
240 FOR I = 1 TO 96: CALL
250 HOME : PRINT "DOWN SH
260 FOR I = 1 TO 96: CALL
270 HOME : PRI
FOR I 1
FOR I
HOME
PRINT
"BOTTOM MIR
AD"PC$",A$2
TO 2000: NEXT
CURRENT PAGE 1";E
PRINT "CURRENT PAG
= 1 TO 2000: NEXT
RLAYED ONTO 1";E$
= 1 TO 3000: NEXT
= 1 TO 3000: NEXT
AGES 1-2
PRINT

TYPE-RIGHT

Cut your debugging time in half! Type-Right catches Applesoft errors as you enter program lines.

When you enter an Applesoft program, most of the errors you make are typographical, such as leaving out a parenthesis or quotation mark, or forgetting the dollar sign on a string variable. The only way to find these errors is to run the program and hope they result in a syntax error. If they do, then you have to search through your program for the offending line, correct the error, and run the program again and again until all the errors are gone. Wouldn't it be great if your computer could issue an error message immediately after you enter a line that contains an error? Type-Right makes it possible!

Type-Right is a machine language program that you can install before typing in an Applesoft program. It resides in high memory, and examines each line of Applesoft as you enter it. If Type-Right detects an error in the line, it beeps and displays the line with the error highlighted and a message showing what kind of error was made. This gives you the chance to correct the error immediately and then continue entering your program. Type-Right can detect most errors, including syntax errors and undefined line errors, and will flag calls to machine language routines. Type-Right notifies you of errors as you enter each line, saving you the time and frustration of looking for them later.

USING TYPE-RIGHT

Using Type-Right is very easy. Just BRUN TYPE.RIGHT and it installs itself in high memory. Now go ahead and enter your Applesoft program. When you press Return after entering a program line, Type-Right checks it for errors. Try entering the following line:

```
10 X = "A"
```

Type-Right will beep and display the following:

```
1 NUMERIC VALUE EXPECTED
10 X = "A"
```

The highlighted 1 indicates that there is one error in the line. The error message tells you what kind of error it is. In this case, you are trying to set a numeric variable equal to a string value, so the message reads 'NUMERIC VALUE EXPECTED.' The Applesoft line itself is displayed with the erroneous section highlighted. In this case the A and quote mark (") characters are highlighted.

If Type-Right detects an error in the line, it beeps and displays the line with the error highlighted.

You may now correct the line. Depending on what you intended, the correct line could be:

```
10 X = "A"
```

or

```
10 X$ = "A"
```

If you want to disconnect Type-Right for any reason, type Control-K followed by Return. This completely removes Type-Right from memory. To reconnect Type-Right you must BRUN it again.

Configuration Options

Some of Type-Right functions may be turned on or off by POKEs; for example, you may not want Type-Right to flag calls to machine language routines. Table 1 shows the locations to POKE to reconfigure Type-Right. POKE a value of 128 to turn the option on, or POKE a value of 0 to turn it off.

The last three options shown in **Table 1** require some explanation. If you turn these options off, lines that contain the errors defined in the option will not be inserted into the Applesoft program. These options are on, normally, so that you can leave a line that contains an error in the program, if you want. This is useful, especially in the case

of machine language calls, and sometimes with undefined lines as well. For example, if you enter the following as the first line of a program:

```
10 GOTO 100
```

Type-Right will issue an undefined line error because there is no **line 100.** If you are planning to enter **line 100** later, you may ignore the error message and continue. The same is true of machine language calls. There is no way for Type-Right to know the correct format for a specific machine language call you may have created. Type-Right will flag the CALL instruction so that you can double check it yourself, and once you've checked it, you can continue entering the program.

Errors Type-Right Won't Catch

Type-Right will find the majority of errors, but it will also miss some types of errors. Here's a good rule of thumb: If Applesoft would not recognize the error, neither will Type-Right. The following, for example, will not generate an error message: Missing quotes at the end of a string, a THEN without any following statements, or leaving the dollar sign ($) off a string function. Lines such as these:

```
10 IF A$ = "ABC THEN PRINT A
20 IF A = 5 THEN
30 A$ = LEFT(X$,3)
```

will therefore not generate error messages.

Type-Right also can't find logic errors in your program. It checks each line for integrity, but doesn't check whether the line makes sense in the context of the program. Thus, entering **line 30** in the following program segment would not generate an error message:

```
10 FOR X = 1 TO 500
20 PRINT X
30 NEXT Y
```

Type-Right will accept line 30 because it is syntactically correct, but when you run the program, you'll get a NEXT WITHOUT FOR error because it expects a NEXT X statement.

Type-Right notifies you of errors as you enter each line.

ENTERING THE PROGRAM

To enter Type-Right you have two options. If you have an assembler, you may enter the assembly language portion of **Listing 1** and assemble it. Note that the program is too long to enter in a single source file. Your assembler must be able to generate a single object file from two source files. Some assemblers use a chain pseudo-op at the end of the first file (**line 855**), while others require you to set up a third file that includes the names of the two source files. If you don't have an assembler, or your assembler can't combine two source listings, you may enter the machine language portions of **Listing 1** directly from the Monitor (type CALL −151 to enter the Monitor). Save with:

```
BSAVE TYPE.RIGHT, A$8A00,L$B70
```

For help with entering *Nibble* listings, see the Typing Tips section.

HOW THE PROGRAM WORKS

Type-Right works by first installing itself in high memory (at $8A00) and changing the input/output (I/O) hooks so that Type-Right can intercept each line of Applesoft as it's entered. This all happens in the initialize routine, in **lines 86-112** of **Listing 1.**

Whenever a program line is entered, the intercept routine (**line 158**) is executed. This routine reads the entered line number (**line 181**), tokenizes the line (**line 182**), and checks to see if Type-Right has been damaged (**line 184**). If Type-Right is damaged, it's removed from memory and an error message is displayed. This could happen if your Applesoft program resets HIMEM and then stores some variables in the space where Type-Right resides.

Assuming all is well, the intercept routine continues at **line 187**, checking the syntax of the entered line and determining whether or not the line should be inserted in the program in **lines 188-201.**

Lines 211-236 remove Type-Right in response to a Control-K command or a checksum error. **Lines 238-255** calculate the checksum of Type-Right to determine if it's been damaged.

Lines 274-341 save the current status and Applesoft data before checking the entered line, and then they check the tokenized line in the input buffer and restore the Applesoft status and data. **Lines 375-407** produce the various sounds used by Type-Right to indicate different types of errors, and **lines 411-560** list the Applesoft line, showing the errors in inverse text. **Lines 569-853** contain all the keyword checks.

In the second source file, **lines 7-38** check for the end of a command and issue an error message if necessary. **Lines 42-135** check for undefined lines. **Lines 139-170** check expressions and their types, and **lines 172-453** make additional checks of expressions, operations, and functions. **Lines 457-510** are responsible for displaying the errors in inverse. **Lines 529-652** display the various error messages, **lines 658-721** contain the index to the keyword checks, **lines 725-734** contain the priority of the various operations (the order in which they are performed), and **lines 740-774** contain the text for all the error messages.

TABLE 1: Configuration POKE Locations

Location *	Function
$8A3A (35386)	Display syntax errors
$8A3B (35387)	Display machine language calls
$8A3C (35388)	Display undefined line errors
$8A3D (35389)	Sound for syntax errors
$8A3E (35390)	Sound for machine language calls
$8A3F (35391)	Sound for undefined line errors
$8A40 (35392)	Display line for syntax errors
$8A41 (35393)	Display line for machine language calls
$8A42 (35394)	Display line for undefined line errors
$8A43 (35395)	Insert lines with syntax errors
$8A44 (35396)	Insert lines with machine language
$8A45 (35397)	Insert lines with undefined line errors

**Location values are hexadecimal, followed by decimal equivalent in parentheses.*

Some of Type-Right's functions may be turned on or off by POKEs.

LISTING 1: TYPE.RIGHT

```
SOURCE FILE: TYPE.RIGHT.S1
SOURCE FILE: TYPE.RIGHT.S2
                1 ****************************************
0000:           2 *
0000:           3 *  TYPE.RIGHT  BY GRANT STEVENS
0000:           4 *  INTERACTIVE APPLESOFT SYNTAX CHECKER
0000:           5 *  COPYRIGHT (C) 1987
0000:           6 *  BY MICROSPARC, INC., CONCORD, MA
0000:           7 *
0000:           8 *    THIS PROGRAM SCANS APPLESOFT UTILRAM LINES AS THEY
0000:           9 *    ARE ENTERED, LOOKING FOR ERRORS.  IF ANY ARE FOUND:
0000:          10 *
0000:          11 *    - THE SPEAKER WILL SOUND.
0000:          12 *
0000:          13 *    - A MESSAGE WILL BE DISPLAYED, INDICATING THE
0000:          14 *      NATURE OF EACH ERROR.
0000:          15 *
0000:          16 *    - THE LINE WILL BE LISTED, WITH THE ERRORS
0000:          17 *      HIGHLIGHTED.
0000:          18 *
0000:          19 *    IF DATA IS PASSED TO A MACHINE LANGUAGE PROGRAM
0000:          20 *    VIA TEXT FOLLOWING A "CALL" OR "&" COMMAND, THIS
0000:          21 *    WILL BE TREATED AS AN ERROR.  THE MESSAGE "ML DATA
0000:          22 *    IGNORED" WILL BE DISPLAYED, AND A DISTINCTIVE
0000:          23 *    SOUND WILL INDICATE THIS SITUATION.
0000:          24 *
0000:          25 *    A THIRD SOUND INDICATES THE UNDEF'D LINE ERROR.
0000:          26 *
0000:          27 *    THIS PROGRAM MAY BE DISCONNECTED BY TYPING CONTROL-K,
0000:          28 *    THEN RETURN, AT THE "]" PROMPT.
0000:          29 *
               30 ****************************************
0000:          31 * APPLE TOOLKIT ASSEMBLER
0000:          32 :
0000:          33 :    RAM USAGE
0000:          34 :
0000:          35 ZREG1   EQU  $00          ;GENERAL PURPOSE REGISTER
0002:          36 ZREG2   EQU  $02          ;GENERAL PURPOSE REGISTER
0004:          37 INVFLAG EQU  $04          ;TO HIGHLIGHTING ERRORS
0006:          38 SPCFLAG EQU  $06          ;TO HIGHLIGHT MISSING ITEMS
0008:          39 ERRSYM1 EQU  $08          ;UP TO 3 SYMBOLS TO BE
0009:          40 ERRSYM2 EQU  $09          ; PRINTED IN THE "TEXT
000A:          41 ERRSYM3 EQU  $0A          ; EXPECTED" ERROR MESSAGE
000B:          42 ERCOUNT EQU  $0B          ;NO OF ERRORS THAT OCCURRED IN LINE
0011:          43 TYP.STR EQU  $11          ;FLAG INDICATES STRING/NUMERIC TYPE
0024:          44 CH      EQU  $24          ;HORIZONTAL POSITION OF CURSOR
0032:          45 INVBYTE EQU  $32          ;TO HIGHLIGHT DISPLAY OUTPUT
0050:          46 LINENUM EQU  $50          ;LINE NUMBER (AS IN GOSUB 510)
0067:          47 PROGRAM EQU  $67          ;START-OF-PROGRAM POINTER
0069:          48 LOMEM   EQU  $69          ;START OF DATA STORAGE
0073:          49 HIMEML  EQU  $73          ;END OF DATA STORAGE
00B1:          50 CHRGET  EQU  $00B1        ;"GET NEXT PROGRAM BYTE" ROUTINE
00B7:          51 CHRGOT  EQU  $00B7        ;"RE-GET LAST BYTE" ROUTINE
00B8:          52 CHRPTR  EQU  $B8          ;APPLESOFT PROGRAM COUNTER
00F3:          53 FLASHBT EQU  $F3          ;FOR FLASHING DISPLAY OUTPUT
00F8:          54 STACK   EQU  $F8          ;STORAGE FOR S-REG IN CASE OF ERROR
057B:          55 CH80    EQU  1403         ;80-COLUMN HORIZONTAL POS OF CURSOR
0000:          56 :
0000:          57 :    I/O USAGE
0000:          58 :
C030:          59 SPKR    EQU  $C030        ;SPEAKER OUTPUT ADDRESS
0000:          60 :
0000:          61 :    ROM USAGE
0000:          62 :
D410:          63 OUTMEM  EQU  $D410        ;OUT OF MEMORY ERROR
D43C:          64 EDITOR  EQU  $D43C        ;NORMAL ENTRY TO APPLESOFT
D46C:          65 INSERT1 EQU  $D46C        ;INSERT LINE INTO PROGRAM
D559:          66 PARSE   EQU  $D559        ;TOKENIZE THE INPUT BUFFER
DA0C:          67 RGETLNO EQU  $DA0C        ;GET ENTERED LINE NUMBER
DB57:          68 SPACOUT EQU  $DB57        ;OUTPUT A SPACE
DB5C:          69 COUT    EQU  $DB5C        ;OUTPUT A CHARACTER
E07D:          70 ABC.CK  EQU  $E07D        ;CHECK IF CHAR IS ALPHA (C-SET = YES)
E2F2:          71 TO.REAL EQU  $E2F2        ;CONVERT INTEGER TO REAL
ED24:          72 PRTAX   EQU  $ED24        ;PRINT INTEGER VALUE FROM A & X
F273:          73 NORMAL  EQU  $F273        ;SET NORMAL DISPLAY MODE
F277:          74 INVERSE EQU  $F277        ;SET INVERSE DISPLAY MODE
F289:          75 HIMEM   EQU  $F289        ;MAKE ROOM FOR UTIL
F94A:          76 PRBL2   EQU  $F94A        ;PRINT SPACES (QUAN IN X-REG)
FCA8:          77 WAIT    EQU  $FCA8        ;DELAY FOR SPEAKER TIMING
FD8B:          78 CROUT   EQU  $FD8B        ;OUTPUT A CARRIAGE RETURN
FF58:          79 RTS     EQU  $FF58        ;ROM RTS INSTRUCTION
0000:          80 :
0000:          81 :
----- NEXT OBJECT FILE NAME IS TYPE.RIGHT
8A00:          82          ORG  $8A00
8A00:          83 :
8A00:          84 :    INITIALIZE
8A00:          85 :
8A00:A5 73     86 START   LDA  HIMEML       ;SAVE OLD HIMEM IN CASE
8A02:8D 46 8A  87          STA  HIMSAV      ; WE ABORT UTIL
8A05:A5 74     88          LDA  HIMEML+1
8A07:8D 47 8A  89          STA  HIMSAV+1
8A0A:A9 8A     90          LDA  #<START     ;SET UP HIMEM
8A0C:A0 00     91          LDY  #>START     ;TO MAKE ROOM FOR PROG
8A0E:AE 00 BF  92          LDX  $BF00
8A11:E0 4C     93          CPX  #$4C        ;IS PRODOS ACTIVE?
8A13:D0 02     94          BNE  START.0     ; NO-O K, AS IS
8A15:A9 86     95          LDA  #<START-$0400 ; YES-LOWER HIMEM
8A17:C5 74     96 START.0 CMP  HIMEML+1     ; TO MAKE ROOM FOR GP BUFFER
8A19:90 06     97          BCC  START.1
8A1B:D0 0A     98          BNE  START.2     ;(BUT DON'T INCREASE HIMEM
8A1D:C4 73     99          CPY  HIMEML      ; OR PREVIOUSLY INSTALLED
8A1F:90 06    100          BCC  START.2     ; PROGRAMS MAY BE THREATENED)
8A21:20 F2 E2 101 START.1 JSR  TO.REAL     ;CONVERT NEW HIMEM VALUE TO REAL
8A24:20 89 F2 102          JSR  HIMEM       ;AND INSTALL IT
8A27:         103 :
8A27:20 F3 8A 104 START.2 JSR  CALCSUM     ;CALC & STORE CHECKSUM
8A2A:8D 50 8A 105          STA  CKSUM
8A2D:A9 4C    106          LDA  #$4C        ;SET UP JUMP TO
8A2F:85 C5    107          STA  $C5         ; INTERCEPT PROGRAM
8A31:A9 51    108          LDA  #>INTRCPT   ; LINES AS THEY ARE
8A33:85 C6    109          STA  $C6         ; ENTERED
8A35:A9 8A    110          LDA  #<INTRCPT   ;(INTERCEPT IS IN CHRGET ROUTINE)
8A37:85 C7    111          STA  $C7
8A39:60       112          RTS              ;GO BACK TO APPLESOFT
8A3A:         113 :
8A3A:         114 :    USER-CONFIGURABLE OPTIONS DATABASE
8A3A:         115 :          (HI BIT SET = YES)
8A3A:         116 :    THESE OPTIONS MAY BE SELECTED AT
8A3A:         117 :    ASSEMBLY TIME OR "POKED" DURING OPERATION
8A3A:         118 :
8A3A:80       119 SNMSG   DFB  $80          ;SHALL SYNTAX ERROR MESSAGES
8A3B:         120 :                        ; BE DISPLAYED?
8A3B:80       121 MLMSG   DFB  $80          ;    ML DATA MESSAGES?
8A3C:80       122 ULMSG   DFB  $80          ;    UNDEF'D LINE MESSAGES?
8A3D:80       123 SNBELL  DFB  $80          ;SHALL SPEAKER SOUND
8A3E:         124 :                        ; UPON SYNTAX ERROR?
8A3E:80       125 MLBELL  DFB  $80          ;    UPON ML DATA "ERROR"?
8A3F:80       126 ULBELL  DFB  $80          ;    UPON UNDEF'D LINE ERROR?
8A40:80       127 SNLIST  DFB  $80          ;LIST LINE IN CASE OF SYNTAX ERROR?
8A41:80       128 MLLIST  DFB  $80          ;    IN CASE OF ML DATA?
8A42:80       129 ULLIST  DFB  $80          ;    IN CASE OF UNDEF'D LINE?
8A43:80       130 SNINS   DFB  $80          ;SHALL LINES WITH SYNTAX ERRORS BE
8A44:         131 :                        ; INSERTED INTO THE PROGRAM?
8A44:80       132 MLINS   DFB  $80          ;    LINES WITH ML DATA?
8A45:80       133 ULINS   DFB  $80          ;    LINES WITH UNDEF'D LINE NUMBERS?
8A46:         134 :
8A46:         135 :    PROGRAM DATA
8A46:         136 :
8A46:00 00    137 HIMSAV  DW   $0000        ;SAVE HIMEM IN CASE WE KILL PROG
8A48:00       138 AREG    DFB  $00          ;SAVE A-REG WHILE WORKING
8A49:00 00    139 SAVCPTR DW   $0000        ;SAVE CHRPTR WHILE WORKING
8A4B:00       140 SAVINVB DFB  $00          ;SAVE INVERSE FLAG WHILE WORKING
8A4C:00       141 SAVFBYT DFB  $00          ;SAVE FLASH FLAG WHILE WORKING
8A4D:00       142 MLERR   DFB  $00          ;ML DATA ERROR FLAG
8A4E:00       143 ULERR   DFB  $00          ;UNDEF'D LINE ERROR FLAG
8A4F:00       144 SNERR   DFB  $00          ;SYNTAX ERROR FLAG
8A50:00       145 CKSUM   DFB  $00
8A51:         146 :
              147 ****************************************
8A51:         148 :
8A51:         149 CSUMBGN EQU  *            ;BEGINNING OF INTEGRITY-CHECKED AREA
8A51:         150 :
8A51:         151 :    INTERCEPT ROUTINE--WHENEVER PROGRAM LINES
8A51:         152 :    AND CERTAIN OTHER LINES ARE ENTERED, THIS
8A51:         153 :    ROUTINE IS EXECUTED.  IGNORE ANYTHING
8A51:         154 :    THAT IS NOT A PROGRAM LINE OR CONTROL-K.
8A51:         155 :    SEND PROGRAM LINES ON TO BE SYNTAX-CHECKED
8A51:         156 :    DISCONNECT UTIL ON CTRL-K
8A51:         157 :
8A51:8D 48 8A 158 INTRCPT STA  AREG         ;PRESERVE ACCUMULATOR
8A54:68       159          PLA              ;CHECK RETURN ADDRESS ON STACK
8A55:C9 4C    160          CMP  #>$D44A+2   ;WAS "JSR $00B1" EXECUTED BY THE
8A57:D0 08    161          BNE  INCPT.1     ; ROM EDITOR COMMAND AT $D44A?
8A59:68       162          PLA              ;(PULL AND RESTORE HI
8A5A:48       163          PHA              ; STACK BYTE IF NECESSARY)
8A5B:C9 D4    164          CMP  #<$D44A+2   ;(CHECK HI STACK BYTE)
8A5D:08       165          PHP
8A5E:A9 4C    166          LDA  #>$D44A+2   ;(RESTORE HI STACK BYTE)
8A60:28       167          PLP
8A61:48       168 INCPT.1 PHA              ;(RESTORE STACK RETURN ADDRESS)
8A62:D0 51    169          BNE  INCPT1      ;BACK TO APPLESOFT IF NOT FROM $D44A
8A64:20 B5 8A 170          JSR  INCPT1      ;IS THIS A NUMBERED PROGRAM LINE?
8A67:B0 48    171          BCS  INCPT.2     ; NO-DON'T TOUCH IT
8A69:68       172          PLA              ; YES-SYNTAX-CHECK IT
8A6A:68       173          PLA              ;POP RETURN ADDRESS
8A6B:20 B5 8A 174          JSR  INCPT1      ;RESTORE ACCUMULATOR AND STATUS
8A6E:A2 FF    175          LDX  #$FF
8A70:86 76    176          STX  $76         ;CLEAR CURRENT LINE NUMBER
8A72:A6 AF    177          LDX  $AF
0A74:86 69    178          STX  $69         ;CLEAR LOMEM
8A76:A6 B0    179          LDX  $B0
8A78:86 6A    180          STX  $6A         ;HI BYTE TOO
8A7A:20 0C DA 181          JSR  RGETLNO     ;READ THE ENTERED LINE NUMBER
8A7D:20 59 D5 182          JSR  PARSE       ;TOKENIZE THE LINE
8A80:84 0F    183          STY  $0F
8A82:20 F3 8A 184          JSR  CALCSUM     ;RECOMPUTE THE CHECKSUM
8A85:CD 50 8A 185          CMP  CKSUM       ;HAS PROGRAM BEEN BOMBED?
8A88:D0 32    186          BNE  KILLC       ; YES-KILL PROG
8A8A:20 4D 8B 187          JSR  PGMLINE     ;SYNTAX-CHECK THE INPUT LINE
8A8D:AD 43 8A 188          LDA  SNINS
8A90:49 80    189          EOR  #$80        ;IS INSERT WITH SYNTAX ERROR DISABLED?
8A92:2D 4F 8A 190          AND  SNERR       ;DID SYNTAX ERROR OCCUR
8A95:30 17    191          BMI  ED          ; YES TO BOTH SO DON'T INSERT LINE
8A97:AD 44 8A 192          LDA  MLINS
8A9A:49 80    193          EOR  #$80        ;IS INSERT WITH ML DATA DISABLED?
8A9C:2D 4D 8A 194          AND  MLERR       ;DID ML DATA OCCUR?
8A9F:30 0D    195          BMI  ED          ; YES TO BOTH SO DON'T INSERT LINE
8AA1:AD 45 8A 196          LDA  ULINS
8AA4:49 80    197          EOR  #$80        ;IS INSERT WITH UNDEF'D STMT DISABLED?
8AA6:2D 4E 8A 198          AND  ULERR       ;DID UNDEF'D STMT OCCUR?
8AA9:30 03    199          BMI  ED          ; YES TO BOTH SO DON'T INSERT LINE
8AAB:4C 6C D4 200          JMP  INSERT1     ;INSERT LINE INTO PROGRAM & END
8AAE:4C 3C D4 201 ED       JMP  EDITOR      ;DON'T INSERT LINE, JUST END
8AB1:         202 :
8AB1:C9 0B    203 INCPT.2 CMP  #$0B        ;IS THIS LINE CTRL-K?
8AB3:F0 0A    204          BEQ  KILLK       ; YES-KILL UTIL
8AB5:         205 :
8AB5:AD 48 8A 206 INCPT1  LDA  AREG
```

LISTING 1: TYPE.RIGHT *(continued)*

```
8AB8:38          207          SEC
8AB9:E9 D0       208          SBC   #$D0        ;RESTORE ACCUMULATOR AND STATUS
8ABB:60          209          RTS
8ABC:            210 ;
8ABC:A0 00       211 KILLC    LDY   #KCMSG      ;SELECT "CHECKSUM ERROR" MESSAGE
8ABE:2C          212          DFB   $2C         ;IGNORE NEXT LINE
8ABF:A0 1E       213 KILLK    LDY   #KKMSG      ;SELECT "USER REQUEST" MESSAGE
8AC1:A9 38       214          LDA   #$38
8AC3:85 C5       215          STA   $C5         ;DISCONNECT UTIL'S INTERCEPT
8AC5:A9 E9       216          LDA   #$E9
8AC7:85 C6       217          STA   $C6
8AC9:A9 D0       218          LDA   #$D0
8ACB:85 C7       219          STA   $C7
8ACD:B9 13 8B    220 KILL.1   LDA   KILLMSG,Y  ;GET A CHAR OF THE SELECTED MESSAGE
8AD0:08          221          PHP
8AD1:20 5C DB    222          JSR   COUT        ;DISPLAY "KILL" MESSAGE
8AD4:C8          223          INY               ;ADVANCE TO NEXT CHAR
8AD5:28          224          PLP               ;WAS THIS THE END OF MESSAGE?
8AD6:10 F5       225          BPL   KILL.1      ;  NO, GO AROUND AGAIN
8AD8:A5 73       226          LDA   HIMEML
8ADA:C9 00       227          CMP   #>START     ;IS HIMEM STILL WHERE WE PUT IT?
8ADC:D0 12       228          BNE   KILL.2      ;  NO-LEAVE IT ALONE
8ADE:A5 74       229          LDA   HIMEML+1
8AE0:C9 8A       230          CMP   #<START     ;(CHECK HI BYTE TOO)
8AE2:D0 0C       231          BNE   KILL.2
8AE4:AD 47 8A    232          LDA   HIMSAV+1    ;  YES-RESTORE IT AS
8AE7:AC 46 8A    233          LDY   HIMSAV      ;    BEFORE STARTING UTIL
8AEA:20 F2 E2    234          JSR   TO.REAL     ;CONVERT NEW HIMEM VALUE TO REAL
8AED:20 89 F2    235          JSR   HIMEM       ;INSTALL NEW HIMEM
8AF0:4C 3C D4    236 KILL.2   JMP   EDITOR      ;RETURN TO APPLESOFT
8AF3:            237 ;
8AF3:38          238 CALCSUM SEC
8AF4:A9 51       239          LDA   #>CSUMBGN   ;INITIALIZE ZREG1 AND Y-INDEX
8AF6:E9 70       240          SBC   #>CSUMEND   ;SUCH THAT: ZREG1 + Y
8AF8:A8          241          TAY               ;   = START OF CHECKSUM-PROTECTED AREA
8AF9:A9 8A       242          LDA   #<CSUMBGN   ;AND: ZREG1 LO + 0
8AFB:E9 00       243          SBC   #$00        ;   = END OF AREA, LO BYTE ONLY
8AFD:85 01       244          STA   ZREG1+1
8AFF:A9 70       245          LDA   #>CSUMEND
8B01:85 00       246          STA   ZREG1
8B03:A9 00       247          LDA   #$00        ;CLEAR CHECKSUM RESULT
8B05:51 00       248 CSUM.1   EOR   (ZREG1),Y   ;"SUM IN" ONE BYTE
8B07:C8          249          INY               ;ADVANCE TO NEXT BYTE
8B08:D0 FB       250          BNE   CSUM.1      ;DID WE ADVANCE TO PAGE BOUNDARY?
8B0A:E6 01       251          INC   ZREG1+1     ;  YES-INCR HI BYTE OF ADDRESS
8B0C:A6 01       252          LDX   ZREG1+1
8B0E:E0 95       253          CPX   #<CSUMEND   ;REACH END OF PROGRAM YET?
8B10:90 F3       254          BCC   CSUM.1      ;  NO-KEEP GOING
8B12:60          255          RTS               ;  YES-END WITH CHECKSUM IN A-REG
8B13:            256 ;
8B13:            257 ;   MESSAGE TEXT
8B13:            258 ;
8B13:            259 KILLMSG EQU   *
0000:            260 KCMSG    EQU   *-KILLMSG
8B13:07          261          DFB   $07
8B14:43 48 45    262          DCI               'CHECKSUM ERROR - UTIL REMOVED'
8B17:43 4B 53
8B1A:55 4D 20
8B1D:45 52 52
8B20:4F 52 20
8B23:2D 20 55
8B26:54 49 4C
8B29:20 52 45
8B2C:4D 4F 56
8B2F:45 C4
001E:            263 KKMSG    EQU   *-KILLMSG
8B31:07          264          DFB   $07
8B32:55 53 45    265          DCI               'USER REQUEST - UTIL REMOVED'
8B35:52 20 52
8B38:45 51 55
8B3B:45 53 54
8B3E:20 2D 20
8B41:55 54 49
8B44:4C 20 52
8B47:45 4D 4F
8B4A:56 45 C4
8B4D:            266 ;
                 267 ****************************************
8B4D:            268 ;
8B4D:            269 ;   PROCESS PROGRAM LINES--WHENEVER PROGRAM LINES
8B4D:            270 ;   ARE ENTERED. THIS ROUTINE IS EXECUTED.
8B4D:            271 ;   SYNTAX-CHECK THE TOKENIZED PROGRAM LINE CURRENTLY
8B4D:            272 ;   IN THE INPUT BUFFER.
8B4D:            273 ;
8B4D:20 78 88    274 PGMLINE JSR   SAVSTAT     ;SAVE SOME STATUS REGISTERS
8B50:20 D0 8B    275          JSR   CLRINVF     ;CLEAR DATA AREA
8B53:A9 00       276          LDA   #$00
8B55:85 08       277          STA   ERCOUNT     ;CLEAR ERROR COUNT
8B57:8D 4D 8A    278          STA   MLERR       ;CLEAR ML DATA ERROR FLAG
8B5A:8D 4E 8A    279          STA   ULERR       ;CLEAR UNDEF'D LINE ERROR FLAG
8B5D:8D 4F 8A    280          STA   SNERR       ;CLEAR SYNTAX ERROR FLAG
8B60:20 73 F2    281          JSR   NORMAL      ;SET NORMAL DISPLAY MODE
8B63:            282 ;
8B63:20 3D 8D    283 PGML.3   JSR   COMMAND     ;CHECK ONE COMMAND
8B66:20 B7 00    284          JSR   CHRGOT      ;CHECK BYTE AT END OF COMMAND
8B69:C9 00       285          CMP   #$00        ;WAS IT END OF LINE?
8B6B:D0 F6       286          BNE   PGML.3      ;  NO-CONTINUE TO NEXT COMMAND
8B6D:            287 ;
8B6D:A5 08       288          LDA   ERCOUNT     ;WERE ANY ERRORS FOUND IN THIS LINE?
8B6F:F0 06       289          BEQ   PGML.5      ;  NO-EXIT
8B71:20 F4 8B    290          JSR   BELL        ;  YES-SOUND SPEAKER
8B74:20 40 8C    291          JSR   ERLIST      ;    AND DISPLAY THE ERRORS
8B77:            292 ;
8B77:20 B1 8B    293 PGML.5   JSR   RESTORE     ;RESTORE STATUS REGISTERS
8B7A:60          294          RTS
8B7B:            295 ;
8B7B:            296 ;   SAVE SOME STATUS & DATA TO BE RESTORED
8B7B:            297 ;   BEFORE CONTROL IS RETURNED TO APPLESOFT
8B7B:            298 ;
8B7B:18          299 SAVSTAT CLC
8B7C:A5 69       300          LDA   LOMEM       ;AT LEAST $60 BYTES ARE REQUIRED
8B7E:69 60       301          ADC   #$60        ;BETWEEN LOMEM AND HIMEM
8B80:A8          302          TAY
8B81:A5 6A       303          LDA   LOMEM+1
8B83:69 00       304          ADC   #$00        ;IS THERE ENOUGH MEMORY
8B85:C5 74       305          CMP   HIMEML+1    ;  TO DO THE SYNTAX CHECK?
8B87:90 06       306          BCC   SAV.0       ;  YES-CONTINUE
8B89:D0 23       307          BNE   MEMERR      ;  NO-GIVE ERROR MESSAGE
8B8B:C4 73       308          CPY   HIMEML
8B8D:B0 1F       309          BCS   MEMERR
8B8F:A0 1F       310 SAV.0    LDY   #$1F
8B91:99 00 00    311 SAV.1    LDA   $00,Y       ;MOVE $20 BYTES OF ZERO PAGE
8B94:91 69       312          STA   (LOMEM),Y   ;  TO FREE RAM FOR STORAGE
8B96:88          313          DEY               ;ADVANCE TO NEXT BYTE
8B97:10 F8       314          BPL   SAV.1       ;CONTINUE TILL ALL $20 BYTES ARE MOVE
8B99:A5 B8       315          LDA   CHRPTR
8B9B:8D 49 8A    316          STA   SAVCPTR     ;SAVE CURRENT VALUE OF
8B9E:A5 B9       317          LDA   CHRPTR+1    ;PROGRAM COUNTER
8BA0:8D 4A 8A    318          STA   SAVCPTR+1
8BA3:A5 32       319          LDA   INVBYTE
8BA5:8D 4B 8A    320          STA   SAVINVB     ;SAVE CURRENT VIDEO DISPLAY MODE
8BA8:A5 F3       321          LDA   FLASHBT     ;(NORMAL, INVERSE, FLASH)
8BAA:8D 4C 8A    322          STA   SAVFBYT
8BAD:60          323          RTS               ;INITIALIZATION IS COMPLETE SO EXIT
8BAE:4C 10 D4    324 MEMERR   JMP   OUTMEM      ;OUT OF MEMORY ERROR
8BB1:            325 ;
8BB1:            326 ;   RESTORE STATUS & DATA FOR APPLESOFT'S USE.
8BB1:            327 ;
8BB1:AD 49 8A    328 RESTORE LDA   SAVCPTR
8BB4:85 B8       329          STA   CHRPTR      ;RESTORE PROGRAM COUNTER
8BB6:AD 4A 8A    330          LDA   SAVCPTR+1
8BB9:85 B9       331          STA   CHRPTR+1
8BBB:AD 4B 8A    332          LDA   SAVINVB
8BBE:85 32       333          STA   INVBYTE     ;RESTORE VIDEO DISPLAY MODE
8BC0:AD 4C 8A    334          LDA   SAVFBYT
8BC3:85 F3       335          STA   FLASHBT
8BC5:A0 1F       336          LDY   #$1F
8BC7:B1 69       337 RST.1    LDA   (LOMEM),Y
8BC9:99 00 00    338          STA   $00,Y       ;RESTORE $20 BYTES OF ZERO PAGE
8BCC:88          339          DEY
8BCD:10 F8       340          BPL   RST.1
8BCF:60          341          RTS
8BD0:            342 ;
8BD0:            343 ;   CLEAR THE DATA AREA FOR STORAGE OF FLAGS
8BD0:            344 ;   INDICATING WHICH PARTS OF THE PROGRAM LINE
8BD0:            345 ;   ARE TO BE LISTED IN INVERSE.
8BD0:            346 ;
8BD0:18          347 CLRINVF CLC
8BD1:A5 69       348          LDA   LOMEM
8BD3:69 20       349          ADC   #$20
8BD5:85 04       350          STA   INVFLAG     ;SET UP POINTER FOR "INVERSE"
8BD7:A5 6A       351          LDA   LOMEM+1     ;DATA AREA (AFTER LEAVING $20 BYTES
8BD9:69 00       352          ADC   #$00        ;FOR STORAGE OF Z.P. BYTES $00-1F)
8BDB:85 05       353          STA   INVFLAG+1
8BDD:18          354          CLC
8BDE:A5 04       355          LDA   INVFLAG
8BE0:69 20       356          ADC   #$20
8BE2:85 06       357          STA   SPCFLAG     ;SET UP PTR FOR NO HIGHLIGHT
8BE4:A5 05       358          LDA   INVFLAG+1   ;ITEM" DATA AREA, $20 BYTES AFTER
8BE6:69 00       359          ADC   #$00        ;START OF "INVERSE" AREA
8BE8:85 07       360          STA   SPCFLAG+1
8BEA:A0 3F       361          LDY   #$3F
8BEC:A9 00       362          LDA   #$00
8BEE:91 04       363 CLF.1    STA   (INVFLAG),Y ;NOW CLEAR THE "INVERSE" AND
8BF0:88          364          DEY               ;"HIGHLIGHT MISSING ITEM" AREAS
8BF1:10 FB       365          BPL   CLF.1
8BF3:60          366          RTS
8BF4:            367 ;
8BF4:            368 ;   SOUND THE SPEAKER, USING THE TONES
8BF4:            369 ;   INDICATING THE HIGHEST PRIORITY
8BF4:            370 ;   ERROR THAT OCCURRED.  PRIORITIES ARE:
8BF4:            371 ;     1 - SYNTAX & TYPE MISMATCH
8BF4:            372 ;     2 - ML DATA
8BF4:            373 ;     3 - UNDEF'D LINE NUMBER
8BF4:            374 ;
8BF4:AD 4F 8A    375 BELL     LDA   SNERR       ;DID SYNTAX ERROR OCCUR?
8BF7:2D 3D 8A    376          AND   SNBELL      ;IS BELL ON SYNTAX ERROR ENABLED?
8BFA:10 06       377          BPL   BELL.1      ;  NO
8BFC:20 32 8C    378          JSR   BELL2       ;  YES TO BOTH, SO SOUND
8BFF:4C 24 8C    379          JMP   BELL1       ;    PRIORITY #1 BELL: HI-LO
8C02:AD 4D 8A    380 BELL.1   LDA   MLERR       ;DID ML DATA ERROR OCCUR?
8C05:2D 3E 8A    381          AND   MLBELL      ;IS BELL ON ML DATA ENABLED?
8C08:10 09       382          BPL   BELL.2      ;  NO
8C0A:20 32 8C    383          JSR   BELL2       ;  YES TO BOTH, SO SOUND
8C0D:20 24 8C    384          JSR   BELL1       ;    PRIORITY #2 BELL: HI-LO-HI
8C10:4C 32 8C    385          JMP   BELL2
8C13:AD 4F 8A    386 BELL.2   LDA   ULERR       ;DID UNDEF'D LINE OCCUR?
8C16:2D 3F 8A    387          AND   ULBELL      ;IS BELL ON UNDEF'D LINE ENABLED?
8C19:10 16       388          BPL   RTS1        ;  NO
8C1B:20 24 8C    389          JSR   BELL1       ;  YES TO BOTH, SO SOUND
8C1E:20 32 8C    390          JSR   BELL2       ;    PRIORITY #3 BELL: LO-HI-LO
8C21:4C 24 8C    391          JMP   BELL1
8C24:            392 ;
8C24:A0 80       393 BELL1    LDY   #$80        ;TOGGLE SPEAKER $80 (128) TIMES
8C26:A9 0E       394 BELL1.1  LDA   #$0E         ;$0E = 728 Hz = "LO" PITCH
8C28:20 A8 FC    395          JSR   WAIT        ;DELAY BEFORE EACH TOGGLE
8C2B:2C 30 C0    396          BIT   SPKR        ;TOGGLE SPEAKER
8C2E:88          397          DEY               ;DONE IT $80 TIMES YET?
8C2F:D0 F5       398          BNE   BELL1.1     ;  NO, DO IT AGAIN
8C31:60          399 RTS1     RTS               ;  YES-EXIT
8C32:            400 ;
8C32:A0 80       401 BELL2    LDY   #$80        ;TOGGLE SPEAKER $80 (128) TIMES
8C34:A9 0C       402 BELL2.1  LDA   #$0C         ;$0C = 937 Hz = "HI" PITCH
8C36:20 A8 FC    403          JSR   WAIT        ;DELAY BEFORE EACH TOGGLE
8C39:2C 30 C0    404          BIT   SPKR        ;TOGGLE SPEAKER
8C3C:88          405          DEY               ;DONE IT $80 TIMES YET?
8C3D:D0 F5       406          BNE   BELL2.1     ;  NO, DO IT AGAIN
8C3F:60          407          RTS               ;  YES-EXIT
8C40:            408 ;
8C40:            409 ;   LIST THE ENTERED LINE, SHOWING ERRORS IN INVERSE
8C40:            410 ;
8C40:AD 4F 8A    411 ERLIST   LDA   SNERR       ;DID SYNTAX ERROR OCCUR?
8C43:2D 40 8A    412          AND   SNLIST      ;IS LIST ON SYNTAX ERROR ENABLED?
8C46:30 11       413          BMI   ERLIST1     ;  YES TO BOTH SO LIST LINE
8C48:AD 4D 8A    414          LDA   MLERR       ;DID ML DATA ERROR OCCUR?
8C4B:2D 41 8A    415          AND   MLLIST      ;IS LIST ON ML DATA ENABLED?
8C4E:30 09       416          BMI   ERLIST1     ;  YES TO BOTH SO LIST LINE
```

```
8C50 AD 4E BA   417           LDA  ULERR       ;DID UNDEF'D LINE OCCUR?
8C53 2D 42 BA   418           AND  ULLIST      ; IS LIST ON UNDEF'D LINE ENABLED?
8C56 30 01      419           BMI  ERLIST1     ; YES TO BOTH SO LIST LINE
8C58 60         420           RTS              ; NO EXIT
8C59:20 8B FD   421 ERLIST1 JSR  CROUT       ;OUTPUT A CARIAGE RETURN
8C5C:20 73 F2   422           JSR  NORMAL      ; SET NORMAL VIDEO MODE
8C5F A5 51      423           LDA  LINENUM+1
8C61 A6 50      424           LDX  LINENUM
8C63:20 24 ED   425           JSR  PRTAX       ;PRINT LINE NUMBER
8C66:20 57 DB   426           JSR  SPACOUT     ;PRINT A SPACE
8C69:           427 ;
8C69:A9 01      428           LDA  #$01
8C6B:A2 FF      429           LDX  #$FF
8C6D:85 B9      430           STA  CHRPTR+1    ;POINT TO START OF INPUT BUFFER
8C6F 86 B8      431           STX  CHRPTR      ;AT $0200 (LESS 1 = $01FF)
8C71            432
8C71:20 B1 00   433 ERLST.1 JSR  CHRGET      ;GET A CHAR FROM INPUT BUFFER
8C74:20 37 8D   434           JSR  GETSPC      ;HIGHLIGHT SPACE BEFORE CHAR?
8C77 F0 16      435           BEQ  ERLST.3     ; NO. GO RIGHT TO PRINTING CHAR
8C79 24 32      436           BIT  INVBYTE     ; YES; WAS INVERSE ALREADY SET?
8C7B 30 09      437           BMI  ERLST.2     ; NO. JUST PRINT INVERSE SPACE
8C7D:20 73 F2   438           JSR  NORMAL      ; YES, PRINT A NORMAL SPACE FIRST
8C80:20 B5 8C   439           JSR  CRON40      ;NEXT LINE IF NO ROOM FOR A SPACE
8C83:20 57 DB   440           JSR  SPACOUT     ;PRINT A NORMAL SPACE
8C86:20 77 F2   441 ERLST.2 JSR  INVERSE     ;SET INVERSE MODE
8C89:20 B5 8C   442           JSR  CRON40      ;NEXT LINE IF NO ROOM FOR A SPACE
8C8C:20 57 DB   443           JSR  SPACOUT     ;PRINT INVERSE SPACE
8C8F:20 73 F2   444 ERLST.3 JSR  NORMAL      ;BACK TO NORMAL MODE
8C92:20 31 8D   445           JSR  GETINV      ;HIGHLIGHT THIS BYTE?
8C95:F0 03      446           BEQ  ERLST.4     ; NO-SKIP
8C97:20 77 F2   447           JSR  INVERSE     ; YES-SET INVERSE MODE
8C9A:20 B7 00   448 ERLST.4 JSR  CHRGOT      ;WHAT CHAR WERE WE PRINTING AGAIN?
8C9D C9 00      449           CMP  #$00        ;WAS IT END OF LINE?
8C9F F0 06      450           BEQ  ERLST.5     ; YES-END
8CA1:20 EC 8C   451           JSR  LIST1       ; NO-LIST THIS BYTE
8CA4 4C 71 8C   452           JMP  ERLST.1     ;  AND ADVANCE TO NEXT BYTE
8CA7            453
8CA7:20 73 F2   454 ERLST.5 JSR  NORMAL      ;GO BACK TO NORMAL DISPLAY MODE
8CAA 4C 88 FD   455           JMP  CROUT       ;CARRIAGE RETURN & EXIT
8CAD            456
8CAD            457 ;   ARE THERE ENOUGH SPACES REMAINING ON THIS LINE
8CAD            458 ;   TO PRINT WHAT WE NEED TO?  IF NOT, GO TO NEXT LINE
8CAD            459
8CAD:20 D7 8C   460 CRON34  JSR  SPTOGO      ;HOW MANY SPACES TO EDGE OF SCREEN?
8CB0:C0 07      461           CPY  #7          ;ENOUGH FOR THE LONGEST KEYWORD?
8CB2:90 09      462           BCC  CRON.1      ; NO-NEXT LINE
8CB4 60         463           RTS              ; YES-DO NOTHING
8CB5            464
8CB5:20 D7 8C   465 CRON40  JSR  SPTOGO      ;HOW MANY SPACES TO EDGE OF SCREEN?
8CB8:C0 00      466           CPY  #0          ;ANY AT ALL?
8CBA:F0 01      467           BEQ  CRON.1      ; NO-NEXT LINE
8CBC 60         468           RTS              ; YES-DO NOTHING
8CBD            469
8CBD            470 ;   CARRIAGE RETURN AND INIT NEXT LINE
8CBD            471 ;
8CBD:48         472 CRON.1  PHA              ;PRESERVE ACCUMULATOR
8CBE:20 88 FD   473           JSR  CROUT       ;CARRIAGE RETURN TO NEXT LINE
8CC1:A5 32      474           LDA  INVBYTE
8CC3:48         475           PHA              ;SAVE CURRENT DISPLAY MODE
8CC4:A5 F3      476           LDA  FLASHBT     ;(NORMAL, INVERSE, FLASH)
8CC6:48         477           PHA
8CC7:20 73 F2   478           JSR  NORMAL      ;SET NORMAL MODE
8CCA:A2 05      479           LDX  #$05
8CCC:20 4A F9   480           JSR  PRBL2       ;START NEXT LINE WITH 5 SPACES
8CCF:68         481           PLA
8CD0:85 F3      482           STA  FLASHBT     ;RESTORE DISPLAY MODE
8CD2:68         483           PLA
8CD3:85 32      484           STA  INVBYTE
8CD5:68         485           PLA              ;RESTORE ACCUMULATOR
8CD6:60         486           RTS              ;AND EXIT
8CD7            487 ;
8CD7            488 ;   HOW MANY SPACES BEFORE RIGHT EDGE OF DISPLAY IS REACHED?
8CD7            489 ;
8CD7:48         490 SPTOGO  PHA              ;PRESERVE ACCUMULATOR
8CD8:38         491           SEC
8CD9:A4 24      492           LDY  CH          ;IS 80-COLUMN MODE ACTIVE?
8CDB:F0 07      493           BEQ  SPTG.1      ; YES, GO TO 80-COL ROUTINE
8CDD:A9 27      494           LDA  #40-1       ;RIGHT EDGE OF 40-COL SCREEN
8CDF:E5 24      495           SBC  CH          ; MINUS CURSOR POSITION
8CE1:A8         496           TAY              ; EQUALS SPACES TO GO
8CE2:68         497           PLA              ;RESTORE ACCUMULATOR
8CE3:60         498           RTS              ;AND EXIT WITH ANSWER IN Y-REG
8CE4:A9 4F      499 SPTG.1  LDA  #80-1       ;RIGHT EDGE OF 80-COL SCREEN
8CE6:ED 7B 05   500           SBC  CH80        ; MINUS CURSOR POSITION
8CE9:A8         501           TAY              ; EQUALS SPACES TO GO
8CEA:68         502           PLA              ;RESTORE ACCUMULATOR
8CEB:60         503           RTS              ;AND EXIT WITH ANSWER IN Y-REG
8CEC            504 ;
8CEC            505 ;   LIST ONE BYTE OF THE PROGRAM LINE
8CEC            506 ;   (BYTE IS IN ACCUMULATOR)
8CEC            507 ;
8CEC:20 B5 8C   508 LIST1   JSR  CRON40
8CEF:C9 80      509           CMP  #$80        ;ASCII OR TOKEN?
8CF1:90 12      510           BCC  LIST2       ; ASCII SO GO PRINT IT
8CF3:48         511           PHA              ; TOKEN
8CF4:20 57 DB   512           JSR  SPACOUT     ;SPACE PRECEDES KEYWORD
8CF7:20 AD 8C   513           JSR  CRON34      ;GO TO NEXT LINE IF THIS LINE DOESN'T
8CFA:68         514           PLA              ; HAVE ENOUGH ROOM FOR A KEYWORD
8CFB:20 05 8D   515           JSR  LIST2       ;PRINT KEYWORD
8CFE:20 B5 8C   516           JSR  CRON40      ;RETURN IF AT END OF SCREEN
8D01:20 57 DB   517           JSR  SPACOUT     ;SPACE FOLLOWS KEYWORD
8D04 60         518           RTS              ;AND EXIT
8D05            519 ;
8D05:C9 80      520 LIST2   CMP  #$80        ;IS THIS BYTE ASCII OR TOKEN?
8D07:B0 03      521           BCS  LIST2.1
8D09:4C 5C DB   522           JMP  COUT        ; ASCII, SO JUST PRINT IT
8D0C:AA         523 LIST2.1 TAX              ; TOKEN, SO FIND IT IN TABLE
8D0D:A9 D0      524           LDA  #$D0
8D0F:85 00      525           STA  ZREG1       ;SET UP START OF TABLE
8D11:85 01      526           STA  ZREG1+1     ; ADDRESS $D0D0
8D13:A0 00      527           LDY  #$00
8D15            528 ;
8D15            529 ;   FIND A KEYWORD IN KEYWORD TABLE AND PRINT IT
8D15            530 ;   (INDEX NUMBER OF DESIRED KEYWORD IS IN X-REG)
8D15            531
8D15:CA         532 LIST2.2 DEX              ;FOUND KEYWORD YET?
8D16:10 0E      533           BPL  LIST2.5     ; YES, GO PRINT IT
8D18:B1 00      534 LIST2.3 LDA  (ZREG1),Y   ; NO  KEEP LOOKING
8D1A 08         535           PHP
8D1B:E6 00      536           INC  ZREG1       ;POINT TO NEXT CHAR
8D1D:D0 02      537           BNE  LIST2.4     ; IN KEYWORD LIST
8D1F:E6 01      538           INC  ZREG1+1     ;HI BYTE TOO
8D21 28         539 LIST2.4 PLP              ;IS THIS THE LAST CHAR OF A KEYWORD
8D22:10 F4      540           BPL  LIST2.3     ; NO-LOOK AT NEXT CHAR
8D24:30 EF      541           BMI  LIST2.2     ; YES-GET NEXT KEYWORD
8D26            542
8D26:B1 00      543 LIST2.5 LDA  (ZREG1),Y   ;GET A CHAR OF KEYWORD
8D28 08         544           PHP
8D29:20 5C DB   545           JSR  COUT        ;PRINT IT
8D2C C8         546           INY              ;ADVANCE TO NEXT CHAR
8D2D 28         547           PLP              ;END OF KEYWORD?
8D2E:10 F6      548           BPL  LIST2.5     ; NO-KEEP PRINTING
8D30 60         549           RTS              ; YES, WE'RE DONE
8D31            550 ;
8D31            551 ;   GET THE STATUS OF THE INVERSE FLAG FOR THE
8D31            552 ;   CURRENT BYTE AND SPACE PRECEDING CURRENT BYTE
8D31            553 ;   Z-FLAG IS CLEAR IF INVERSE, SET IF NORMAL
8D31            554 ;
8D31:20 AA 92   555 GETINV  JSR  FLAGADR     ;FIND 'INVERSE' DATA BIT IN TABLE
8D34:31 04      556           AND  (INVFLAG),Y ; SET Z-FLAG APPROPRIATELY
8D36 60         557           RTS
8D37:20 AA 92   558 GETSPC  JSR  FLAGADR     ;FIND "INVERSE SPACE" DATA BIT
8D3A:31 06      559           AND  (SPCFLAG),Y ; SET Z-FLAG APPROPRIATELY
8D3C 60         560 RTS2    RTS
8D3D            561 ;
                562 ;.....................................
8D3D            563 ;
8D3D            564 ;   SCAN ONE COMMAND FOR ERRORS.  SEND ERROR MESSAGES TO
8D3D            565 ;   THE SCREEN, AND SET UP FLAGS SHOWING WHICH PARTS OF THE
8D3D            566 ;   LISTED LINE CONTAIN ERRORS AND SHOULD BE DISPLAYED
8D3D            567 ;   IN INVERSE
8D3D            568 ;
8D3D:BA         569 COMMAND TSX
8D3E:86 F8      570           STX  STACK       ;SAVE STACK POINTER IN CASE OF ERRO
8D40:20 B1 00   571           JSR  CHRGET      ;GET FIRST CHAR OF COMMAND
8D43:F0 F7      572           BEQ  RTS2        ;BLANK COMMAND--DO NOTHING
8D45:C9 C0      573           CMP  #$C0        ;A LEGAL KEYWORD TO BEGIN COMMAND?
8D47:90 08      574           BCC  CMD.1       ; YES-CONTINUE
8D49:20 83 92   575           JSR  SETINV      ; NO HIGHLIGHT ERROR
8D4C:A2 07      576           LDX  #KWBEGIN    ;SELECT ERROR MESSAGE
8D4E:4C DC 92   577           JMP  ERROR       ;AND PRINT IT
8D51:AB         578 CMD.1   TAY              ;SAVE THIS COMMAND
8D52:A9 8F      579           LDA  #<EOC CK-1  ;PUSH A SUBROUTINE ADDRESS ON STACK
8D54:48         580           PHA              ;SO ONCE SYNTAX CHECK IS COMPLETE
8D55:A9 75      581           LDA  #>EOC CK-1  ;END OF COMMAND CHECK WILL FOLLOW
8D57:48         582           PHA
8D58:98         583           TYA              ;IS THIS COMMAND ASCII OR TOKEN?
8D59:10 0D      584           BPL  LET1        ; ASCII, SO IT MUST BE A 'LET' COMM
8D5B:0A         585           ASL  A           ; TOKEN, SO LET'S CHECK IT OUT
8D5C:AB         586           TAY
8D5D:B9 BA 93   587           LDA  INDEX+1,Y   ;LOAD ADDRESS FROM TABLE AND
8D60:48         588           PHA              ; SET UP A 'RETURN' TO THE CHECK
8D61:B9 B9 93   589           LDA  INDEX,Y     ; FOR THE RETRIEVED TOKEN
8D64:48         590           PHA
8D65:4C B1 00   591           JMP  CHRGET      ;"JUMP" INTO THAT CHECK
8D68            592
8D68:20 7D E0   593 LET1    JSR  ABC.CK      ;A LETTER?
8D6B:B0 08      594           BCS  LET1.1      ; YES,CHECK IT AS A 'LET' COMMAND
8D6D:20 A0 92   595           JSR  SETSPC      ; NO HIGHLIGHT ERROR
8D70:A2 0B      596           LDX  #KWVEXP     ;SELECT 'KEYWORD OR VARIABLE EXPECT
8D72:4C DC 92   597           JMP  ERROR       ;AND PRINT ERROR MESSAGE
8D75:4C 8A BE   598 LET1.1  JMP  LET
8D78            599 ;
                600 ;.....................................
8D78            601 ;
8D78            602 ;   KEYWORD CHECKS--EACH ROUTINE CHECKS A COMMAND
8D78            603 ;   BEGINNING WITH THAT KEYWORD, GENERATING ERROR
8D78            604 ;   MESSAGES TO THE SCREEN, AND SETTING UP INVERSE FLAGS
8D78            605 ;   TO LIST THE LINE WITH ERRORS HIGHLIGHTED
8D78            606 ;
8D78            607         MSB  OFF
8D78:20 52 92   608 FOR     JSR  VARNAME     ;REAL VARIABLE MUST FOLLOW 'FOR'
8D7B:A9 D0      609           LDA  #$D0        ;THEN BY KEYWORD '='
8D7D:20 86 BF   610           JSR  SYMBLCK
8D80:20 4E 90   611           JSR  EVALNUM     ;THEN A NUMERIC EXPRESSION
8D83:A9 C1      612           LDA  #$C1        ;THEN KEYWORD 'TO'
8D85:20 86 BF   613           JSR  SYMBLCK
8D88:20 4E 90   614           JSR  EVALNUM     ;ANOTHER NUMERIC EXPRESSION
8D8B:A9 C7      615           LDA  #$C7        ;KEYWORD 'STEP' IS OPTIONAL
8D8D:20 94 BF   616           JSR  SYMBEOC     ;IS 'STEP' THERE?
8D90:F0 17      617           BEQ  RTS3        ; NO-MUST BE END OF COMMAND
8D92:4C 4E 90   618           JMP  EVALNUM     ;IF IT IS, SCAN STEP EXPRESSION
8D95            619
8D95:F0 12      620 NEXT    BEQ  RTS3        ;'NEXT' BY ITSELF IS O K
8D97:20 7D E0   621           JSR  ABC.CK      ;OTHERWISE, DOES VARIABLE FOLLOW?
8D9A:B0 05      622           BCS  NEXT.1      ; YES-CONTINUE
8D9C:A9 10      623           LDA  #TXVAR      ; NO-SELECT 'VARIABLE EXPECTED'
8D9E:4C C2 92   624           JMP  TXPEOC      ;PRINT ERROR MESSAGE
8DA1:20 52 92   625 NEXT.1  JSR  VARNAME     ;SCAN THE REAL VARIABLE NAME
8DA4:20 92 BF   626           JSR  COMAEOC     ;IS A COMMA NEXT?
8DA7:D0 F8      627           BNE  NEXT.1      ; YES-SCAN NEXT VARIABLE
8DA9:60         628 RTS3    RTS              ; NO-EXIT
8DAA            629
8DAA:20 B1 00   630 DATA.1  JSR  CHRGET      ;ADVANCE TO CHAR AFTER COMMA
8DAD:F0 FA      631 DATA    BEQ  RTS3        ;"DATA" BY ITSELF IS O K
8DAF:C9 2C      632           CMP  #','        ;OTHERWISE, IS COMMA NEXT?
8DB1:F0 F7      633           BEQ  DATA.1      ; YES  KEEP GOING
8DB3:C9 22      634           CMP  #'"'        ; NO  IS IT A QUOTE?
8DB5:D0 09      635           BNE  DATA.2      ; NO-CONTINUE
8DB7:20 8D 91   636           JSR  TO.EOQ      ; YES-ADVANCE TO END OF QUOTATION
8DBA:20 92 BF   637           JSR  COMAEOC     ;COMMA OR END OF COMMAND MUST FOLLO
8DBD:4C A0 8D   638           JMP  DATA        ;THEN CONTINUE SCANNING
8DC0:20 B1 00   639 DATA.2  JSR  CHRGET      ;SCAN NEXT CHAR
8DC3:C9 2C      640 DATA.3  CMP  #','        ;A COMMA?
8DC5:F0 E3      641           BEQ  DATA.1      ; YES-CONTINUE
8DC7:C9 22      642           CMP  #'"'        ; NO, A QUOTE?
```

```
8DC9 D0 F5      643          BNE  DATA 2          NO-CONTINUE
8DCB 20 8D 91   644          JSR  TO EOQ         YES ADVANCE TO END OF QUOTE
8DCE D0 F3      645          BNE  DATA 3         IF MORE DATA THEN CONTINUE
8DD0 60         646          RTS                 ; TILL DONE, THEN EXIT
8DD1            647 ;
8DD1:C9 22      648 INPUT    CMP  #'"            IS THERE A QUOTED PROMPT?
8DD3 D0 0F      649          BNE  GET            NO, GO TO "INPUT" PART
8DD5 20 8D 91   650          JSR  TO EOQ         ; YES, SCAN PROMPT
8DD8 D0 05      651          BNE  INPUT 1        GO TO "INPUT" PART
8DDA A9 22      652          LDA  #'"            UNLESS CLOSING QUOTE IS MISSING
8DDC 4C CD 92   653          JMP  TXPSET1        IF IT IS, PRINT ERROR MESSAGE
8DDF A9 3B      654 INPUT 1  LDA  #';
8DE1:20 86 8F   655          JSR  SYMBLCK        SEMICOLON MUST FOLLOW QUOTE
8DE4            656
8DE4 20 28 92   657 GET      JSR  VAR            VARIABLE IS NEXT, FOLLOWED BY COMMA
8DE7 20 92 8F   658          JSR  COMAEOC        (YES, "GET" WORKS W/ LIST OF VAR'S)
8DEA:D0 F8      659          BNE  GET            OR END OF COMMAND
8DEC:60         660          RTS
8DED            661
8DED:20 AD 8F   662 DEL      JSR  GETLNUM        STARTING LINE NO FOLLOWS "DEL"
8DF0:20 84 8F   663          JSR  COMMA          THEN COMMA
8DF3:4C AD 8F   664          JMP  GETLNUM        AND ENDING LINE NO
8DF6            665
8DF6 20 52 92   666 DIM      JSR  VARNAME        VARIABLE MUST FOLLOW "DIM"
8DF9 20 6F 92   667          JSR  VARTYPE        STRING OR INT VARIABLE O.K. TOO
8DFC A9 28      668          LDA  #'(
8DFE 20 86 8F   669          JSR  SYMBLCK        LEFT PARENTHESIS IS NEXT
8E01 20 4E 90   670 DIM 1    JSR  EVALNUM        FOLLOWED BY A DIMENSION
8E04 C9 2C      671          CMP  #','           IS COMMA NEXT?
8E06 D0 06      672          BNE  DIM 2          NO-SKIP
8E08 20 B1 00   673          JSR  CHRGET         YES-ADVANCE PAST COMMA
8E0B 4C 01 8E   674          JMP  DIM 1          ; AND SCAN NEXT DIMENSION
8E0E C9 29      675 DIM 2    CMP  #')            IS RIGHT PARENTHESIS NEXT?
8E10 F0 09      676          BEQ  DIM 3          ; YES-CONTINUE
8E12 A9 2C      677          LDA  #','
8E14 85 08      678          STA  ERRSYM1        ; NO ERROR MESSAGE SAYS
8E16 A9 29      679          LDA  #')            ; COMMA OR ")" MUST FOLLOW
8E18 4C D1 92   680          JMP  TXPSET2        PRINT ERROR MESSAGE
8E1B 20 B1 00   681 DIM 3    JSR  CHRGET         ADVANCE PAST FINAL PARENTHESIS
8E1E 20 92 8F   682          JSR  COMAEOC        DOES COMMA FOLLOW?
8E21 D0 D3      683          BNE  DIM            ; YES-ADVANCE TO NEXT DIM'D VAR
8E23 60         684          RTS                 ; NO-EXIT
8E24            685
8E24 20 28 92   686 READ     JSR  VAR            VARIABLE MUST FOLLOW "READ"
8E27 20 92 8F   687          JSR  COMAEOC        THEN OPTIONAL COMMA
8E2A D0 F8      688          BNE  READ           AND MORE VARIABLES
8E2C 60         689          RTS                 FINALLY, END OF COMMAND
8E2D            690
8E2D 20 4E 90   691 CALL     JSR  EVALNUM        SCAN ADDRESS BEING CALLED
8E30 F0 34      692 AMPER    BEQ  RTS4           EXIT IF END OF COMMAND
8E32 A5 B8      693          LDA  CHRPTR         ;
8E34 48         694          PHA                 ;OTHERWISE MARK START
8E35 20 A3 93   695          JSR  TO EOC         ;AND END OF DATA BEING
8E38 68         696          PLA                 ;SENT TO MACH. LANG. ROUTINE
8E39 20 8D 92   697          JSR  SETINVZ        ;HIGHLIGHT THIS M.L. DATA
8E3C A2 08      698          LDX  #WLDATA        ;SELECT "M.L. DATA IGNORED" MESSAGE
8E3E 4C 0C 92   699          JMP  ERROR          ;PRINT "ERROR" MESSAGE
8E41            700
8E41            701 PLOT     EQU  *
8E41            702 POKE     EQU  *
8E41 20 4F 90   703          JSR  EVALNUM        NUMERIC EXPRESSION MUST BE NEXT
8E44 20 84 8F   704          JSR  COMMA          FOLLOWED BY COMMA
8E47 4C 4F 90   705          JMP  EVALNUM        AND A SECOND NUMERIC EXPRESSION
8E4A            706
8E4A            707 HLIN     EQU  *
8E4A            708 VLIN     EQU  *
8E4A 20 41 8E   709          JSR  PLOT           SCAN PLOT RANGE FOLLOWING COMMAND
8E4D A9 C5      710          LDA  #197           KEYWORD "AT" MUST FOLLOW
8E4F 20 86 8F   711          JSR  SYMBLCK
8E52 4C 4E 90   712          JMP  EVALNUM        FINALLY, THE "AT" COORDINATE
8E55            713 ;
8E55:C9 C1      714 HPLOT    CMP  #193           DOES KEYWORD "TO" FOLLOW "HPLOT"?
8E57 D0 03      715          BNE  HPLOT 1        ; NO SKIP
8E59 20 B1 00   716          JSR  CHRGET         ; YES-ADVANCE TO COORDINATES
8E5C 20 41 8E   717 HPLOT 1  JSR  PLOT           SCAN COORDINATES
8E5F A9 C1      718          LDA  #193           IS KEYWORD "TO" NEXT?
8E61 20 94 8F   719          JSR  SYMBEOC
8E64 D0 F6      720          BNE  HPLOT 1        ; YES-THERE ARE MORE COORDINATES
8E66 60         721 RTS4     RTS                 ; NO-EXIT
8E67            722 ;
8E67            723 DRAW     EQU  *
8E67            724 XDRAW    EQU  *
8E67 20 4E 90   725          JSR  EVALNUM        SCAN INDEX OF ITEM TO DRAW
8E6A A9 C5      726          LDA  #197           KEYWORD "AT" IS OPTIONAL
8E6C 20 94 8F   727          JSR  SYMBEOC
8E6F F0 F5      728          BEQ  RTS4           IF NOT USED, THEN END OF COMMAND
8E71 4C 41 8E   729          JMP  PLOT           BUT IF SO, SCAN COORDINATES
8E74            730 ;
8E74 A9 AB      731 ONERR    LDA  #171           KEYWORD "GOTO" MUST FOLLOW "ONERR"
8E76 20 86 8F   732          JSR  SYMBLCK
8E79 4C 0A 90   733          JMP  LNUM CK        FOLLOWED BY LINE NUMBER
8E7C            734 ;
8E7C            735 RECALL   EQU  *
8E7C            736 STORE    EQU  *
8E7C A5 B8      737          LDA  CHRPTR         SAVE PC OF START OF ARRAY NAME
8E7E 48         738          PHA                 ; IN CASE OF ERROR
8E7F 20 52 92   739          JSR  VARNAME        SCAN ARRAY NAME TO RECALL/STORE
8E82 20 6F 92   740          JSR  VARTYPE        INTEGER ARRAYS O.K. TOO
8E85 68         741          PLA
8E86 AA         742          TAX                 PUT START OF ARRAY NAME IN X-REG AND
8E87 4C 6A 90   743          JMP  NUM CK         MAKE SURE ARRAY WAS NOT A STRING
8E8A            744 ;
8E8A 20 28 92   745 LET      JSR  VAR            SCAN VARIABLE TO BE ASSIGNED
8E8D A9 D0      746          LDA  #208           KEYWORD "=" MUST FOLLOW
8E8F 20 86 8F   747          JSR  SYMBLCK
8E92 A5 11      748          LDA  TYP STR        GET VARIABLE TYPE
8E94 F0 03      749          BEQ  LET 1          WHAT TYPE WAS IT?
8E96 4C 5F 90   750          JMP  EVALSTR        ; STRING SO SCAN STRING EXPRESSION
8E99 4C 4E 90   751 LET 1    JMP  EVALNUM        ; NUMERIC SO SCAN NUMERIC EXPRESSION
8E9C            752 ;
8E9C F0 3B      753 RUN      BEQ  RTS5           "RUN" BY ITSELF IS O.K.
8E9E B0 03      754          BCS  RUN 1          OTHERWISE, MUST BE FOLLOWED BY DIGITS
8EA0 4C 0A 90   755          JMP  LNUM CK        SCAN LINE NUMBER TO "RUN"
8EA3 A9 01      756 RUN 1    LDA  #TXLNUM        ERROR MESSAGE IF OTHER THAN
8EA5 4C C2 92   757          JMP  TXPEOC         ; LINE NUMBER FOLLOWS "RUN"
8EA8            758 ;
8EA8 20 4E 90   759 IF       JSR  EVALNUM        SCAN BOOLEAN EXPRESSION AFTER "IF"
8EAB C9 AB      760          CMP  #171           IS KEYWORD "GOTO" NEXT?
8EAD D0 06      761          BNE  IF 2           ; NO-CONTINUE
8EAF 20 B1 00   762          JSR  CHRGET         ; YES ADVANCE PAST "GOTO"
8EB2 4C 0A 90   763 IF 1     JMP  LNUM CK        ; AND SCAN LINE NUMBER
8EB5:C9 C4      764 IF 2     CMP  #196           IF NOT "GOTO" IS IT "THEN"?
8EB7 F0 09      765          BEQ  IF 3           ; YES-CONTINUE
8EB9:A9 C4      766          LDA  #196           ; NO-SELECT MESSAGE
8EBB 85 08      767          STA  ERRSYM1        ; THAT "THEN" OR "GOTO"
8EBD A9 AB      768          LDA  #171           ; IS EXPECTED
8EBF 4C D1 92   769          JMP  TXPSET2        PRINT ERROR MESSAGE
8EC2 20 B1 00   770 IF 3     JSR  CHRGET         ADVANCE PAST "THEN"
8EC5 90 EB      771          BCC  IF 1           IF DIGIT IS NEXT, SCAN LINE NUMBER
8EC7 68         772          PLA                 IF NOT, CANCEL END OF COMMAND CHECK
8EC8 68         773          PLA
8EC9 A5 B8      774 BACKUP   LDA  CHRPTR         & TREAT REMD R OF CMD AS NEW LINE
8ECB D0 02      775          BNE  BKUP 1         SO BACK UP PC 1 BYTE AS THOUGH
8ECD C6 B9      776          DEC  CHRPTR+1       IT POINTED TO THE END OF THE
8ECF C6 B8      777 BKUP 1   DEC  CHRPTR         PRECEDING INSTRUCTION
8ED1 60         778          RTS                 AND EXIT
8ED2            779 ;
8ED2 20 B1 00   780 REM1     JSR  CHRGET         ADVANCE TO NEXT CHAR
8ED5 C9 00      781 REM      CMP  #$00           DOES ANYTHING FOLLOW "REM"?
8ED7 D0 F9      782          BNE  REM1           ; YES KEEP SCANNING
8ED9 60         783 RTS5     RTS                 ; NO EXIT
8EDA            784 ;
8EDA 20 4E 90   785 ON       JSR  EVALNUM        SCAN NUMERIC INDEX
8EDD C9 AB      786          CMP  #171           IS KEYWORD "GOTO" NEXT?
8EDF F0 0D      787          BEQ  ON 1           ; YES-CONTINUE
8EE1 C9 B0      788          CMP  #176           ; NO, HOW ABOUT KEYWORD "GOSUB"?
8EE3 F0 09      789          BEQ  ON 1           ; YES CONTINUE
8EE5 A9 AB      790          LDA  #171           ; NO SELECT MESSAGE
8EE7 85 08      791          STA  ERRSYM1        ; SAYING "GOTO" OR "GOSUB"
8EE9 A9 B0      792          LDA  #176           ; IS EXPECTED
8EEB 4C D1 92   793          JMP  TXPSET2        PRINT ERROR MESSAGE
8EEE 20 B1 00   794 ON 1     JSR  CHRGET         ADVANCE PAST GOTO/GOSUB
8EF1 20 0A 90   795 ON 2     JSR  LNUM CK        SCAN LINE NUMBER FOLLOWING
8EF4 20 92 8F   796          JSR  COMAEOC        IS A COMMA NEXT?
8EF7 D0 F8      797          BNE  ON 2           ; YES-GO SCAN NEXT LINE NUMBER
8EF9:60         798          RTS                 ; NO-END OF COMMAND
8EFA            799 ;
8EFA 20 41 8E   800 WAIT     JSR  POKE           GET 2 NUMERIC OPERANDS
8EFD A9 2C      801          LDA  #','
8EFF 20 94 8F   802          JSR  SYMBEOC        DOES COMMA FOLLOW?
8F02:F0 D5      803          BEQ  RTS5           ; NO END OF COMMAND
8F04 4C 4E 90   804          JMP  EVALNUM        ; YES-GET 3RD OPERAND
8F07            805 ;
8F07 A9 C2      806 DEF      LDA  #194           KEYWORD "FN" MUST FOLLOW "DEF"
8F09 20 86 8F   807          JSR  SYMBLCK
8F0C 20 52 92   808          JSR  VARNAME        THEN A FUNCTION NAME
8F0F A9 28      809          LDA  #'(            AND A LEFT PARENTHESIS
8F11 20 86 8F   810          JSR  SYMBLCK
8F14 20 52 92   811          JSR  VARNAME        NEXT IS THE OPERAND VARIABLE
8F17 20 A9 91   812          JSR  PAREN2         AND RIGHT PARENTHESIS
8F1A A9 D0      813          LDA  #208           NEXT MUST BE KEYWORD "="
8F1C 20 86 8F   814          JSR  SYMBLCK
8F1F 4C 4E 90   815          JMP  EVALNUM        AND FINALLY, THE FORMULA
8F22            816 ;
8F22 20 4B 90   817 TAB      JSR  EVALN1         SCAN NUMBER OF SPACES TO TAB
8F25 20 A9 91   818          JSR  PAREN2         ADVANCE PAST CLOSING PARENTHESIS
8F28 4C 2E 8F   819          JMP  PRINT 1        AND CONTINUE SCANNING
8F2B 20 B1 00   820 PRINT 0  JSR  CHRGET         ADVANCE PAST SYMBOL
8F2E 20 B7 00   821 PRINT 1  JSR  CHRGOT         REFRESH STATUS & ACCUMULATOR
8F31 F0 A6      822 PRINT    BEQ  RTS5           END OF COMMAND HERE IS O.K.
8F33 C9 2C      823          CMP  #','           IS COMMA NEXT?
8F35 F0 F4      824          BEQ  PRINT 0        ; YES-CONTINUE
8F37 C9 3B      825          CMP  #';            NO, HOW ABOUT SEMICOLON?
8F39 F0 F0      826          BEQ  PRINT 0        ; YES CONTINUE
8F3B C9 C0      827          CMP  #192           NO, IS IT KEYWORD "TAB("?
8F3D F0 E3      828          BEQ  TAB            ; YES SCAN IT
8F3F C9 C3      829          CMP  #195           NO, IS IT KEYWORD "SPC("?
8F41 F0 DF      830          BEQ  TAB            ; YES, TREAT IT SAME AS "TAB("
8F43 20 7F 90   831          JSR  EVAL           ; NO-SCAN EXPRESSION TO BE PRINTED
8F46 4C 2E 8F   832          JMP  PRINT 1        THEN CONTINUE SCANNING
8F49            833
8F49 F0 8E      834 LIST     BEQ  RTS5           LIST BY ITSELF IS O.K.
8F4B B0 05      835          BCS  LIST 1         IF LINE NUMBER ISN'T NEXT, SKIP
8F4D 20 AD 8F   836          JSR  GETLNUM        OTHERWISE, SCAN LINE NUMBER
8F50 F0 50      837          BEQ  RTS6           END OF COMMAND HERE IS O.K.
8F52:C9 C9      838 LIST 1   CMP  #201           DOES KEYWORD "-" FOLLOW?
8F54 F0 11      839          BEQ  LIST 2         ; YES-CONTINUE
8F56 C9 2C      840          CMP  #','           NO, HOW ABOUT COMMA?
8F58 F0 0D      841          BEQ  LIST 2         ; YES CONTINUE
8F5A A9 C9      842          LDA  #201           NOTHING ELSE IS LEGAL HERE
8F5C 85 08      843          STA  ERRSYM1        ; SO MESSAGE SAYS HYPHEN,
8F5E A9 2C      844          LDA  #','           ; COMMA OR LINE NUMBER IS EXPECTED
8F60 85 09      845          STA  ERRSYM2
8F62 A9 01      846          LDA  #TXLNUM        SELECT ABOVE MESSAGE
8F64:4C D5 92   847          JMP  TXPSET3        PRINT ERROR MESSAGE
8F67:20 B1 00   848 LIST 2   JSR  CHRGET         ADVANCE PAST SYMBOL
8F6A:F0 36      849          BEQ  RTS6           END OF COMMAND HERE IS O.K.
8F6C:90 05      850          BCC  LIST 3         OTHERWISE, MUST BE LINE NUMBER
8F6E:A9 01      851          LDA  #TXLNUM        IF IT'S NOT THEN
8F70 4C C2 92   852          JMP  TXPEOC         ; PRINT ERROR MESSAGE
8F73 4C AD 8F   853 LIST 3   JMP  GETLNUM        OTHERWISE, SCAN LINE NUMBER
8F76            854 ;
8F76            855          CHN  TYPE=RIGHT$2   JUMP TO SECOND HALF OF SOURCE CODE
```

Note: **This is the second of two source listings.**

```
8F76                    1 *
8F76                    2 *   GENERAL SUBROUTINES
8F76                    3 *
8F76                    4 *
8F76                    5 *   CHECK THAT END OF COMMAND OCCURS HERE
8F76                    6 *
8F76 20 B7 00           7 EOC.CK  JSR  CHRGOT       ;END OF COMMAND?
8F79 F0 27              8         BEQ  RTS6         ; YES-EXIT
8F7B A9 3A              9         LDA  #':'         ; NO-MESSAGE INDICATES COLON
8F7D 85 08             10         STA  ERRSYM1      ;  OR RETURN IS EXPECTED HERE
8F7F A9 0B             11         LDA  #TXRETN      ;SELECT ABOVE MESSAGE
8F81 4C D1 92          12         JMP  TXPSET2      ;PRINT ERROR MESSAGE
8F84                   13 *
8F84                   14 *   CHECK THAT REQUIRED SYMBOL OCCURS HERE
8F84                   15 *
8F84 A9 2C             16 COMMA  LDA  #','         ;SPECIAL CHECK FOR COMMA
8F86 A0 00             17 SYMBLCK LDY  #$00        ;START HERE FOR OTHER SYMBOLS
8F88 D1 B8             18         CMP  (CHRPTR),Y  ;IS NEXT BYTE EQUAL TO ACCUM?
8F8A F0 03             19         BEQ  CGET1       ; YES-ADVANCE PAST SYMBOL
8F8C 4C CD 92          20         JMP  TXPSET1     ; NO-PRINT ERROR MESSAGE
8F8F 4C B1 00          21 CGET1  JMP  CHRGET
8F92                   22 *
8F92                   23 *   CHECK THAT EITHER OPTIONAL SYMBOL OR
8F92                   24 *   END OF COMMAND OCCURS HERE
8F92                   25 *
8F92 A9 2C             26 COMAEOC LDA  #','        ;SPECIAL CHECK FOR COMMA
8F94 A0 00             27 SYMBEOC LDY  #$00        ;START HERE FOR OTHER SYMBOLS
8F96 D1 B8             28         CMP  (CHRPTR),Y  ;IS NEXT BYTE EQUAL TO ACCUM?
8F98 D0 09             29         BNE  SYME.1      ;  NO-CHECK FOR END OF COMMAND
8F9A 20 B7 00          30         JSR  CHRGOT      ;  YES-SET FLAGS FOR THIS BYTE
8F9D 08                31         PHP
8F9E 20 B1 00          32         JSR  CHRGET      ;  AND ADVANCE TO NEXT BYTE
8FA1 28                33         PLP
8FA2 60                34 RTS6   RTS               ;NOW EXIT
8FA3 85 08             35 SYME.1 STA  ERRSYM1     ;SAVE SYMBOL IF NEEDED FOR ERROR MSG
8FA5 20 B7 00          36         JSR  CHRGOT      ;IS END OF COMMAND HERE?
8FA8 F0 F8             37         BEQ  RTS6        ; YES-EXIT
8FAA 4C C4 92          38         JMP  TXPEOC1     ; NO-PRINT ERROR MESSAGE
8FAD                   39 *
8FAD                   40 *   SCAN A LINE NUMBER
8FAD                   41 *
8FAD A9 00             42 GETLNUM LDA  #$00
8FAF 85 01             43         STA  ZREG1+1     ;INIT LINE NO TO #0
8FB1 85 00             44         STA  ZREG1       ;LO BYTE TOO
8FB3 A5 B8             45         LDA  CHRPTR      ;SAVE PROGRAM COUNTER (LO BYTE ONLY)
8FB5 48                46         PHA              ; IN CASE OF ERROR
8FB6 20 B7 00          47         JSR  CHRGOT      ;GET FIRST DIGIT OF LINE NO
8FB9 90 06             48         BCC  GLNUM.1     ; IT'S A DIGIT, GO SCAN LINE NUMBER
8FBB 68                49         PLA              ;IT'S NOT, SO ERROR MESSAGE
8FBC A9 01             50         LDA  #TXLNUM     ; SAYS "LINE NUMBER EXPECTED"
8FBE 4C CD 92          51         JMP  TXPSET1     ;PRINT ERROR MESSAGE
8FC1 E9 2F             52 GLNUM.1 SBC  #$2F        ;CONVERT DIGIT FROM ASCII TO VALUE
8FC3 48                53         PHA              ;SAVE THIS DIGIT
8FC4 A5 01             54         LDA  ZREG1+1     ;LOOK AT LINE NUMBER SCANNED SO FAR
8FC6 C9 19             55         CMP  #$19        ;IS LNUM > 63999?
8FC8 90 0F             56         BCC  GLNUM.3     ; NO-CONTINUE
8FCA 68                57         PLA              ; YES-ERROR
8FCB 20 B1 00          58 GLNUM.2 JSR  CHRGET      ;GET ANY REMAINING DIGITS ON ERROR
8FCE 90 FB             59         BCC  GLNUM.2     ;IF MORE DIGITS FOLLOW, GO SCAN THEM
8FD0 68                60         PLA              ;RETRIEVE PC AT START OF LINE NUMBER
8FD1 20 8D 92          61         JSR  SETINV2     ;HIGHLIGHT ENTIRE LINE NUMBER
8FD4 A2 09             62         LDX  #LNUM64     ;SELECT MESSAGE "LINE NO > 63999"
8FD6 4C DC 92          63         JMP  ERROR       ;PRINT ERROR MESSAGE
8FD9                   64 *
8FD9                   65 *   MULTIPLY LINE NUMBER SO FAR BY 10
8FD9                   66 *
8FD9 A5 00             67 GLNUM.3 LDA  ZREG1       ;LINE NUMBER SO FAR, LO BYTE
8FDB 0A                68         ASL  A
8FDC 2A                69         ROL  A
8FDD 08                70         PHP              ;SAVE CARRY = LO BYTE, BIT 6
8FDE 6A                71         ROR  A           ;NOW CARRY = LO BYTE, BIT 7
8FDF A5 01             72         LDA  ZREG1+1
8FE1 2A                73         ROL  A           ;PULL BIT 7 INTO HI BYTE
8FE2 28                74         PLP
8FE3 2A                75         ROL  A           ;AND NOW BIT 6
8FE4 48                76         PHA              ;SAVE 4 x OLD LINE NO, HI BYTE
8FE5 A5 00             77         LDA  ZREG1
8FE7 0A                78         ASL  A
8FE8 0A                79         ASL  A           ;4 x OLD LINE NO, LO BYTE
8FE9 18                80         CLC
8FEA 65 00             81         ADC  ZREG1       ;4 x OLD + 1 x OLD
8FEC 85 00             82         STA  ZREG1       ;= 5 x OLD LINE NO, LO BYTE
8FEE 68                83         PLA              ;RETRIEVE 4 x OLD LINE NO, HI BYTE
8FEF 65 01             84         ADC  ZREG1+1     ;4 x OLD + 1 x OLD
8FF1 85 01             85         STA  ZREG1+1     ;= 5 x OLD LINE NO, HI BYTE
8FF3 06 00             86         ASL  ZREG1       ;THEN x 2 = x 10 TOTAL
8FF5 26 01             87         ROL  ZREG1+1     ;HI BYTE TOO
8FF7 68                88         PLA              ;GET NEW DIGIT
8FF8 65 00             89         ADC  ZREG1       ;ADD TO 10 x OLD LINE NO
8FFA 85 00             90         STA  ZREG1       ; AND SAVE AS NEW LINE NO
8FFC 90 02             91         BCC  GLNUM.4
8FFE E6 01             92         INC  ZREG1+1     ;DON'T FORGET THE CARRY, IF ANY
9000 20 B1 00          93 GLNUM.4 JSR  CHRGET      ;ANY MORE DIGITS?
9003 90 BC             94         BCC  GLNUM.1     ; YES, GO GET THEM
9005 68                95         PLA              ;END WITH FIRST DIGIT ADDR IN X-REG
9006 AA                96         TAX              ; IN CASE OF UNDEF'D LINE NO
9007 4C B7 00          97         JMP  CHRGOT      ;AND EXIT
900A                   98 *
900A                   99 *   CHECK THAT THE REFERENCED LINE NUMBER EXISTS
900A                  100 *
900A 20 AD 8F         101 LNUM.CK JSR  GETLNUM     ;FIRST SCAN THE REF'D LINE NUMBER
900D 8A               102         TXA              ;SAVE PC OF FIRST DIGIT
900E 48               103         PHA              ; IN CASE OF ERROR
900F A5 68            104         LDA  PROGRAM+1
9011 A4 67            105         LDY  PROGRAM
9013 85 03            106         STA  ZREG2+1     ;POINT TO START OF PROGRAM
9015 84 02            107         STY  ZREG2
9017 A0 00            108 LNUM.1  LDY  #$00
9019 B1 02            109         LDA  (ZREG2),Y   ;LOAD LINK ADDRESS, LO BYTE
901B C8               110         INY              ;THEN HI BYTE
901C 11 02            111         ORA  (ZREG2),Y   ;END OF PROGRAM? (LINK ADDR = $0000)
901E D0 09            112         BNE  LNUM.2      ; NO-CONTINUE
9020 68               113         PLA              ; YES-REF'D LINE NO WASN'T FOUND
9021 20 8D 92         114         JSR  SETINV2     ;HIGHLIGHT LINE NUMBER
9024 A2 0A            115         LDX  #UNDSTMT    ;SELECT 'UNDEF'D STATEMENT'
9026 4C E6 92         116         JMP  ERRMSG      ;PRINT ERROR MESSAGE
9029 C8               117 LNUM.2  INY              ;NOW LOOK AT LINE NUMBER
902A B1 02            118         LDA  (ZREG2),Y   ;LO BYTE FIRST
902C C5 00            119         CMP  ZREG1       ;= REF'D LINE, LO BYTE?
902E D0 0B            120         BNE  LNUM.3      ; NO-ADVANCE TO NEXT LINE
9030 C8               121         INY              ; YES-CHECK HI BYTE
9031 B1 02            122         LDA  (ZREG2),Y   ;LOAD HI BYTE
9033 C5 01            123         CMP  ZREG1+1     ;IS THIS THE REF'D LINE NO?
9035 D0 04            124         BNE  LNUM.3      ; NO-ADVANCE TO NEXT LINE
9037 68               125         PLA              ; YES-BALANCE STACK
9038 4C B7 00         126         JMP  CHRGOT      ;  AND END
903B A0 00            127 LNUM.3  LDY  #$00
903D B1 02            128         LDA  (ZREG2),Y   ;LOAD LINK ADDRESS, LO BYTE
903F 48               129         PHA              ;SAVE IT
9040 C8               130         INY              ;ADVANCE TO HI BYTE
9041 B1 02            131         LDA  (ZREG2),Y   ;LOAD IT
9043 85 03            132         STA  ZREG2+1     ;POINT TO NEXT LINE
9045 68               133         PLA
9046 85 02            134         STA  ZREG2       ;LO BYTE TOO
9048 4C 17 90         135         JMP  LNUM.1      ;NOW CHECK NEXT LINE
904B                  136 *
904B                  137 *   EVALUATE EXPRESSIONS
904B                  138 *
904B 20 B1 00         139 EVALN1  JSR  CHRGET      ;ADVANCE TO FIRST CHAR OF EXPRESSION
904E A5 B8            140 EVALNUM LDA  CHRPTR      ;LOAD PC OF FIRST CHAR
9050 48               141         PHA              ;SAVE IT IN CASE OF ERROR
9051 20 7F 90         142         JSR  EVAL        ;SCAN AN EXPRESSION
9054 68               143         PLA
9055 AA               144         TAX              ;BRING BACK PC OF FIRST CHAR
9056 4C 6A 90         145         JMP  NUM.CK      ;ASSURE THAT EXPRESSION WAS NUMERIC
9059 4C B7 00         146 CGOT1   JMP  CHRGOT
905C                  147
905C 20 B1 00         148 EVALS1  JSR  CHRGET      ;ADVANCE TO FIRST CHAR OF EXPRESSION
905F A5 B8            149 EVALSTR LDA  CHRPTR      ;LOAD PC OF FIRST CHAR
9061 48               150         PHA              ;SAVE IT IN CASE OF ERROR
9062 20 7F 90         151         JSR  EVAL        ;SCAN AN EXPRESSION
9065 68               152         PLA
9066 AA               153         TAX              ;BRING BACK PC OF FIRST CHAR
9067                  154 ...     JMP  STR.CK      ;ASSURE THAT EXPRESSION WAS NUMERIC
9067                  155
9067                  156 *   ASSURE THAT STRING/NUMERIC TYPES ARE AS REQUIRED
9067                  157 *
9067:A9 FF            158 STR.CK  LDA  #$FF        ;$FF INDICATES STRING TYPE REQUIRED
9069 2C               159         DFB  $2C         ; = IGNORE NEXT LINE
906A A9 00            160 NUM.CK  LDA  #$00        ;$00 INDICATES NUMERIC TYPE REQUIRED
906C C5 11            161 TYPE.CK CMP  TYP.STR     ;DO REQUIRED AND ACTUAL TYPES AGREE?
906E F0 E9            162         BEQ  CGOT1       ; YES-EXIT
9070 48               163         PHA              ; NO--SAVE TYPE INDICATOR
9071 8A               164         TXA              ;RETRIEVE PC OF START OF EXPRESSION
9072 20 8D 92         165         JSR  SETINV2     ;  AND HIGHLIGHT ENTIRE EXPRESSION
9075 A2 02            166         LDX  #NUMVEXP    ;START BY TRYING "NUMERIC EXPECTED"
9077 68               167         PLA              ;BUT WAS NUMERIC REALLY EXPECTED?
9078 F0 02            168         BEQ  TYPCK.1     ; YES, WE'RE ALL SET
907A A2 03            169         LDX  #STRVEXP    ; NO-USE "STRING EXPECTED"
907C 4C DC 92         170 TYPCK.1 JMP  ERROR       ;PRINT ERROR MESSAGE
907F                  171 *
907F                  172 *   SCAN (PSEUDO-EXECUTE) AN APPLESOFT EXPRESSION
907F                  173 *
907F A9 00            174 EVAL    LDA  #$00        ;PLACE END-OF-EVALUATION
9081 48               175         PHA              ; MARKER ON STACK
9082 20 11 91         176 EVAL.1  JSR  EVAL1       ;SCAN ONE ELEMENT OF EXPRESSION
9085 C9 D2            177         CMP  #210        ;DOES A BINARY OPERATION FOLLOW?
9087 B0 63            178         BCS  EVAL.9      ; NO-CHECK IF END OF EXPRESSION
9089 C9 C8            179         CMP  #200        ;SAME QUESTION
908B 90 5F            180         BCC  EVAL.9      ; NO
908D E9 C8            181         SBC  #200        ; YES-COMPUTE INDEX FOR OPERATION
908F AA               182         TAX
9090 68               183 EVAL.2  PLA              ;PRIORITY OF PREVIOUS OPERATION
9091 DD 39 94         184         CMP  HIER,X      ;PRIORITY OF THIS OPERATION
9094 90 06            185         BCC  EVAL.3      ;THIS > OLD, SO DO THIS FIRST
9096 20 F2 90         186         JSR  DO.OPER     ;THIS <= OLD, SO DO OLD FIRST
9099 4C 90 90         187         JMP  EVAL.2      ;AND EXAMINE ANOTHER "OLD" OPERATION
909C                  188 *
909C 48               189 EVAL.3  PHA              ;RE-PUSH PREVIOUS OPERATION'S PRIORITY
909D A5 11            190         LDA  TYP.STR
909F 48               191         PHA              ;PUSH TYPE,
90A0 A5 B8            192         LDA  CHRPTR
90A2 48               193         PHA              ; THEN CHRPTR,
90A3 BD 39 94         194         LDA  HIER,X
90A6 48               195         PHA              ; THEN PRIORITY
90A7 E0 07            196         CPX  #$07        ;IS THIS A COMPARISON (<=>)?
90A9 B0 15            197         BCS  EVAL.5      ; YES-TREAT IT SEPARATELY
90AB 8A               198         TXA              ;IS IT "+"?
90AC F0 BC            199         BEQ  EVAL.4      ; YES-NO TYPE CHECK
90AE A5 11            200         LDA  TYP.STR     ; OTHERWISE, ASSURE TYPE IS NUMERIC
90B0 F0 08            201         BEQ  EVAL.4
90B2 20 83 92         202         JSR  SETINV      ;STRING EXPRESSION, SO ERROR
90B5 A2 BC            203         LDX  #NOTWSTR    ;SELECT "OPER NOT ALLOWED W/ STRING"
90B7 4C DC 92         204         JMP  ERROR       ;PRINT ERROR MESSAGE
90BA 20 B1 00         205 EVAL.4  JSR  CHRGET      ;ADVANCE PAST OPERATION
90BD 4C 82 90         206         JMP  EVAL.1      ; AND CONTINUE SCANNING
90C0 A9 00            207 EVAL.5  LDA  #$00        ;CLEAR RECORD OF COMPARISONS
90C2 85 00            208         STA  ZREG1
90C4 20 B7 00         209         JSR  CHRGOT      ;WHAT WAS THAT COMPARISON AGAIN?
90C7 38               210 EVAL.6  SEC              ;  IN    >    =    <
90C8 E9 CF            211         SBC  #207        ;OUT   $00  $01  $02
90CA C9 03            212         CMP  #3          ;IS THIS A COMPARISON?
90CC B0 B4            213         BCS  EVAL.1      ; NO-CONTINUE SCANNING
90CE C9 01            214         CMP  #1          ;OUT  C-CLR C-SET C-SET
90D0 2A               215         ROL  A           ;OUT   $00  $03  $05
90D1 49 01            216         EOR  #1          ;OUT   $01  $02  $04
90D3 45 00            217         EOR  ZREG1       ;HAS ANY COMPARISON NOW BEEN
90D5 C5 00            218         CMP  ZREG1       ; MADE TWICE (E.G. "X >=< Y")?
90D7 B0 08            219         BCS  EVAL.7      ; NO-CONTINUE
90D9 20 83 92         220         JSR  SETINV      ; YES-HIGHLIGHT SECOND OCCURRENCE
90DC A2 06            221         LDX  #DUPLSYM    ;SELECT "DUPLICATE SYMBOL"
90DE 4C DC 92         222         JMP  ERROR       ;PRINT ERROR MESSAGE
90E1 85 00            223 EVAL.7  STA  ZREG1       ;SAVE HISTORY OF COMPARISONS
90E3 20 B1 00         224         JSR  CHRGET      ;ADVANCE PAST THIS COMPARISON
90E6 4C C7 90         225         JMP  EVAL.6      ;CHECK FOR ANOTHER COMPARISON
90E9 20 F2 90         226 EVAL.8  JSR  DO.OPER      ;PSEUDO-EXECUTE OPERATION ON STACK
90EC 68               227 EVAL.9  PLA              ;ANY OPERATIONS LEFT ON STACK?
90ED D0 FA            228         BNE  EVAL.8      ; YES-EXECUTE THE NEXT ONE
```

```
90EF 4C B7 00  229              JMP  CHRGOT        ;NO-EXIT
90F2           230
90F2           231      ;PSEUDO-EXECUTE PREVIOUSLY SCANNED OPERATION
90F2           232      ;NOW STORED ON THE STACK.
90F2           233
90F2 86 02     234 DO.OPER STX  ZREG2         ;MUST PRESERVE X-REG
90F4 A8        235              TAY                ;PUT PRIORITY OF OPERATION IN Y-REG
90F5 68        236              PLA                ;PULL AND
90F6 85 00     237              STA  ZREG1         ; SAVE RETURN ADDRESS FROM WHICH
90F8 68        238              PLA                ; "DO.OPER" WAS CALLED
90F9 85 01     239              STA  ZREG1+1       ;HI BYTE TOO
90FB 68        240              PLA
90FC AA        241              TAX                ;PULL CHRPTR
90FD E8        242              INX                ; AND ADVANCE PAST OPERATION SYMBOL
90FE 68        243              PLA                ;PULL TYPE
90FF 20 6C 90  244              JSR  TYPE.CK       ;CHECK FOR TYPE MISMATCH
9102 A5 01     245              LDA  ZREG1+1
9104 48        246              PHA                ;RESTORE RETURN ADDRESS NOW THAT
9105 A5 00     247              LDA  ZREG1         ; OUR DATA IS OFF THE STACK
9107 48        248              PHA
9108 A6 02     249              LDX  ZREG2         ;RESTORE X-REG
910A C0 03     250              CPY  #$03          ;IS OPERATION A COMPARISON (>=<)?
910C D0 6C     251              BNE  RTS7          ; NO-EXIT
910E 4C 11 92  252              JMP  SET.NUM       ; YES-RESULT IS BOOLEAN (NUMERIC)
9111           253
9111           254      ;SCAN ONE ELEMENT OF AN EXPRESSION
9111           255
9111 20 B7 00  256 EVAL1   JSR  CHRGOT        ;SET STATUS FOR FIRST BYTE OF EXPR
9114 90 36     257              BCC  NUMBER        ;SCAN NUMBER IF FIRST IS A DIGIT
9116 C9 2E     258              CMP  #"."
9118 F0 32     259              BEQ  NUMBER        ; OR A DECIMAL POINT
911A 20 7D E0  260              JSR  ABC.CK        ;IF ALPHABETIC
911D B0 27     261              BCS  VAR1          ; THEN SCAN VARIABLE
911F C9 C6     262              CMP  #198          ;IF KEYWORD "NOT"
9121 F0 58     263              BEQ  UNARY
9123 C9 C8     264              CMP  #200          ; OR KEYWORD "+"
9125 F0 54     265              BEQ  UNARY
9127 C9 C9     266              CMP  #201          ; OR KEYWORD "-"
9129 F0 50     267              BEQ  UNARY         ; THEN SCAN UNARY OPERAND
912B C9 22     268              CMP  #""           ;IF A QUOTATION MARK
912D F0 5A     269              BEQ  QUOTE         ; THEN SCAN THE QUOTATION
912F C9 28     270              CMP  #"("          ;IF A LEFT PARENTHESIS
9131 F0 6E     271              BEQ  PAREN         ; THEN SCAN PARENTHETIC EXPRESSION
9133 C9 D2     272              CMP  #210          ;IF AN APPLSOFT FUNCTION (E.G. SIN)
9135 B0 12     273              BCS  FN1           ; THEN SCAN ITS OPERAND(S)
9137 C9 C2     274              CMP  #194          ;IF KEYWORD "FN"
9139 F0 73     275              BEQ  DEF.FN        ; THEN SCAN FUNCTION NAME AND OPERAND
913B 20 A8 92  276              JSR  SETSPC        ;IF NOT A VALID SYMBOL THEN
913E A2 04     277              LDX  #EXPREXP      ; SELECT "EXPRESSION EXPECTED"
9140 4C DC 92  278              JMP  ERROR         ; AND PRINT ERROR MESSAGE
9143 4C B7 00  279 CGOT2   JMP  CHRGOT
9146 4C 28 92  280 VAR1    JMP  VAR
9149 4C BC 91  281 FN1     JMP  FN
914C           282
914C           283      ;SCAN A REAL NUMBER
914C           284
914C A9 00     285 NUMBER  LDA  #$00           ;SINCE THIS IS A NUMBER
914E 85 11     286              STA  TYP.STR       ; SET TYPE TO "NUMERIC"
9150 F0 03     287              BEQ  NUM.2         ;(SKIP NEXT LINE FIRST TIME)
9152 20 B1 00  288 NUM.1   JSR  CHRGET        ;ADVANCE TO NEXT CHAR OF NUMBER
9155 20 B7 00  289 NUM.2   JSR  CHRGOT        ;SET STATUS FLAGS
9158 90 F8     290              BCC  NUM.1         ;IF A DIGIT THEN ADVANCE PAST IT
915A C9 2E     291              CMP  #"."          ;IS DECIMAL POINT NEXT?
915C D0 05     292              BNE  NUM.4         ; NO-CONTINUE
915E 20 B1 00  293 NUM.3   JSR  CHRGET        ; YES-ADVANCE PAST IT
9161 90 FB     294              BCC  NUM.3         ;  AND ANY FOLLOWING DIGITS
9163 C9 45     295 NUM.4  CMP  #"E"          ;IS "E" (FOR "EXPONENT") NEXT?
9165 D0 13     296              BNE  RTS7          ; NO-NUMBER ENDS HERE
9167 20 B1 00  297              JSR  CHRGET        ; YES-ADVANCE PAST IT
916A C9 C8     298              CMP  #200          ;FOLLOWED BY KEYWORD "+"?
916C F0 04     299              BEQ  NUM.5         ; YES, THAT'S O.K.
916E C9 C9     300              CMP  #201          ; OR KEYWORD "-"
9170 D0 03     301              BNE  NUM.6         ; NO-SKIP
9172 20 B1 00  302 NUM.5   JSR  CHRGET        ;ADVANCE PAST "+" OR "-"
9175 20 B7 00  303 NUM.6   JSR  CHRGOT        ;OTHERWISE, SET STATUS BITS
9178 90 F8     304              BCC  NUM.5         ;IF MORE DIGITS, SCAN THEM
917A 60        305 RTS7    RTS                ;THEN EXIT
917B           306
917B           307      ;SCAN OPERAND OF UNARY OPERATOR (POSITIVE, NEGATIVE, NOT)
917B           308
917B 20 B1 00  309 UNARY   JSR  CHRGET        ;ADVANCE PAST OPERATOR
917E A5 B8     310              LDA  CHRPTR        ;SAVE PC OF CHAR AFTER OPERATOR
9180 48        311              PHA                ; IN CASE OF TYPE MISMATCH ERROR
9181 20 11 91  312              JSR  EVAL1         ;SCAN THE UNARY OPERATOR'S OPERAND
9184 68        313              PLA
9185 AA        314              TAX                ;RETRIEVE PC OF FIRST CHAR OF EXPR
9186 4C 6A 90  315              JMP  NUM.CK        ;ASSURE THAT OPERAND WAS NUMERIC
9189           316
9189           317      ;SCAN A QUOTATION
9189           318
9189 A9 FF     319 QUOTE   LDA  #$FF
918B 85 11     320              STA  TYP.STR       ;SET "STRING" TYPE
918D 20 B1 00  321 TO.EOQ  JSR  CHRGET        ;GET ONE CHAR OF QUOTATION
9190 C9 00     322              CMP  #$00          ;END-OF-LINE?
9192 F0 AF     323              BEQ  CGOT2         ; YES-SET FLAGS FOR EOL AND EXIT
9194 C9 22     324              CMP  #""           ;ENDING QUOTATION MARK?
9196 D0 F5     325              BNE  TO.EOQ        ; NO-GO TO NEXT CHAR
9198 20 B7 00  326              JSR  CHRGOT        ;SET FLAGS FOR CLOSING QUOTE
919B 08        327              PHP
919C 20 B1 00  328              JSR  CHRGET        ;ADVANCE PAST CLOSING QUOTE
919F 28        329              PLP
91A0 60        330              RTS                ;AND EXIT
91A1           331
91A1           332      ;SCAN EXPRESSION (NUMERIC OR STRING) IN PARENTHESES
91A1           333
91A1 A9 28     334 PAREN   LDA  #"("
91A3 20 86 8F  335              JSR  SYMBLCK       ;ASSURE THAT LEFT PARENTHESIS EXISTS
91A6 20 7F 90  336              JSR  EVAL          ;FOLLOWED BY AN EXPRESSION
91A9 A9 29     337 PAREN2  LDA  #")"
91AB 4C 86 8F  338              JMP  SYMBLCK       ;AND A RIGHT PARENTHESIS
91AE           339
91AE           340      ;SCAN DEFINED FUNCTION OR APPLESOFT REAL FUNCTION
91AE           341 ;
91AE 20 4F 92  342 DEF.FN  JSR  VNAME1        ;SCAN FUNCTION NAME
91B1 A5 B8     343 REAL.FN LDA  CHRPTR        ;SAVE PC OF START OF OPERAND
91B3 48        344              PHA                ; IN CASE OF TYPE MISMATCH ERROR
91B4 20 A1 91  345              JSR  PAREN         ;SCAN OPERAND IN PARENTHESES
91B7 68        346              PLA
91B8 AA        347              TAX                ;RETRIEVE PC OF START OF OPERAND
91B9 4C 6A 90  348              JMP  NUM.CK        ;ASSURE OPERAND WAS NUMERIC
91BC           349 ;
91BC           350      ;SCAN AN APPLESOFT FUNCTION
91BC           351 ;
91BC 48        352 FN      PHA                ;SAVE THE FUNCTION
91BD 20 B1 00  353              JSR  CHRGET        ;ADVANCE PAST FUNCTION
91C0 68        354              PLA                ;BUT LOOK AT IT AGAIN
91C1 C9 E8     355              CMP  #235          ;WAIT--IS IT REALLY A KEYWORD?
91C3 90 01     356              BCC  FN.1          ; YES-ALL IS O.K.
91C5 00        357              BRK                ; NO--SOMETHING IS VERY WRONG
91C6 C9 D7     358 FN.1    CMP  #215          ;IS THIS THE "SCRN" FUNCTION?
91C8 F0 58     359              BEQ  SCRN          ; YES-GO SCAN IT
91CA C9 E3     360              CMP  #227          ;IS IT A REAL OR STRING FUNCTION?
91CC 90 E3     361              BCC  REAL.FN       ; REAL SO GO SCAN IT
91CE C9 E4     362              CMP  #228          ;IS IT "STR$"?
91D0 F0 46     363              BEQ  STR           ; YES-GO SCAN IT
91D2 C9 E7     364              CMP  #231          ;IS IT "CHR$"?
91D4 F0 42     365              BEQ  CHR           ; YES-GO SCAN IT
91D6 90 2F     366              BCC  S.TO.R        ;GO SCAN "LEN", "VAL" OR "ASC"
91D8 C9 EA     367              CMP  #234          ;IS IT "MID$"?
91DA F0 17     368              BEQ  MID           ; YES-GO SCAN IT
91DC 20 E5 91  369              JSR  STHENR        ;FOR "LEFT$" & "RIGHT$", SCAN STRI
91DF 20 A9 91  370              JSR  PAREN2        ; THEN REAL, THEN RIGHT PARENTHESI
91E2 4C 1B 92  371              JMP  SET.STR       ; & SET RESULT TYPE TO STRING
91E5           372 ;
91E5 A9 28     373 STHENR  LDA  #"("           ;ASSURE THAT EXPR STARTS
91E7 20 86 8F  374              JSR  SYMBLCK       ; WITH OPEN PARENTHESIS
91EA 20 6F 90  375              JSR  EVALSTR       ;SCAN STRING EXPRESSION
91ED 20 84 8F  376              JSR  COMMA         ;COMMA IS NEXT
91F0 4C 4E 90  377              JMP  EVALNUM       ;FINALLY, NUMERIC EXPRESSION
91F3           378 ;
91F3 20 E5 91  379 MID     JSR  STHENR        ;SCAN STRING, THEN REAL EXPRESSION
91F6 C9 2C     380              CMP  #","          ;IS THERE A THIRD OPERAND?
91F8 D0 06     381              BNE  MID.1         ; NO-SKIP
91FA 20 B1 00  382              JSR  CHRGET        ; YES-ADVANCE PAST COMMA
91FD 20 4E 90  383              JSR  EVALNUM       ;  AND SCAN NUMERIC EXPRESSION
9200 20 A9 91  384 MID.1   JSR  PAREN2        ;FINALLY, CLOSING PARENTHESIS
9203 4C 1B 92  385              JMP  SET.STR       ;SET RESULT TYPE TO STRING AND EXI
9206           386 ;
9206 A5 B8     387 S.TO.R  LDA  CHRPTR        ;SAVE P.C. OF START OF EXPR
9208 48        388              PHA                ; IN CASE OF TYPE MISMATCH
9209 20 A1 91  389              JSR  PAREN         ;SCAN OPERAND IN PARENTHESES
920C 68        390              PLA
920D AA        391              TAX                ;RETRIEVE P.C. OF START OF EXPR
920E 20 67 90  392              JSR  STR.CK        ;AND ASSURE THAT OPERAND WAS STRIN
9211 A9 00     393 SET.NUM LDA  #$00           ;SET RESULT OF "LEN", "VAL" AND
9213 85 11     394              STA  TYP.STR       ; "ASC" TO NUMERIC TYPE
9215 4C B7 00  395 CGOT3   JMP  CHRGOT        ;AND EXIT
9218           396 ;
9218           397 STR     EQU  *
9218           398 CHR     EQU  *
9218 20 B1 91  399              JSR  REAL.FN       ;SCAN REAL OPERAND OF "STR$" OR "CI
921B A9 FF     400 SET.STR LDA  #$FF          ;SET RESULT OF FUNCTION
921D 85 11     401              STA  TYP.STR       ; TO STRING TYPE
921F 4C B7 00  402              JMP  CHRGOT        ;AND EXIT
9222           403 ;
9222 20 41 8E  404 SCRN    JSR  PLOT          ;FOR SCRN FUNCITON, SCAN COORDINATI
9225 4C A9 91  405              JMP  PAREN2        ;AND FINAL PARENTHESIS
9228           406 ;
9228           407 ;    SCAN AN APPLESOFT VARIABLE
9228           408 ;
9228 20 52 92  409 VAR     JSR  VARNAME       ;SCAN THE VARIABLE'S NAME
922B 20 6F 92  410              JSR  VARTYPE       ;ALSO THE $/% TYPE INDICATOR IF AN
922E C9 28     411              CMP  #"("          ;IS IT AN ARRAY?
9230 D0 E3     412              BNE  CGOT3         ; NO-WE'RE DONE
9232 A5 11     413              LDA  TYP.STR       ; YES-SAVE THE STRING/NUMERIC TYPE
9234 48        414              PHA                ;  FOR WHEN WE'RE DONE
9235 20 4B 90  415 VAR.1   JSR  EVALN1        ;SCAN ARRAY'S INDEX
9238 C9 2C     416              CMP  #","          ;ARE THERE MORE INDICES?
923A F0 F9     417              BEQ  VAR.1         ; YES-SCAN THEM
923C C9 29     418              CMP  #")"          ;DOES RIGHT PARENTHESIS FOLLOW?
923E F0 09     419              BEQ  VAR.2         ; YES-ALL IS WELL
9240 A9 2C     420              LDA  #","          ; NO-MESSAGE INDICATES
9242 85 08     421              STA  ERRSYN1       ;  COMMA OR RIGHT PARENTHESIS
9244 A9 29     422              LDA  #")"          ;  IS EXPECTED HERE
9246 4C D1 92  423              JMP  TXPSET2       ;PRINT ERROR MESSAGE
9249 68        424 VAR.2   PLA                ;RETRIEVE TYPE OF ORIGINAL VARIABLE
924A 85 11     425              STA  TYP.STR       ;SET RESULT TO THAT TYPE
924C 4C B1 00  426              JMP  CHRGET        ; AND EXIT
924F           427 ;
924F           428 ;   SCAN ALPHANUMERIC VARIABLE NAME
924F           429 ;
924F 20 B1 00  430 VNAME1  JSR  CHRGET        ;ADVANCE TO FIRST CHAR OF VAR'S NA
9252 20 B7 00  431 VARNAME JSR  CHRGOT        ;GET FIRST CHAR
9255 20 7D E0  432              JSR  ABC.CK        ;IS IT ALPHABETIC?
9258 B0 08     433              BCS  VNAME.1       ; YES-O.K. SO FAR
925A 20 A0 92  434              JSR  SETSPC        ; NO-PUT A HIGHLIGHT SPACE HERE
925D A2 05     435              LDX  #VAREXP       ;SELECT "VARIABLE EXPECTED" MESSAGE
925F 4C DC 92  436              JMP  ERROR         ;PRINT ERROR MESSAGE
9262 20 B1 00  437 VNAME.1 JSR  CHRGET        ;ADVANCE TO NEXT CHAR
9265 90 FB     438              BCC  VNAME.1       ;IF NUMERIC KEEP GOING
9267 20 7D E0  439              JSR  ABC.CK        ;IS IT ALPHABETIC?
926A B0 F6     440              BCS  VNAME.1       ; YES-KEEP GOING
926C 4C B7 00  441              JMP  CHRGOT        ; NO-END OF VARIABLE NAME
926F           442 ;
926F           443 ;    SCAN VARIABLE TYPE INDICATOR (% OR $)
926F           444 ;
926F A9 00     445 VARTYPE LDA  #$00           ;INITIALIZE TYPE TO NUMERIC
9271 85 11     446              STA  TYP.STR
9273 20 B7 00  447              JSR  CHRGOT        ;GET TYPE INDICATOR
9276 C9 25     448              CMP  #"%"          ;IS IT INTEGER TYPE?
9278 F0 06     449              BEQ  VTYPE.1       ; YES-LEAVE IT NUMERIC
927A C9 24     450              CMP  #"$"          ;IS IT STRING TYPE?
927C D0 97     451              BNE  CGOT3         ; NO-NO TYPE INDICATOR--EXIT
927E C6 11     452              DEC  TYP.STR       ; YES-SET STRING TYPE
9280 4C B1 00  453 VTYPE.1 JMP  CHRGET        ;ADVANCE PAST TYPE INDICATOR AND EX
9283           454 ;
```

```
93E3:66 8E      679          DW   XDRAW-1
93E5:4D 90      680          DW   EVALNUM-1    ;HTAB
93E7:57 FF      681          DW   RTS-1        ;HOME
93E9:4D 90      682          DW   EVALNUM-1    ;ROT=
93EB:4D 90      683          DW   EVALNUM-1    ;SCALE=
93ED:57 FF      684          DW   RTS-1        ;SHLOAD
93EF:57 FF      685          DW   RTS-1        ;TRACE
93F1:57 FF      686          DW   RTS-1        ;NOTRACE
93F3:57 FF      687          DW   RTS-1        ;NORMAL
93F5:57 FF      688          DW   RTS-1        ;INVERSE
93F7:57 FF      689          DW   RTS-1        ;FLASH
93F9:4D 90      690          DW   EVALNUM-1    ;COLOR=
93FB:57 FF      691          DW   RTS-1        ;POP
93FD:4D 90      692          DW   EVALNUM-1    ;VTAB
93FF:4D 90      693          DW   EVALNUM-1    ;HIMEM:
9401:4D 90      694          DW   EVALNUM-1    ;LOMEM:
9403:73 8E      695          DW   ONERR-1
9405:57 FF      696          DW   RTS-1        ;RESUME
9407:7B 8E      697          DW   RECALL-1
9409:7B 8E      698          DW   STORE-1
940B:4D 90      699          DW   EVALNUM-1    ;SPEED=
940D:89 8E      700          DW   LET-1
940F:09 90      701          DW   LNUM.CK-1    ;GOTO
9411:9B 8E      702          DW   RUN-1
9413:A7 8E      703          DW   IF-1
9415:57 FF      704          DW   RTS-1        ;RESTORE
9417:2F 8E      705          DW   AMPER-1
9419:09 90      706          DW   LNUM.CK-1    ;GOSUB
941B:57 FF      707          DW   RTS-1        ;RETURN
941D:D4 8E      708          DW   REM-1
941F:57 FF      709          DW   RTS-1        ;STOP
9421:D9 8E      710          DW   ON-1
9423:F9 8E      711          DW   WAIT-1
9425:57 FF      712          DW   RTS-1        ;LOAD
9427:57 FF      713          DW   RTS-1        ;SAVE
9429:06 8F      714          DW   DEF-1
942B:40 8E      715          DW   POKE-1
942D:30 8F      716          DW   PRINT-1
942F:57 FF      717          DW   RTS-1        ;CONT
9431:48 8F      718          DW   LIST-1
9433:57 FF      719          DW   RTS-1        ;CLEAR
9435:E3 8D      720          DW   GET-1
9437:57 FF      721          DW   RTS-1        ;NEW
9439:           722 ;
9439:           723 ;   PRIORITY OF OPERATIONS
9439:           724 ;
9439:04         725 HIER     DFB  4            ;PRIORITY OF +
943A:04         726          DFB  4            ;PRIORITY OF -
943B:05         727          DFB  5            ;PRIORITY OF *
943C:05         728          DFB  5            ;PRIORITY OF /
943D:06         729          DFB  6            ;PRIORITY OF ^
943E:02         730          DFB  2            ;PRIORITY OF AND
943F:01         731          DFB  1            ;PRIORITY OF OR
9440:03         732          DFB  3            ;PRIORITY OF >
9441:03         733          DFB  3            ;PRIORITY OF =
9442:03         734          DFB  3            ;PRIORITY OF <
9443:           735 ;
9443:           736 ;   ERROR MESSAGE TEXT
9443:           737 ;
9443:           738 ERRTEXT EQU  *
0001:           739 TEXTEXP EQU  1
9443:54 45 58   740          DCI              "TEXT EXPECTED: "
9446:54 20 45
9449:58 50 45
944C:43 54 45
944F:44 3A A0
0002:           741 NUMVEXP EQU  2
9452:4E 55 4D   742          DCI              "NUMERIC VALUE EXPECTED"
9455:45 52 49
9458:43 20 56
945B:41 4C 55
945E:45 20 45
9461:58 50 45
9464:43 54 45
9467:C4
0003:           743 STRVEXP EQU  3
9468:53 54 52   744          DCI              "STRING VALUE EXPECTED"
946B:49 4E 47
946E:20 56 41
9471:4C 55 45
9474:20 45 58
9477:50 45 43
947A:54 45 C4
0004:           745 EXPREXP EQU  4
947D:45 58 50   746          DCI              "EXPRESSION EXPECTED"
9480:52 45 53
9483:53 49 4F
9486:4E 20 45
9489:58 50 45
948C:43 54 45
948F:C4
0005:           747 VAREXP  EQU  5
9490:56 41 52   748          DCI              "VARIABLE EXPECTED"
9493:49 41 42
9496:4C 45 20
9499:45 58 50
949C:45 43 54
949F:45 C4
0006:           749 DUPLSYM EQU  6
94A1:44 55 50   750          DCI              "DUPLICATED SYMBOL"
94A4:4C 49 43
94A7:41 54 45
94AA:44 20 53
94AD:59 4D 42
94B0:4F CC
0007:           751 KMBEGIN EQU  7
94B2:54 48 49   752          DCI              "THIS KEYWORD MAY NOT BEGIN A COMMAND"
94B5:53 20 4B
94B8:45 59 57
94BB:4F 52 44
94BE:20 4D 41
94C1:59 20 4E
94C4:4F 54 20
94C7:42 45 47
94CA:49 4E 20
94CD:41 20 43
94D0:4F 4D 4D
94D3:41 4E C4
0008:           753 MLDATA  EQU  8
94D6:4D 41 43   754          DCI              "MACHINE LANGUAGE DATA IGNORED"
94D9:48 49 4E
94DC:45 20 4C
94DF:41 4E 47
94E2:55 41 47
94E5:45 20 44
94E8:41 54 41
94EB:20 49 47
94EE:4E 4F 52
94F1:45 C4
0009:           755 LNUM64  EQU  9
94F3:4C 49 4E   756          DCI              "LINE NUMBER > 63999"
94F6:45 20 4E
94F9:55 4D 42
94FC:45 52 20
94FF:3E 20 36
9502:33 39 39
9505:89
000A:           757 UNDSTMT EQU  10
9506:55 4E 44   758          DCI              "UNDEFINED STATEMENT NUMBER"
9509:45 46 49
950C:4E 45 44
950F:20 53 54
9512:41 54 45
9515:4D 45 4E
9518:54 20 4E
951B:55 4D 42
951E:45 D2
000B:           759 KWVEXP  EQU  11
9520:4B 45 59   760          DCI              "KEYWORD OR VARIABLE EXPECTED"
9523:57 4F 52
9526:44 20 4F
9529:52 20 56
952C:41 52 49
952F:41 42 4C
9532:45 20 45
9535:58 50 45
9538:43 54 45
953B:C4
000C:           761 NOTWSTR EQU  12
953C:4F 50 45   762          DCI              "OPER NOT ALLOWED WITH STRINGS"
953F:52 20 4E
9542:4F 54 20
9545:41 4C 4C
9548:4F 57 45
954B:44 20 57
954E:49 54 48
9551:20 53 54
9554:52 49 4E
9557:47 D3
9559:           763 ;
9559:           764 ;   ADDITIONAL MESSAGE TEXT
9559:           765 ;
9558:           766 SYMBTXT EQU  *-1
0001:           767 TXLNUM  EQU  *-SYMBTXT
9559:4C 49 4E   768          DCI              "LINE NO"
955C:45 20 4E
955F:CF
0008:           769 TXRETN  EQU  *-SYMBTXT
9560:3C 52 45   770          DCI              "<RETURN>"
9563:54 55 52
9566:4E BE
0010:           771 TXVAR   EQU  *-SYMBTXT
9568:56 41 52   772          DCI              "VARIABLE"
956B:49 41 42
956E:4C C5
9570:           773 ;
9570:           774 CSUMEND EQU  *            ;END OF INTEGRITY-CHECKED AREA

--- SUCCESSFUL ASSEMBLY: NO ERRORS
```

END OF LISTING 1

```
          KEY PERFECT 5.0                  13015202    8F00 - 8F4F       28D6
             RUN ON                        08778543    8F50 - 8F9F       296F
           TYPE RIGHT                      CFFE1B5B    8FA0 - 8FEF       2504
==================================       C35AD0CE    8FF0 - 903F       21D1
CODE-5.0   ADDR# - ADDR#   CODE-4.0       37CBA54A    9040 - 908F       2DB1
--------   -------------   --------       EB577C8A    9090 - 90DF       2678
904D0EBC    8A00 - 8A4F        2684       2E58665C    90E0 - 912F       217A
25CB07FE    8A50 - 8A9F        2944       4897746C    9130 - 917F       2A0A
0FB7E9C7    8AA0 - 8AEF        25C0       EC264147    9180 - 91CF       2903
DF013030    8AF0 - 8B3F        29E1       6377BA15    9100 - 921F       270F
9A339A76    8B40 - 8B8F        264C       14DE0CFC    9220 - 926F       2663
7D677757    8B90 - 8BDF        291B       B8E88BD0    9270 - 92BF       224F
40E82F7C    8BE0 - 8C2F        2634       A918AA31    92C0 - 930F       29A4
75C6E117    8C30 - 8C7F        290D       5BAA38FE    9310 - 935F       26A3
57EC3A22    8C80 - 8CCF        2D9A       E3812629    9360 - 93AF       2404
F9AF1883    8CD0 - 8D1F        2688       E67A7488    9380 - 93FF       298C
CFA76A75    8D20 - 8D6F        28BF       77A87C14    9400 - 944F       27E8
2D5F7D85    8D70 - 8DBF        269F       58389C60    9450 - 949F       20AB
5D9B9224    8DC0 - 8E0F        2945       60063CB1    94A0 - 94EF       2749
63108E3F    8E10 - 8E5F        27F9       EA091454    94F0 - 953F       2BFA
99EB75A1    8E60 - 8EAF        22A7       2E90173F    9540 - 956F       1590
A6F09ABB    8EB0 - 8EFF        26C6       9B9F5ADC = PROGRAM TOTAL =    0B70
```