```
!                ** EDIT4B **
!
!
!    THE SOURCE TEXT OF THIS EDITOR IS IN THREE DISTINCT PARTS.
! THE FIRST OF THESE, 'EDIT4A', IS VARIABLE BETWEEN DIFFERENT
! VERSIONS OF THE PROGRAM. IT DEFINES VARIOUS PROPERTIES OF
! THE EDITOR SUCH AS MAXIMUM LINE LENGTH, MAIN BUFFER
! SIZE AND SO ON, BUT MORE IMPORTANTLY, THE NAMES AND EFFECTS OF
! ALL THE COMMANDS TO WHICH THAT PARTICULAR VERSION WILL
! RESPOND.  TO EXTEND THE COMMANDS AVAILABLE, ONLY THIS TABLE
! NEED BE ALTERED IF THE NEW COMMANDS CAN BE DEFINED IN TERMS OF
! OPERATIONS PROVIDED BY THE BASIC EDITOR.
!    THE OTHER TWO PARTS ARE FIXED. THE FIRST OF THESE 'EDIT4B',
! CONSISTS PRIMARILY OF THE COMMAND DECODER WHILE THE LAST, 'EDIT4C'
! IS THE EDITOR PROPER.  IN OPERATION, EDIT4B GENERATES A PROGRAM
! FROM THE USER'S COMMAND STRING AND THE TABLE IN EDIT4A. THIS
! PROGRAM IS THEN EXECUTED INTERPRETIVELY BY EDIT4C.
!
!    TO GENERATE A NEW VERSION OF THE EDITOR, RUN 'EDIGEN' WITH
! 'EDIDEF' AS INPUT TO GIVE 'EDIT4A' AS OUTPUT.  ANY ERRORS ARE
! TYPED OUT ON STREAM 1,.
!  COMPILE EDIT4A+EDIT4B+EDIT4C -> .EDITOR OBJ
!
!
!
%OWNINTEGER  STAR = 100001
!
%OWNINTEGER  UP ARROW = 94, DOUBLE QUOTE = 34, TILDE = 92
!
%INTEGER  SIGNBIT ; SIGNBIT = 1 << 17
!
%OWNINTEGER  PF = 2047 ; %INTEGER  DF ; DF = \PF
%OWNINTEGER   NLDF = 20480  ;! = NL<<11 = NL IN DATA FIELD
!
%INTEGER  NOT255 ; NOT255 = \255
!
%OWNINTEGER  ON = -1, OFF = 0
!
!  THE EDITOR HAS THREE LOGICAL INPUT CHANNELS AND THREE LOGICAL
!  OUTPUT CHANNELS.  THESE ARE MAPPED ONTO SYSTEM I/O STREAMS VIA
!  THE FOLLOWING TWO ARRAYS.
!  THE USE MADE OF THESE 3 INPUT AND 3 OUTPUT CHANNELS IS:
!   CHANNEL              INPUT                          OUTPUT
!      0            COMMAND STRINGS                MESSAGES TO USER
!      1            PRIMARY INPUT FILE             EDITED OUTPUT FILE
!      2            SECONDARY INPUT FILE           NUMBERED LISTINGS
%OWNINTEGERARRAY  INPUT(0:2) = 0,1,2
%OWNINTEGERARRAY  OUTPUT(0:2) = 0,3,2
%OWNINTEGER     SOURCE FILE = 1   ;! STREAM ATTACHED TO FILE BEING EDITED
%OWNINTEGER      WORK1 = 3, WORK2 = 4
!
%INTEGER  IN                ;!   INPUT STREAM SELECTOR
!
%INTEGER  MAIN    ;!  POINTS TO MAIN BUFFER
%INTEGER  TEMP    ;!  POINTS TO TEMPORARY WORKING LIST
!
%INTEGER  ASL    ;! FREE SPACE LIST
%INTEGER  FREE     ;!  NO. OF CELLS CURRENTLY IN THAT LIST
!
%INTEGER  P    ;! WORKING POINTER USED EXCLUSIVELY BY BASIC LIST
```

```
61  ! PROCESSING ROUTINES
62  !
63  %OWNINTEGERARRAY  STATUS(0:2)=0,0,0     ;! WORD # 0 INDICATES
64  ! THAT CORRESPONDING INPUT STREAM IS 'INPUT ENDED'
65  !
66  %INTEGER  F       ;! ERROR MARKER
67  !   F = 2    ->    READING COMMAND STRING
68  !   F = 0    ->    NO ERROR
69  !   F = -1   ->    FAILURE TO FIND SPECIFIED TEXT
70  !   F = -2   ->    INDIRECT PROGRAM IS LOOPED
71  !
72  %INTEGER  AMAX,MOD1
73  %OWNINTEGER  MOD2 = 0
74  !
75  %OWNINTEGER  OLD FP = 0,OLD CBP = 0
76  %OWNINTEGER  I = 'X'      ;!  'I'  HOLDS THE SELECTOR FOR THE EDITING
77  !                         ;!       FUNCTION CURRENTLY SELECTED.  IT IS
78  !                         ;!       INITIALISED TO 'X' TO DEAL WITH
79  !                         ;!       FAULT9 OCCURRING IN THE INITIAL READ
80  !                         ;!       OF A FILE SHORTER THAN ONE BUFFER FULL
81  !
82  %SWITCH  SW,PW('A':'Z')   ;!  DISPATCH TABLES TO FUNCTIONS
83  ! 'SW' IS USED FOR INITIAL CALL, 'PW' FOR ENTRIES AFTER AN 'INPUT
84  !  ENDED' FAULT
85  !
86  !
87  %INTEGER  FP,LP  ;!   FILE AND LINE POINTER   :- 'LP'  POINTS TO
88  ! THE START OF THE LINE CONTAINING 'FP'.  A 'LINE' IS A STRING
89  ! (POSSIBLY NULL) OF CHARACTERS OTHER THAN A NEWLINE, TERMINATED
90  ! BY A NEWLINE.
91  !
92  %INTEGER  LC     ;! NUMBER OF LINES CURRENTLY ON OUTPUT FILE
93  !
94  %INTEGER  WK1,WK2,WK3,WK4   ;! VOLATILE WORK SPACE WRITTEN TO ONLY
95  !  AT MAIN PROGRAM LEVEL AND BY 'READ COMMAND STRING'
96  ! ** NOTE THAT 'FAULT9:' DOES NOT ALTER ANY OF THESE LOCATIONS **
97  !
98  %INTEGER  FP1,FP2     ;! SET BY 'SEEK'
99  %INTEGER  LTEMP       ;! WORK LOCATION FOR FUNCTION 'L'
100 %INTEGER  PFLAG    ;!  'PFLAG' # 0 IMPLIES THAT LAST COMMAND CONTAINED
101 ! A PRINT REQUEST
102 !
103 %OWNINTEGER  PLC = 0       ;! PAGE LINE COUNTER USED BY 'LPLIST'
104 !
105 !
106 !
107 %INTEGERARRAY  CB(0:TOPCB), REPT,RESET,EXIT(0:TOPLVL)
108 %INTEGER       CBP,          LVL
109 !
110 ! STORAGE USED BY SEQUENCING MECHANISM TO STORE COMMAND STRING,
111 ! REPEAT COUNTER, AND STACKED 'PROGRAM COUNTER' RESP.
112 ! 'CBP' IS THE SAID PROGRAM COUNTER.
113 ! 'RESET' AND 'EXIT' HOLD START AND FINISH POINTERS FOR THE CURRENT
114 ! LEVEL OF BRACKETTING
115 !
116 %OWNINTEGER  BATCH = 0     ;!  BATCH MODE SWITCH - 0 'OFF', -1 'ON'
117 %OWNINTEGER  LISTING = 0    ;!  LISTING SWITCH - 0 'OFF', -1 'ON'
118 %OWNINTEGER  LISTSW = 0     ;!  LISTING RESET - 0 'OFF', -1 'ON'
119 %INTEGER  CL    ;!  IN BATCH MODE, HOLDS LINE NUMBER OF CURRENT
120 !               LINE AS IN ORIGINAL, EXCLUSIVE OF ANY ALTERATIONS MADE.
```

(bit missed out here)

```
121   !  NOW SET UP LIST ARRAY. SIZE IS SET DYNAMICALLY TO ALLOW FOR
122   ! RUN-TO-RUN VARIATION IN DEVICE HANDLERS REQUIRED AND BETWEEN-MACHINE
123   ! VARIATION IN CORE AVAILABLE
124   !
125   ! REDUCE 'LISTMAX' TO THE LARGEST VALUE POSSIBLE IF THE VALUE
126   ! REQUESTED IN 'EDIT4A' IS TOO BIG TO FIT INTO THE CORE AVAILABLE.
127   !
128      AMAX = FREE STORE - 800    ;!  ALLOW FOR 3 I/O BUFFERS + STACK
129      LISTMAX = AMAX %IF AMAX - LISTMAX < 0
130   !
131   !
132   !
133   %INTEGERARRAY  LIST(1:AMAX)
134   !
135   !
136   %INTEGERFN UCASE(%INTEGER  I)
137      %RESULT = I %IF I & 96 # 96
138      %RESULT = I - 32
139   %END
140   !
141   !
142   !
143   !
144   !
145   !
146   ! THE FOLLOWING TWO FUNCTIONS PERFORM THE MAPPING OF 7-BIT CODES
147   ! TO AN INTERNAL REPRESENTATION AND BACK AGAIN.  IN THIS PARTICULAR
148   ! IMPLEMENTATION IT RESTRICTS THE CHARACTER SET TO 63.  ON
149   ! A MACHINE WITH A DIFFERENT WORD LENGTH THAN N*6 BITS IT COULD
150   ! BE WORTH ALTERING THIS MAPPING.
151   %INTEGERFN  SIXBIT(%INTEGER Q)
152   %OWNINTEGER  SIXBIT UP ARROW = 30
153      %IF Q # NL %THENSTART
154         %RESULT = Q & 63 %IF Q # '@'
155         %RESULT = SIXBIT UP ARROW
156      %FINISH
157      %RESULT = DOUBLE QUOTE
158   %END
159   !
160   !
161   !
162   !
163   %INTEGERFN  SEVENBIT(%INTEGER  Q)
164   %OWNINTEGER  SIXBIT UP ARROW = 30
165      Q = Q & 63
166      %IF Q # DOUBLE QUOTE %THENSTART
167         %RESULT = (((\Q)&32)<<1)!!Q %IF Q # SIXBIT UP ARROW
168         %RESULT = '@'
169      %FINISH
170      %RESULT = NL
171   %END
172   !
173   !
174   !
175   !
176   %ROUTINE  IFC(%INTEGER  N)
177      SELECT INPUT( INPUT(N) )
178   %END
179   !
180   !
```

```
!
!
%ROUTINE  OTC(%INTEGER  N)
   SELECT OUTPUT( OUTPUT(N) )
%END
!
!
%ROUTINE  PAGE
   NEWLINES(PLC+2)
   PRINT SYMBOL('-') ; SPACES(78) ; PRINT SYMBOL('-')
   NEWLINES(4) ; PLC = 60
%END
!
!
%ROUTINE  LPLIST
%INTEGER  I,J
   PLC = PLC - 1       ;!    SET PLC = -1 TO INITIALISE
   PAGE %IF PLC <= 0
   WRITE(LC+1,4) ; SPACES(3)
   I = LP
1: %RETURN %IF I = 0
   I = LIST(I&PF) ; J = I >> 11
   PRINT SYMBOL(J)
   -> 1 %IF J # NL
%END
!
!
!
!
%ROUTINE  PUNCH(%INTEGERNAME  Z)
%INTEGER  I,TLP
!  THIS ROUTINE  DUMPS THE WHOLE OF LIST 'Z' ONTO THE CURRENT PRIMARY
!  OUTPUT STREAM.
!  IF 'LISTING' IS 'ON' A LINE-NUMBERED LISTING IS ALSO
!  OUTPUT TO THE SECONDARY OUTPUT STREAM.
!
   OTC(1)
   TLP = LP ; LP = Z & PF & LISTING
1: %IF Z # 0 %THENSTART
      I = LIST(Z)
      %IF LP # 0 %AND I # 0 %THENSTART
         OTC(2)
         LPLIST
         LP = 0
         OTC(1)
      %FINISH
      PRINT SYMBOL(I >> 11) %IF I & DF # 0
      %IF I & DF = NLDF %THENSTART
         LC = LC + 1 ; LP = I & PF & LISTING
      %FINISH
      LIST(Z) = ASL ; ASL = Z
      Z = I & PF ; FREE = FREE + 1
      -> 1
   %FINISH
!
   LP = TLP ; OLD FP = 0    ;!  CLEAR LOOP TRAP FP
   OTC(0)
%END
!
!
```

```
241   !
242   !
243   %ROUTINE  SETLP
244   !       THIS ROUTINE SETS 'LP' TO THE START OF THE LINE
245   ! CURRENTLY CONTAINING 'FP' ..... BUT, FOR EFFICIENCY'S SAKE
246   ! IT ONLY SEARCHES FORWARD FROM THE CURRENT POSITION OF 'LP'
247   ! AND FAILS IF 'FP' IS NOT FOUND.
248   !
249   %INTEGER  I,J
250       I = LP
251   1: %IF I # FP %THENSTART
252           J = LIST(I) ; I = J & PF
253           -> 1 %IF J&DF # NLDF
254           LP = I ; -> 1
255       %FINISH
256   %END
257   !
258   !
259   !
260   !
261   %ROUTINE  PUNCH TO LP
262   !   THIS ROUTINE 'PUNCHES' DOWN TO BUT EXCLUDING LINE 'LP'
263   !   AND LEAVES MAIN = LP
264   %INTEGER  I,J
265       SETLP ; %RETURN %IF MAIN = LP
266       I = MAIN
267   1: %IF MAIN # LP %THENSTART
268           J = MAIN ; MAIN = LIST(MAIN) & PF ; -> 1
269       %FINISH
270       LIST(J) = LIST(J) & DF
271       PUNCH(I)
272   %END
273   !
274   !
275   !
276   !
277   !
278   !
279   %ROUTINE  ABORT(%INTEGER  N)
280   %SWITCH  S(0:6)
281       OTC(0) ; NEWLINES(2)
282       %PRINTTEXT '** IMP EDIT:  '
283       -> S(N)
284   !
285   S(0): %PRINTTEXT 'CONTEXT?' ; -> 2
286   S(1): %PRINTTEXT 'DECODE?' ; -> 2
287   S(2): %PRINTTEXT 'EXECUTE?' ; -> 2
288   S(3): %PRINTTEXT '$CLOSE' ; -> 34
289   S(4): %PRINTTEXT '::'
290   34:    %PRINTTEXT ' MISSING' ; -> 2
291   S(5): %PRINTTEXT 'ASL EMPTY' ; -> 2
292   S(6): %PRINTTEXT 'EDITOR CORRUPT'
293   !
294   2: PUNCH TO LP
295       WRITE(CL,6) ; PRINT SYMBOL('/') ; WRITE(LC+1,1)
296       OTC(2) ; CLOSE OUTPUT     ;!  CLOSE OFF LISTING DEVICE
297       %MONITOR 31
298   %END
299   !
300   !
```

```
!
!
%ROUTINE  ERROR(%INTEGER  N)
%SWITCH S(1:4)
    OTC(0) ; -> S(N)
!
S(1): %PRINTTEXT '*OFF TOP' ; -> 1
S(2): %PRINTTEXT '*EOF' ; WRITE(IN,1) ; -> 1
S(3): %RETURN %IF LVL # 0 ; %PRINTTEXT '*EOL' ; -> 1
S(4): %PRINTTEXT '*LOOP TRAP'
!
1: NEWLINE ; ABORT(2) %IF BATCH = ON
%END
!
!
!
!
%ROUTINE  READ COMMAND STRING
%INTEGER  I,J,X,Y,TYPE,P,SOURCE,TMODE,MODE,LINE TEXT
%OWNINTEGER  TABLE = 1, TT = 2, OK = 0
!
%ROUTINE  STORE(%INTEGER  I)
    CB(CBP) = ( WK4 << 7 ) !! ( I & 127 ) ; CBP = CBP - 1
%END
!
!
%ROUTINE  PACK(%INTEGER  P)
%INTEGER  I,J
    J = 0
%CYCLE  I = 0,1,2
    J = ( J<<6 ) !! SIXBIT( (CB(P+I) ) )
%REPEAT
    CB(P) = J
%END
!
!
!
%INTEGERFN  TERM
    %RESULT = I %IF I = S
    %RESULT = \I
%END
!
%INTEGERFN  DIGIT
    %RESULT = I %IF '1'<= I <='9'
    %RESULT = I %IF I = '0' %AND SOURCE = TABLE
    %RESULT = \I
%END
!
%INTEGERFN  LETTER
    %RESULT = I %IF 'A'<= I <='Z'
    %RESULT = \I
%END
!
%INTEGERFN  RESERVED
    %RESULT = I %IF I = MACRO MARKER %OR I = REUSETEXT
    %RESULT = I %IF I = '*' %OR I = '(' %OR I = ')' %OR I = TERM
    %RESULT = \I
%END
!
!   THE FOLLOWING FUNCTION AND TWO ROUTINES RUN PARALLEL TO
```

```
361   ! NEXT SYMBOL, READ SYMBOL, AND SKIP SYMBOL RESPECTIVELY.  THE
362   ! DIFFERENCE IS THAT THESE TAKE THEIR INPUT FROM EITHER
363   ! THE EDITOR COMMAND STREAM 'TT' OR THE MACRO DEFINITION
364   ! TABLE DEPENDING ON WHETHER 'SOURCE' =TT OR =TABLE  RESPECTIVELY.
365   ! THERE IS AN AUTOMATIC SWITCH BACK TO 'TT'
366   ! WHEN THE END OF THE CURRENT DEFN IS REACHED.
367   %INTEGERFN   NEXT CHAR
368      %RESULT = UCASE(NEXT SYMBOL) %IF SOURCE = TT
369      %IF P = 0 %THENSTART
370         X = X + 1
371         %IF X = Y %OR Y = -1 %THENSTART      ;!   END OF ENTRY
372            Y = -1 ; %RESULT = -1
373         %FINISH
374         P = MACRO(X)
375      %FINISH
376      %RESULT = SEVENBIT(P)
377   %END
378   !
379   !
380   %ROUTINE  READ CHAR(%INTEGERNAME  I)
381      I = NEXT CHAR
382      %IF SOURCE = TT %THEN SKIP SYMBOL %ELSE P = P >> 6
383      PRINT SYMBOL(I) %IF SOURCE & BATCH = TT
384   %END
385   !
386   !
387   %ROUTINE  SKIP CHAR
388   %INTEGER  I
389      READ CHAR(I)
390   %END
391   !
392   !
393   !
394   !
395   %ROUTINE  SET TYPE
396   !  THIS ROUTINE SUPPLIES THE DECODER WITH THE PARAMETER TYPES REQUIRED
397   !  BY THE VARIOUS EDITING PRIMITIVES AVAILABLE FROM 'EDIT4C'. REQUESTS
398   !  FOR NON-EXISTENT PRIMITIVES ARE DETECTED AT THIS STAGE.
399   %OWNINTEGER  CYO=13888 ,ADI=4361 ,UBO=86144 ,PLG=66311 ,XTN=99598
400   %OWNINTEGER  MSZ = 54490
401   %ROUTINE  TRY(%INTEGER  P)
402   %INTEGER  K,L
403      L = SIXBIT(I)
404   %CYCLE  K = 1,1,3
405      %IF P&63 = L %THENSTART
406         TYPE = !TYPE! ; %RETURN
407      %FINISH
408      P = P >> 6
409   %REPEAT
410   %END
411   !
412      TYPE = -1 ; TRY(CYO) ; %RETURN %IF TYPE > 0
413      TYPE = -2 ; TRY(ADI) ; TRY(UBO) ; %RETURN %IF TYPE > 0
414      TYPE = -3 ; TRY(PLG) ; TRY(XTN) ; %RETURN %IF TYPE > 0
415      TYPE = -4 ; TRY(MSZ) ; %RETURN %IF TYPE > 0
416      TYPE = 0
417   %END
418   !
419   !
420   !
```

```
421   !    THIS ROUTINE IS CALLED TO READ IN TEXT PARAMETERS.  THE ONLY
422   ! TRICK IS THAT 'TMODE' IS USED TO REMEMBER WHETHER THE LAST
423   ! PARAMETER WAS A TEXT ONE OR NOT. IF IT WAS THE REDUNDANT DELIMITER
424   ! IS OMITTED. E.G.  C/FRED//JIM/ -> C/FRED/JIM/
425   %ROUTINE  READ STRING
426   %INTEGER  L,M
427   %OWNINTEGER  DEL
428      L = WK4 ; M = 0
429      -> 2 %IF TMODE = SOURCE = TT    ;!  ALREADY IN TEXT MODE ?  - YES
430      TMODE = SOURCE
431   1: READ CHAR(I) ; -> 1 %IF I = ' ' %OR I = NL
432      %IF I = REUSETEXT %THENSTART
433         %IF LINE TEXT < 0 %THENSTART
434            TYPE = -5 ; %RETURN    ;!  SYNTAX ERROR
435         %FINISH
436         CB(CBP+1) = (LINE TEXT << 7)!!( CB(CBP+1) & 127 )
437         TMODE = 0
438         %RETURN
439      %FINISH
440      DEL = I
441      -> 2 %IF I # LETTER %AND I # DIGIT %AND I # RESERVED
442      TYPE = -3 ; %RETURN     ;! FAULTY TEXT DELIMITER
443   !
444   2: %IF WK4 = CBP %THENSTART      ;! COMMAND TOO LONG
445         TYPE = -2 ; %RETURN
446      %FINISH
447   !
448      %IF  ( WK4 - L ) - LINE LENGTH >= 0 %THENSTART
449         TYPE = -1 ; %RETURN
450      %FINISH
451   !
452      READ CHAR(I)
453      -> 3 %IF I = DEL
454      %IF I = NL %THENSTART
455         M = M + 1
456         ABORT(1) %IF SOURCE & BATCH = TT  ;! 'ON' = -1 IN 2'S COMP
457      %FINISH
458      WK4 = WK4 + 1 ; CB(WK4) = I << 11 ; -> 2
459   !
460   3: %IF L = WK4 %THENSTART     ;!  DELETE ALL REFERENCES TO NULL ...
461         TYPE = -4 ; %RETURN     ;!   ... TEXT PARAMETERS
462      %FINISH
463   !
464      LINE TEXT = L %IF SOURCE = TT
465   !
466   !  NOW SET LENGTH CELL TO HOLD:
467   !  ( NUMBER OF NEWLINES << 8 ) + ( LENGTH OF TEXT STRING )
468   !  THE POINTER TO THIS CELL IS SET UP BY THE CALLING PROGRAM
469      CB(L) = (M << 8) + ( (WK4 - L) & 255 )
470      WK4 = WK4 + 1
471   %END
472   !
473   !
474   !
475   !    THIS ROUTINE IS CALLED TO READ IN NUMERICAL PARAMETERS.
476   ! THE SIGN AND DIGIT PART ARE BOTH OPTIONAL AND BY DEFAULT ARE
477   ! '+' AND '1' RESPECTIVELY. THUS <NULL> = + = +1 = 1  AND
478   ! - = -1.  ALL NUMBERS ARE TAKEN AS MODULO 'STAR'.  '*' IS TREATED
479   ! AS 'STAR'  '-*' IS ACCEPTABLE.
480   !    DIGITS ARE TAKEN TO BE DECIMAL.
```

```
%ROUTINE  READ NO(%INTEGERNAME  X)
%INTEGER  SIGN
    X = 0 ; SIGN = 0 ; -> 1
0: SKIP CHAR
1: I = NEXT CHAR ; -> 0 %IF I = ' ' %OR I = '+'
    -> 40 %IF I = '*'
    -> 4 %IF I = DIGIT ; -> 2 %IF I = '-' %AND SIGN = 0
    X = 1 ; -> 5
!
2: SIGN = -1 ; -> 0
3: I = NEXT CHAR ; -> 5 %UNLESS '0'<= I <='9'
4: X = ( ( ( X<<2 ) + X )<<1 ) + I - '0'
    SKIP CHAR ; -> 3
40: X = STAR ; SKIP CHAR ; -> 6        ;!  PERMITS -*
5: X = X - STAR + 1 %IF X >= STAR       ;!  REDUCE TO MODULO(STAR) + 1
6: X = -X %IF SIGN < 0
%END
!
!
!
!          ** END OF ROUTINES **
!
!    THE DECODER ASSEMBLES SYMBOLIC COMMANDS INTO THE COMMAND BUFFER
! USING THE COMMAND NAME TYPED IN TO REFERENCE THE MACRO TABLE. ALL
! COMMANDS GO VIA THIS TABLE.
!    PARAMETERS ARE LOADED INTO 'CB' UPWARDS FROM THE BOTTOM,
! FUNCTION CODES FROM THE TOP DOWNWARDS.  ANY FREE SPACE IS THUS LEFT
! IN THE MIDDLE. 'WK4' POINTS TO THE BOTTOM END OF IT, 'CBP' TO THE
! TOP.  THIS FREE SPACE IS ALSO USED AS TEMPORARY WORK SPACE WHEN
! READING IN MACRO NAMES.
!
!
    CLOSE INPUT %IF F = 2  ;! RE-ENTRY AFTER CONTROL D - YES
    ABORT(3) %IF F & BATCH = 2   ;!  $CLOSE MISSING
    IFC(0)
    OTC(0) ; F = 2
0: %IF BATCH # ON %THENSTART
       MODE = 2048 ; NEWLINE %IF PFLAG # 0 ; PROMPT('>')
    %FINISHELSESTART
       MODE = 512 ; PUNCH TO LP
    %FINISH
    PFLAG = 0
    WK4 = 0 ; CBP = TOPCB ; LVL = 0 ; LINE TEXT = -1
!
1: SOURCE = TT ; READ CHAR(I)
    TMODE = 0
    -> 4 %IF I = '!'
    -> 5 %IF I = TERM
    -> 1 %IF I = ' ' %OR I = NL
    OK = 0       ;! YOU CAN'T RESTART A PARTLY CORRUPTED PREVIOUS COMMAND
    -> 6 %IF I = '('
    -> 7 %IF I = ')'
!
! READ IN FUNCTION NAME FROM TT - ERROR CHECKING LEFT TO LOOK-UP PHASE
!
    X = WK4 + 1
    %IF I # MACRO MARKER %THENSTART   ;!  MULTI-LETTER FN. NAME ? - NO
       %IF I = ':' = NEXT CHAR %THENSTART  ;! INPUT MODE ? - YES
10:       SKIP CHAR ; -> 10 %IF NEXT CHAR = ' '
          -> 90 %IF NEXT CHAR # NL
```

```
541             I = MACRO(1) ;!  NAME PART OF INPUT MODE ENTRY IN MACRO TABLE
542     ! ***** NOTE THAT THE ENTRY FOR INPUT MODE('::') MUST COME FIRST *****
543           %FINISHELSESTART
544             I = I << 12
545           %FINISH
546     !
547           CB(WK4) = 2 ; CB(X) = I
548         %FINISHELSESTART
549     !
550           CB(X) = MACRO MARKER ; WK1 = 1
551     2:    %CYCLE  J = WK1,1,2
552             CB(X+J) = 0
553           %REPEAT
554           %CYCLE  J = WK1,1,2
555             I = NEXT CHAR ; -> 20 %IF I # LETTER
556             SKIP CHAR ; CB(X+J) = I
557           %REPEAT
558     !
559     20:   PACK(X) ; X = X + 1 ; WK1 = 0
560           I = NEXT CHAR ; -> 2 %IF I = LETTER
561           CB(WK4) = X - WK4
562         %FINISH
563     !
564     !   NOW LOOK UP NAME IN MACRO DEFN TABLE
565     !
566         WK1 = CB(WK4) ; X = 0
567     21: Y = MACRO(X) ; -> 83 %IF Y = -1
568         %IF Y # WK1 %THENSTART
569     22:   X = X + Y ; X = ( X + MACRO(X) ) & 255 ; -> 21
570         %FINISH
571     !
572         %CYCLE  J = 1,1,Y-1
573           -> 22 %IF CB(WK4+J) # MACRO(X+J)
574         %REPEAT
575     !   MACRO NAME HAS BEEN FOUND AT THIS POINT
576     !
577         X = X + J + 1                    ;! SET POINTER TO DEFINITION
578         WK3 = MACRO(X)
579         -> 83 %IF WK3 & MODE = 0     ;!  WRONG MODE ... (BATCH/INTERACTIVE)
580         Y = X + (WK3 & 255) ; X = X + 1 ; P = MACRO(X)
581     !
582     !
583     !    NOW PROCESS THIS COMMAND ITEM
584     !
585     3:  SOURCE = TABLE ; READ CHAR(I)
586         -> 1 %IF Y = -1     ;! END OF MACRO TABLE ENTRY
587         -> 6 %IF I = '('
588         -> 7 %IF I = ')'
589         SET TYPE ; -> 84 %IF TYPE = 0
590         WK2 = CBP ; STORE(I)     ;! STORES  ( WK4 << 7 )!!( I & 127 )
591         -> 3 %IF TYPE = 1         ;! NO PARAMETER NEEDED
592     !
593         %IF TYPE # 2  %THENSTART     ;!  TEXT PARAMETER ? - NO
594         J = WK4 ; WK4 = WK4 + 1
595     30: %IF NEXT CHAR = '#' %THENSTART     ;!  CALL FOR PARAMETER FROM TT
596           -> 84 %IF SOURCE # TABLE
597           SKIP CHAR ; SOURCE = TT
598           TMODE = 0          ;!  END OF CONSECUTIVE TEXT PARAMETERS
599         %FINISH
600         READ NO(WK1)
```

```
601         %IF WK1 < 0 %THENSTART     ;!  NEGATIVE PARAMETER ? - YES
602            -> 89 %IF TYPE = 3      ;!  ILLEGAL FOR TYPE = 3
603            -> 89 %IF WK3 & (MODE >> 1) = 0      ;!  WRONG MODE
604            WK1 = -WK1      ;!   FOR TYPE = 4 MAKE PARAMETER POSITIVE AND ...
605            CB(WK2) = CB(WK2) - 1    ;! .... CHANGE FUNCTION N TO FUNCTION N-1
606         %FINISH
607         CB(J) = WK1
608         -> 3
609         %FINISH
610      !
611      !
612      !   MUST BE A TEXT PARAMETER
613      !
614         %IF NEXTCHAR = TILDE %THENSTART     ;!  GET PARAMETER FROM TT
615            -> 84 %IF SOURCE # TABLE
616            SKIP CHAR ; SOURCE = TT
617         %FINISH
618         READ STRING
619         -> 85 %IF TYPE = -3 ; -> 80 %IF TYPE = -2
620         -> 81 %IF TYPE = -1 ; -> 880 %IF TYPE = -5
621         CBP = CBP + 1 %IF TYPE = -4  ;! REMOVE REFERENCE TO NULL STRING
622         -> 3
623      !
624      !
625      !
626      !     SPECIAL ROUTINES TO PROCESS CONTROL SYMBOLS FOLLOW
627      !
628      !
629      !
630      4: SOURCE = TT          ;!    '!' :-  RESTART PREVIOUS COMMAND
631         -> 88 %IF OK = 0     ;!  COMMAND BUFFER IS EMPTY
632         READ NO( REPT(0) ) ; -> 89 %IF REPT(0) < 0
633         -> 88 %IF NEXT SYMBOL # TERM
634         I = 0          ;!  SET  I # TERM
635         -> 51
636      !
637      !
638      !
639      !  ENTER HERE WHEN COMMAND TERMINATOR IS FOUND
640      5: -> 0 %IF CBP = TOPCB    ;!   EMPTY COMMAND STRING ? - YES
641         REPT(0) = 1 ; RESET(0) = TOPCB ; EXIT(0) = CBP ; STORE(')')
642      51: -> 91 %IF LVL # 0
643         CBP = TOPCB ; F = 0 ; OLD CBP = 0
644         OK = -1              ;! COMMAND BUFFER SUCCESSFULLY LOADED
645      52: %RETURN %IF I = TERM ; READ SYMBOL(I) ; -> 52
646      !
647      !
648      !
649      6: STORE('(')          ;!   PROCESS LEFT PARENTHESIS
650         LVL = LVL + 1 ; REPT(LVL) = WK4 ; WK4 = WK4 + 2
651         %IF SOURCE = TABLE %THEN -> 3 %ELSE -> 1
652      !
653      !
654      !
655      7: STORE(')')        ;!   PROCESS RIGHT PARENTHESIS
656         J = REPT(LVL) ; LVL = LVL - 1 ; -> 91 %IF LVL < 0
657         CB(J+1) = CBP          ;!   EXIT ADDRESS
658         TYPE = 3 ; -> 30 %IF SOURCE = TABLE
659         READ NO( CB(J) ) ; -> 89 %IF CB(J) < 0
660         -> 1
```

```
661   !
662   !
663   !
664   80: %PRINTTEXT 'INSTR' ; -> 82
665   81: %PRINTTEXT 'TEXT'
666   82: %PRINTTEXT ' TOO LONG' ; -> 93
667   !
668   83: %PRINTTEXT 'FN='''
669       %CYCLE  I = 1,1,CB(WK4) - 1
670           %CYCLE  J = 12,-6,0
671               WK1 = CB( WK4 + I ) >> J
672               PRINTSYMBOL( SEVENBIT(WK1) ) %IF WK1 & 63 # 0
673           %REPEAT
674       %REPEAT
675       -> 86
676   !
677   84: ABORT(6)          ;!  ERROR FOUND IN MACRO TABLE : 'EDITOR CORRUPT'
678   85: %PRINTTEXT 'QUOTE=''' ; PRINT SYMBOL(I)
679   86: %PRINTTEXT ''''' ; -> 92
680   !
681   88: %PRINTTEXT '!' ; -> 881
682   880: PRINT SYMBOL(REUSETEXT)
683   881: %PRINTTEXT ' IMPOSSIBLE' ; -> 93
684   !
685   89: %PRINTTEXT '-VE NO.' ; -> 92
686   90: %PRINTTEXT 'SYNTAX' ; -> 92
687   91: %PRINTTEXT 'BRACKETS'
688   !
689   92: %PRINTTEXT ' ?'
690   93: READ SYMBOL(I) ; -> 93 %IF I # NL
691       PFLAG = 1 ; -> 0 %IF BATCH = OFF ; ABORT(1)
692   !
693   %END
694   !
695   !
696   !
697   !
698   %ROUTINE  INIT
699   !    THIS ROUTINE SETS UP THE FREE SPACE LIST WITH 'ASL' POINTING TO THE
700   ! BEGINNING OF IT.  ALSO THE MAIN BUFFER IS SET UP TO BE ONE
701   ! END-OF-BUFFER MARKER POINTED TO BY 'MAIN', 'LP' AND 'FP'.
702   !     'FREE' IS INITIALISED TO THE NUMBER OF CELLS IN THE ASL.
703   !     'LC' IS SET TO ZERO AS 'MAIN' MUST BE EMPTY AT THIS POINT.
704   %INTEGER  I
705       %CYCLE  I = 1,1,LISTMAX-1
706           LIST(I) = I - 1
707       %REPEAT
708       ASL = I ; FREE = I  ;! FREE HOLDS NUMBER OF CELLS IN LIST 'ASL'
709       MAIN = LISTMAX ; LIST(MAIN) = 0  ;! SET UP END MARKER ON LIST 'MAIN'
710       LP = MAIN ; FP = LP
711       LC = 0 ; CL = 1
712       TEMP = 0 ; P = 0
713       MOD2 = MOD1 %IF MOD1 # 0 ; MOD1 = 0
714   %END
715   !
716   !
717   !
718   %ROUTINE  JOIN ON(%INTEGERNAME  X)
719   !    THIS ROUTINE APPENDS LIST 'X' TO LIST 'MAIN'. 'X' IS SET TO
720   ! ZERO ON EXIT
```

```
!   **** ESSENTIAL TO THE SPEC. IS THAT THE FIRST CELL OF LIST 'X'
!   OVERWRITES THE PHYSICAL REGISTER OCCUPIED BY THE TERMINATOR
!   CELL ON LIST 'MAIN'.  THIS MEANS THAT AFTER 'JOINING ON' A POINTER
!   PREVIOUSLY LEFT POINTING TO THE TERMINATOR, POINTS TO THE HEAD
!   OF THE PIECE JOINED ON. ****
%INTEGER  I,J,K
    %RETURN %IF X = 0  ;!  EXIT IF THERE'S NOTHING TO JOIN ON
    J = MAIN
1:  I = J      ;!      FIND END OF MAIN LIST IN BUFFER
    J = LIST(I) & PF
    -> 1 %IF J # 0
!   AT THIS POINT 'I' IS POINTING TO THE END MARKER CELL.  OVERWRITE
!   THE END MARKER WITH THE FIRST CELL OF LIST 'X' AND VICE VERSA
!   THIS JOINS 'X' ONTO 'MAIN' AND ALSO REDUCES 'X' TO AN END MARKER CELL
!
    K = LIST(I) ; LIST(I) = LIST(X) ; LIST(X) = K
!
! NOW PUT THE MARKER BACK ON THE END OF THE ENLARGED MAIN LIST
!
2:  J = I
    I = LIST(J) & PF
    -> 2 %IF I # 0
!   'J' NOW POINTS TO THE LAST CELL IN 'MAIN' AND 'X' TO END MARKER CELL.
!
    LIST(J) = ( LIST(J) & DF )!!( X & PF )
    X = 0
%END
!
!
!
!
%ROUTINE  CHOP IT OFF
!   THIS ROUTINE IS CALLED TO DISCONNECT A LIST (BUILT USING 'STORE')
! FROM THE ASL
    %RETURN %IF P = 0    ;! NOTHING TO DISCONNECT
    LIST(P) = LIST(P) & DF
    P = 0
%END
!
!
!
!
%ROUTINE  STORE(%INTEGER  CHAR)
!    THIS ROUTINE IS CALLED WHENEVER A CELL IS REQUIRED FROM ASL.
! THE DATA IS STORED IN THE MORE SIGNIFICANT END OF LIST(ASL) THEN
! 'ASL' AND 'P' ADVANCED, ASL -> P, LINK -> ASL, SO THAT SUCCESSIVE
! CALLS STORE IN SUCCESSIVE CELLS IN ASL.  AFTER EACH CALL 'P'
! POINTS TO THE LAST CELL STORED IN.
%INTEGER  L
    %IF ASL = 0 %THEN ABORT(5) %ELSESTART  ;! ABORT IF ASL IS EMPTY
        P = ASL ; ASL = LIST(P) & PF
        LIST(P) = ( CHAR & DF )!! ASL     ;! STORE CHARACTER
        FREE = FREE - 1
    %FINISH
%END
!
!
!
!
%ROUTINE  READ INPUT
```

```
781   !   THIS ROUTINE READS EITHER AS MUCH AS IS POSSIBLE OR ONE LINE
782   ! DEPENDING ON WHETHER WE ARE USING MAIN OR SUBSIDIARY INPUT
783   !     IT ALWAYS READS AT LEAST ONE LINE.
784   %INTEGER  I
785      I = 0
786      %RETURN %IF STATUS(IN) # 0 ; IFC(IN)
787   !
788   1: READ SYMBOL(I) ; STORE(UCASE(I) << 11)
789      -> 1 %IF I # NL %OR (IN = 1 %AND FREE > MINIMUM FREE)
790   !
791   ! MAKE IT GO 'INPUT ENDED' NOW, RATHER THAN AT THE VERY BEGINNING
792   ! OF THE NEXT READ:- ONLY DO THIS ON PRIMARY INPUT
793      I = NEXT SYMBOL %IF IN = 1
794      STATUS(IN) = 0     ;! NORMAL TERMINATION
795      IFC(0)
796   %END
797   !
798   !
799   !
800   !
801   %ROUTINE  FILL MAIN
802   !
803   !   DUMPS DOWN TO BUT EXCLUDING LINE 'LP' ONTO BACKING STORE THEN
804   !   TOPS UP BUFFER FROM INPUT STREAM.
805   !   I.E.  CURRENT LINE BECOMES TOP LINE IN BUFFER
806   !
807      IN = 1
808   !    DON'T EMPTY THE BUFFER IF YOU KNOW YOU CAN'T REFILL IT
809      %RETURN %IF STATUS(IN) # 0
810   !
811      PUNCH TO LP    ;! PUNCHES TO 'LP' AND SETS MAIN = LP
812      TEMP = ASL ; P = 0
813      READ INPUT
814      CHOP IT OFF ; JOIN ON(TEMP)
815   %END
816   !
817   !
818   !
819   !
820   %ROUTINE  SEEK
821   !   'SEEK' SEARCHES FOR THE STRING CONTAINED IN CB(WK1+1)
822   ! TO CB(WK1+CB(WK1)) INCLUSIVE . THE SCOPE OF THE SEARCH IS FROM
823   ! THE CURRENT POSITION OF FP TO THE END OF THE CURRENT LINE.
824   ! THE SEARCH WILL HOWEVER WRAP AROUND ONTO THE NEXT LINE IF THIS
825   ! IS NECESSARY TO COMPLETE THE MATCHING OF A STRING CONTAINING
826   ! NEWLINE CHARACTERS.
827   !    FP1 IS SET TO THE START OF THE TARGET STRING, FP2 TO THE FIRST
828   ! CHARACTER AFTER IT.
829   %INTEGER  I,J,K,L,X,Y
830   !   SET UP PARAMETER POINTERS 'X' AND 'Y'
831      X = WK1 + 1 ; Y = WK1 + (CB(WK1) & 255)
832   0: K = CB(X) ; L = FP    ;! SET UP SEARCH FOR FIRST CHARACTER
833   !
834   1: -> 5 %IF L = 0 ; FP1 = L & PF
835      L = LIST(FP1) ; J = L & DF
836      -> 2 %IF J = K      ;!  GOT FIRST CHAR, GO TRY REST OF STRING
837   10: -> 4 %IF J = NLDF ; -> 1  ;! NO LUCK YET, FAIL IF AT END OF LINE
838   !
839   2: FP2 = L & PF
840      -> 3 %IF X = Y        ;!  SINGLE CHAR STRING ? - YES, SUCCESS !
```

```
841       I = X                          ;! MULTI-CHAR STRING, SET UP LOOP COUNTER
842    !
843    20: I = I + 1                     ;! BUMP COUNTER - 1ST CHAR ALREADY MATCHED
844       J = LIST(FP2) ; -> 5 %IF J = 0      ;!  END-OF-BUFFER MARKER ? - YES
845       %IF J & DF # CB(I) %THENSTART       ;!  FAILURE - GOT WRONG STRING ...
846          J = L & DF ; -> 10               ;! ... RESTART SEARCH FOR 1ST CHAR
847       %FINISH
848       FP2 = J & PF
849       -> 20 %IF I # Y                ;!  MATCHED WHOLE STRING ? - NO
850    !
851    3: F = 0                          ;! STRING FOUND
852       F = -2 %IF CBP = OLD CBP %AND FP = OLD FP ;! ARE WE IN A LOOP? - YES
853       OLD CBP = CBP ; OLD FP = FP
854       %RETURN
855    !
856    4: F = -1                         ;! SEARCH FAILURE
857       %IF LVL = 0 %THENSTART            ;! DIAGNOSTIC IF NOT ON AUTOMATIC
858          %PRINTTEXT '*'''
859          %CYCLE  I = X,1,Y
860             PRINTSYMBOL( CB(I) >> 11 )
861          %REPEAT
862          %PRINTTEXT ''' NOT FOUND' ; NEWLINE
863       %FINISH
864       %RETURN
865    !
866    5: -> 4 %IF STATUS(IN) # 0
867       FILL MAIN ; -> 0
868    %END
869    !
870    !
871    !
872    !
873    %ROUTINE  CHECK SPACE
874    !    THIS ROUTINE REMOVES LINES FROM THE TOP OF THE BUFFER AND WRITES
875    ! THEM TO BACKING STORE IF THIS IS NECESSARY.  THE CASE WHERE
876    ! THE TOP LINE IS ALSO THE CURRENT LINE IS NOT SATISFACTORILY DEALT
877    ! WITH THOUGH IN THIS PARTICULAR PROGRAM IN CAN'T CAUSE A CRASH
878    %INTEGER  I,J,K
879       %RETURN %IF FREE > LINE LENGTH %OR LP = MAIN
880       K = MAIN
881       %CYCLE  I = FREE,1,MINIMUM FREE
882          J = K ; K = LIST(J) & PF
883          -> 2 %IF K = LP
884       %REPEAT
885    1: K = LIST(J) ; -> 2 %IF K&DF = NLDF
886       J = K & PF ; -> 1
887    !
888    2: K = LIST(J) ; LIST(J) = NLDF ; PUNCH(MAIN)
889       MAIN = K & PF
890    %END
891    !
892    !
893    !
894    !
```