K+INT(20*RND(1))+1,(207)
2070 POKE K+INT(25*RND(1))+1,(207)
2080 POKE K+INT(30*RND(1))+1,(207)
2090 K=K+40
2100 IF K<54100 THEN MUSIC"A":GOTO 2030
2110 IF PEEK(X)<>206 THEN 2200
2120 PRINT"■";"JAMMY!"
2125 MUSIC"AA+AA+AA+AA+A"
2130 FOR X=1 TO 1500:NEXT
2140 GOTO 100
2200 PRINT"■";"SPLICED!"
2210 FOR X=1 TO 1500:NEXT
2220 GOTO 90
2225 MUSIC"D-D-D-D-D-D-D"
3000 REM DITTO DODGE:I1=54247:PRINT"■"
3002 PRINT"TRY TO GET TO THE TOP LEFT CORNER"
3003 PRINT"USE 4-LEFT"
3004 PRINT" '  8-UP"
3005 PRINT" '  6-RIGHT"
3006 PRINT" '  2-DOWN"
3007 PRINT"EVERY 6 GOES THEY MULTIPLY"
3008 PRINT"AND YOU MUST'NT HIT ONE"
3009 PRINT"YOUR MAN IS IN THE BOTTOM RIGHT CORNER"
3010 PRINT"nb.YOU MAY GO OFF THE SCREEN"
3011 PRINT"PRESS A KEY WHEN READY"
3012 GET A$:IF A$="" THEN 3012
3013 PRINT"■"
3015 POKE 53248,(208)
3020 I2=0
3030 FOR X=1 TO 12
3040 I=INT(1000*RND(1))+53249
3045 MUSIC"#F"
3050 IF PEEKI<>98 THEN POKE I,98
3060 NEXT X
3070 GET A$
3080 IF A$<>"" THEN POKE I1,(0):I2=I2+1:MUSIC"#G"
3090 IF A$="4" THEN I1=I1-1
3100 IF A$="8" THEN I1=I1-40
3110 IF A$="6" THEN I1=I1+1
3120 IF A$="2" THEN I1=I1+40
3130 IF PEEK(I1)=98 THEN 3300
```

```
3135 POKE I1,(202)
3140 IF I1=53248 THEN 3400
3150 IF I2=6 THEN 3020
3160 GOTO 3070
3300 PRINT"■"
3310 MUSIC"D-D-D-D-D-D"
3320 PRINT"DIDN'T MAKE IT!"
3330 FOR X=1 TO 1500:NEXT
3340 GOTO 90
3400 PRINT"■"
3410 PRINT"MADE IT!"
3415 MUSIC"AA+AA+AA+AA"
3420 FOR X=1 TO 1500:NEXT
3430 GOTO 100
4000 REM ADDITION
4001 PRINT"■"
4002 PRINT"YOU WILL SEE AN ADDITION PROBLEM"
4003 PRINT"WITH 5 SECS. TO ANSWER"
4004 PRINT"PRESS WHEN READY"
4005 GET A$:IF A$="" THEN 4005
4006 PRINT"■"
4010 A1=INT(10*RND(1))
4020 A2=INT(10*RND(1))
4030 A3=INT(10*RND(1))
4070 PRINT"YOU HAVE 5 SECONDS"
4080 PRINTA1;"+";A2;"+";A3
4082 T=VAL(TI$)
4084 INPUTA
4086 T1=VAL(TI$)
4090 IF A<>A1+A2+A3 THEN 4200
4092 IF T1-5>T THEN 4300
4094 T2=5-(T1-T)
4096 PRINT"YOU HAD ";T2;" SECONDS LEFT"
4100 PRINT"CORRECT"
4110 MUSIC"A-AA-AA-A"
4120 FOR X=1 TO1000:NEXT
4130 PRINT"■"
4140 GOTO 100
4200 MUSIC"D-D-D-D-D"
4210 PRINT"■"
4220 PRINT"WRONG,BACK TO START"
4230 FOR X=1 TO 1500:NEXT
4240 GOTO 90
4300 PRINT"■"
4310 MUSIC"D-D-D-D-D"
4320 PRINT"TOO LONG,BACK TO START"
4330 FOR X=1 TO 1500:NEXT
4340 GOTO90
5000 GOTO 100
6000 GOTO 100
7000 PRINT"■":S=0:SC=0
7001 PRINT" TRY TO DROP YOUR FACE IN THE GAP A FEW"
7002 PRINT" TIMES BY PRESSING A KEY WHEN ABOVE IT."
7003 PRINT
7004 PRINT" YOU HAVE A LIMITED NO. OF GOES."
7005 PRINT"PRESS A KEY WHEN READY"
7006 GET A$:IF A$="" THEN 7006
7007 PRINT"■"
7020 FOR X=54128 TO 54167
7030 POKE X,(208)
7040 NEXT X
7050 FOR X=54128 TO 54167
7060 POKE X,(0):POKE X+1,(0):POKE X+2,(0)
```

```
7062 IF X>54126 THEN POKE X,(208)
7064 POKE 53268,(207)
7066 GET A$
7068 IF A$<>"" THEN 7090
7080 NEXT X:GOTO 7050
7090 S=S+1
7100 IF S>7 THEN 7190
7130 IF X+1=54148 THEN SC=SC+1:W=X+1:GOTO 7170
7140 IF X+2=54148 THEN SC=SC+1:W=X+2:GOTO 7170
7160 MUSIC"D":GOTO 7030
7170 POKE 53268,(0):POKE W,(207):MUSIC"A"
7180 IF SC<3 THEN 7020
7182 PRINT"WELL DONE"
7184 FOR X=1 TO 1500:NEXT
7186 GOTO 100
7190 PRINT"TOO MANY GOES,START AGAIN"
7192 FOR X=1 TO 1500:NEXT
7200 GOTO 90
8000 GOTO 100
9000 Q=INT(3*RND(1))+1
9010 PRINT"■"
9020 PRINT"▒▒▒▒▒   ▒▒▒▒▒   ▒▒▒▒▒"
9030 PRINT"▒▒▒▒▒   ▒▒▒▒▒   ▒▒▒▒▒"
9040 PRINT"▒▒▒▒▒   ▒▒▒▒▒   ▒▒▒▒▒"
9050 PRINT"▒▒1▒▒   ▒▒2▒▒   ▒▒3▒▒"
9060 PRINT"▒▒▒▒▒   ▒▒▒▒▒   ▒▒▒▒▒"
9070 PRINT"▒▒▒▒▒   ▒▒▒▒▒   ▒▒▒▒▒"
9080 PRINT"BEHIND ONE IS A LION"
9090 PRINT"DON'T PICK THAT ONE"
9095 CURSOR 0,10
9100 INPUT"WHICH?";J
9110 IF Q=J THEN 9200
9115 IF J>3 THEN 9095
9120 PRINT"■"
9130 PRINT"NEARLY HAD YOU FOR SUPPER!"
9135 MUSIC"BA+BA+BA+ABBA+AGAA"
9140 FOR X=1 TO 1500:NEXT
9150 GOTO 100
9200 ON Q GOTO 9300,9400,9500
9300 CURSOR 0,1
9310 PRINT" *** "
9320 PRINT"◥■■◤"
9330 PRINT" ◤◥ "
9340 PRINT" ▀▀ "
9350 PRINT"     "
9360 PRINT"     "
9365 PRINT:PRINT:PRINT:PRINT"HE GOT YOU!"
9370 GOTO 9570
9400 CURSOR 0,1
9410 PRINT TAB(8);" *** "
9420 PRINT TAB(8);"◥■■◤"
9430 PRINT TAB(8);" ◤◥ "
9440 PRINT TAB(8);" ▀▀ "
9450 PRINT TAB(8);"     "
9460 PRINT TAB(8);"     "
9465 PRINT:PRINT:PRINT:PRINT"HE GOT YOU!"
9470 GOTO 9570
9500 CURSOR 0,1
9510 PRINT TAB(16);" *** "
9520 PRINT TAB(16);"◥■■◤"
9530 PRINT TAB(16);" ◤◥ "
9540 PRINT TAB(16);" ▀▀ "
9550 PRINT TAB(16);"     "
9560 PRINT TAB(16);"     "
```

```
9565 PRINT:PRINT:PRINT:PRINT"HE GOT YOU!"
9570 MUSIC"D-G-D-G-D-G-D-D"
9580 FOR X=1 TO 1500:NEXT
9590 PRINT"■"
9600 PRINT"START AGAIN"
9610 GOTO 90
10000 GOTO 100
11000 PRINT"■"
11010 PRINT"    ▌                    ▲"
11020 PRINT"    ▌                    ▌"
11030 PRINT"▄▄▄▌                   **▌"
11040 PRINT"\oo▒        Twee       *oo*"
11050 PRINT"▒◇◇▒          -Twee    ▀▄▀"
11060 PRINT"▒◇◇▒           *       ▄▀▄"
11070 PRINT"▀▀▀           * *      ▒▒ "
11080 PRINT" ▌▌                    ▄  "
11090 PRINT" ▌▌                    ▌▌ "
11100 PRINT"▀  ▀                   ▀  ▀"
11110 PRINT"Soldier       / \      Native"
11120 PRINT"                       "
11125 PRINT"                Bird"
11130 INPUT "WHO KILLED THE BIRD(Sol.-Nat.)?";K$
11140 K=INT(20*RND(1))+1
11150 IF K<10 THEN K1$="S":GOTO 11180
11160 K1$="N"
11180 IF LEFT$(K$,1)=K1$ THEN 11200
11190 MUSIC"D-D-D-D-D-D"
11195 PRINT"■";"WRONG":FOR X=1 TO 800:NEXT:GOTO 90
11200 MUSIC"AA+AA+AA+AA"
11210 PRINT"■":PRINT"RIGHT":FOR X=1 TO 800:NEXT:GOTO 100
12000 REM RACE ME:PRINT"■"
12010 PRINT"YOU WILL RACE ME IN A MAZE TO THE TOP"
12020 PRINT"USE 4-LEFT"
12030 PRINT"  8-UP"
12040 PRINT"  6-RIGHT"
12060 PRINT"PRESS WHEN READY"
12070 GET A$:IF A$="" THEN 12070
12080 PRINT"  ▌ ▐▀▀                ▌ ▐▀▀"
12090 PRINT"  ▌ ▐ ▌                ▌ ▐ ▌"
12100 PRINT"  ▌▄▐ ▌                ▌▄▐ ▌"
12110 PRINT"  ▌ ▐ ▌                ▌ ▐ ▌"
12120 PRINT"  ▌ ▐▀▀                ▌ ▐▀▀"
12130 PRINT"  ▌ ▐ ▌                ▌ ▐ ▌"
12140 PRINT"  ▌▄▐ ▌                ▌▄▐ ▌"
12150 PRINT"  ▌ ▐ ▌                ▌ ▐ ▌"
12160 PRINT"  ▌ ▐▀▀                ▌ ▐▀▀"
12170 PRINT"  ▌ ▐▀▀                ▌ ▐▀▀"
12180 PRINT"  ▌ ▐ ▌                ▌ ▐ ▌"
12190 PRINT"  ▌▄▐ ▌                ▌▄▐ ▌"
12200 PRINT"  ▌ ▐▀▀                ▌ ▐▀▀"
12210 PRINT"  ▌ ▐▀▀                ▌ ▐▀▀"
12220 PRINT"  ▌ ▐ ▌                ▌ ▐ ▌"
12230 PRINT"  ▀▀▀▀                 ▀▀▀▀"
12240 Q1=54093:Q2=54111
12245 GET A$
12250 POKE Q1,(207):POKE Q2,(207)
12270 IF A$<>"4" THEN GOTO 12280
12275 IF PEEK(Q1-1)=0 THEN Q1=Q1-1:GOTO 12310
12280 IF A$<>"8" THEN GOTO 12290
12285 IF PEEK(Q1-40)=0 THEN Q1=Q1-40:GOTO 12310
```

```
12290 IF A$<>"6" THEN GOTO 12310
12295 IF PEEK(Q1+1)=0 THEN Q1=Q1+1:GOTO 12310
12310 IF INT(50*RND(1))<22 THEN 12245
12320 IF PEEK(Q2-1)=0 THEN Q2=Q2-1:GOTO 12355
12340 IF PEEK(Q2+1)=0 THEN Q2=Q2+1:GOTO 12355
12350 IF PEEK(Q2-40)=0 THEN Q2=Q2-40:GOTO 12355
12355 IF Q1=53491 THEN 12500
12356 IF Q2=53509 THEN 12600
12400 GOTO 12245
12500 MUSIC"AA+AA+AA+AA"
12510 PRINT"■";"YOU DID IT!"
12520 FOR X=1 TO 1500:NEXT
12530 GOTO 100
12600 MUSIC"D-D-D-D-D-D"
12610 PRINT"■";"BEAT YOU!"
12620 FOR X=1 TO 1500:NEXT
12630 GOTO 90
12999 END
20000 MUSIC"DA+EAAA+A-A+AAA-AG+AD+EAGDA"
20010 PRINT"■"
20020 PRINT"YOU HAVE DONE IT!!"
20030 INPUT"WILL YOU HAVE ANOTHER GO,MASTER?";Z$
20040 IF LEFT$(Z$,1)="Y"THEN 90
20050 PRINT"BYE"
20060 END
```

Sharpsoft Ltd., 86-90 Paul Street, London EC2A 4NE
Printed by Oldham Press (T.U.) Chatham, Kent.