AUTO MENU

SECOND FEATURE

Make your menu *programming easier. Just enter the names of the menu items and the corresponding line ranges, and a fully functioning shell will be automatically programmed for you.*

Filet Mignon, Chicken Kiev, Veal Parmigiana, stuffed mushrooms, baked potato, cheesecake .... Whoops — that's from a different menu generator. *This* menu generator helps you get through with your programming so you can go out and enjoy what's on that other menu.

From the user's point of view, the menu can be the most important part of a program. It allows the user to move through the program, implementing specific choices with a minimum of effort, while at the same time checking that erroneous choices are not made. A good menu should give precise feedback so that the consequences of a specific menu choice are clear. After a menu item is chosen and a different area of the program is entered, where the user *is* and how to get back to where the user *was* should always be obvious.

Writing the menu structure is a big step in designing the whole program. Once the menus and submenus are established, the actual programming is clear — just fill in the blanks with your program code. These are the steps I follow when writing a program:

1. Get the idea for the program and think of a title
2. Set up the main menu choices
3. Set up the submenu choices
4. Map out the line numbers for each routine and subroutine
5. Assign variable names
6. Think through the algorithm for each routine and write down the main ideas or lines
7. Do the actual coding, run tests and debug

As you can see, the first four of these major problem-solving tasks involve the system of menus.

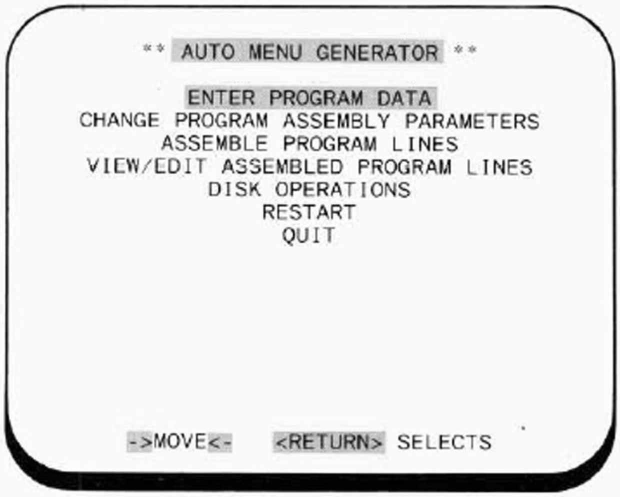
Automatic Menu Generator helps you complete these four problem-solving steps when writing a program. You enter the names of the menu and submenu options, and supply ranges for the line numbers. Automatic Menu Generator converts the information you supply into BASIC program lines that can be saved to disk.

USING THE PROGRAM

Let's step through a tutorial that sets up a menu for a database program. Automatic Menu Generator will help you organize your program and write the menu.

When AUTO.MENU (**Listing 1**) is first run, the main menu appears (**Figure 1**). Press the left and right, or up and down arrows to move the highlighted menu bar. To choose the highlighted menu option, press < RETURN >. (By the way, this is the same kind of menu that the program produces for you.)

FIGURE 1: Main Menu Display



```
            ** AUTO MENU GENERATOR **

               ENTER PROGRAM DATA
      CHANGE PROGRAM ASSEMBLY PARAMETERS
           ASSEMBLE PROGRAM LINES
       VIEW/EDIT ASSEMBLED PROGRAM LINES
              DISK OPERATIONS
                  RESTART
                   QUIT





       ->MOVE<-     <RETURN> SELECTS
```

Entering Program Data

Choose the ENTER PROGRAM DATA option. Immediately, MAIN MENU DATA ENTRY is displayed in inverse at the top of the screen. This identifies where you are in the program. The prompt:

```
ENTER MAIN MENU TITLE (2-32 CHARS) (Q)UIT
-->
```

appears. Type in the title you want for the main menu of your program. Verify your choice by typing **Y**. This is the first of several irreversible choices you will verify throughout the program. When you press <RETURN>, the menu title appears at the top of the screen, just as it will appear in the completed menu.

You then will see the prompts as shown in **Figure 2**. The first one asks you to enter your first main menu option. This is a database, so type ADD DATA TO FILE and press <RETURN>. Next, you are asked to identify:

```
LINE# TO GOSUB TO FOR OPTION 1 -->
```

All routines selected from menus are GOSUB routines. This does not, of course, preclude the use of GOTOs within a routine, but it does mean that when you want to exit from it, a RETURN statement must be executed. For this example, select **line 1000** for the start of your ADD DATA TO FILE routine and **line 2000** for the end.

The next question asks if you want a submenu for option 1. The ADD DATA TO FILE part of our database program is relatively straightforward, so a submenu is not needed. Press **N**. Enter Y to confirm your last entry and you should see three choices at the bottom of the screen:

```
<CR>CONTINUE, <SP>MENU COMPLETED, (Q)UIT:
```

Pressing **Q** would end data entry and pressing the space bar (SP) would indicate that your menu is completed. Instead, press <RETURN> to continue. When you do, your first menu option appears centered directly under the main menu title. In addition, the line number range you entered appears to the right of the first option, as well as a Y or N to indicate whether you selected the submenu option. These line numbers are included for your reference only; they will not appear in the final menu.

You will notice that the line number range ends at line 1999 instead of 2000. This allows your next main menu option to start at line 2000. (Only ending line numbers that are multiples of 100 are decremented by one. All others are left alone.) If the name of your menu option is quite lengthy, the end of it may appear to be cut off. Don't worry — this is done so that, while you're working, you can see the line number and submenu information on the right.

Enter the remaining main menu choices so that your display matches **Figure 3**. Note that three of the main menu options include submenus. Press the space bar to indicate that your main menu is complete.

Next, Automatic Menu Generator scans the main menu options to see if any of them require submenus. The first submenu request is in the SORT DATA option, so the top half of the screen displays the information in **Figure 4**. SORT DATA appears in inverse, centered as a heading for this first submenu. The program automatically enters a RETURN TO MAIN MENU option in this submenu, for obvious reasons.

As you enter data for your submenu, you can refer to the top of the screen to identify the submenu number, the submenu line number range and the corresponding main menu option. You must now decide what kind of submenu options will be available under the SORT DATA main menu option.

For example, we want to sort both alphabetically and numerically, so enter BEGIN ALPHABETICAL SORT as submenu option #2. The line numbers you enter for the submenu choice must be in the range 3000-3999. As the prompts appear, respond to them as shown in **Figure 5**. If you make a mistake and want to reenter the data for submenu option #2, respond **N** to the last prompt. The quit option ends data entry and returns you to AUTO.MENU's main menu. When your entry is confirmed as correct, the prompt:

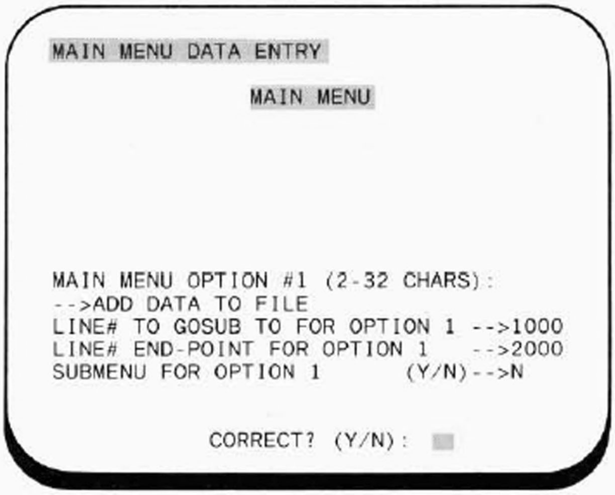
```
<CR>CONTINUE,<SP>MENU COMPLETED,<Q>UIT?
```

is displayed. Press <RETURN> since you have more submenu options to enter.

When you press <RETURN> your submenu option and the line numbers you specified appear under the RETURN TO MAIN MENU option. Type BEGIN NUMERICAL SORT as submenu option #3 and enter line numbers 3400-4000. This time, press the space bar, indicating that this submenu is complete.

At this point, the program checks your main menu options for another submenu to organize. In this case, there are two more requests for submenus. Go ahead and produce submenus for PRINT DATA and DISK OPTIONS on your own. Be sure to press <RETURN> at the end of each submenu option, and press the space bar when you are finished with a submenu. Unless you want the program to construct an incomplete menu, don't type Q to exit. Instead, use the space bar option for a normal exit. As you experi-

FIGURE 2: Main Menu Data Entry



```
MAIN MENU DATA ENTRY

              MAIN MENU





MAIN MENU OPTION #1 (2-32 CHARS):
-->ADD DATA TO FILE
LINE# TO GOSUB TO FOR OPTION 1 -->1000
LINE# END-POINT FOR OPTION 1   -->2000
SUBMENU FOR OPTION 1       (Y/N)-->N

          CORRECT? (Y/N):
```

ment, you will notice that the program does not accept certain line number ranges. In fact, a rather elaborate series of error checks is performed after each line number is entered.

When all of the submenus have been entered, the program returns to the main menu. At this point, the first four problem-solving steps in your database program have been accomplished.

Changing Parameters

The CHANGE PROGRAM ASSEMBLY PARAMETERS option allows you to set three different parameters that control parts of the assembly process. Pressing <RETURN> while the parameter is highlighted toggles the Y/N parameter switch. The effect of toggling the switch to Y is described below for each parameter:

Display Mode String — The menu string is displayed in the top-left corner of the screen.

Include Error Handling — An error handling routine is written into your menu program starting at **line 62000**. The appropriate ONERR statement is also included at the beginning of the program.

Equalize Menu String Lengths — This option pads the menu strings with spaces so that the inverse bar is the same length for all options.

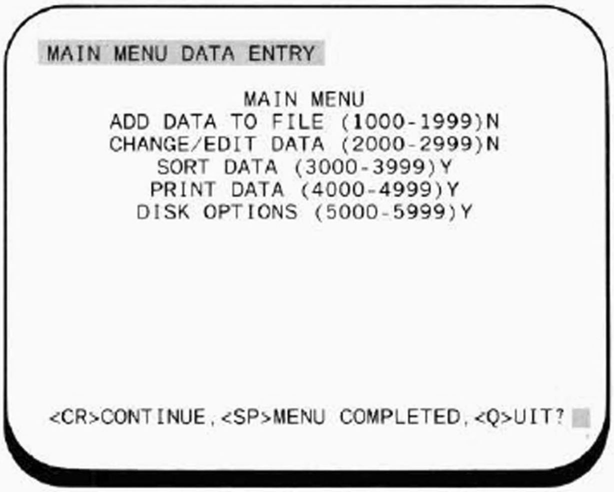
Assembling Your Program

This is the exciting part. When the ASSEMBLE PROGRAM LINES option is chosen, Automatic Menu Generator starts writing the program for which you have input parameters. The message ASSEMBLING BASIC PROGRAM LINES appears in the top-left corner of the screen. On the bottom line of the screen, PASS 1 appears. After a delay of several seconds, program lines start to scroll across the screen. There may be delays of 10 to 20 seconds as your Apple collects string garbage. During PASS 1, the program creates program lines that depend upon the data you have input. During PASS 2 these program lines are integrated with program lines that are not program specific and are simply read in from data statements located within Automatic Menu Generator itself. The assembly process takes up to three minutes, depending on the number of menu options you entered. When it is complete, you are returned to the main menu.

Editing Your Program

The VIEW/EDIT ASSEMBLED PROGRAM LINES option lets

FIGURE 3: Main Menu Construction Completed



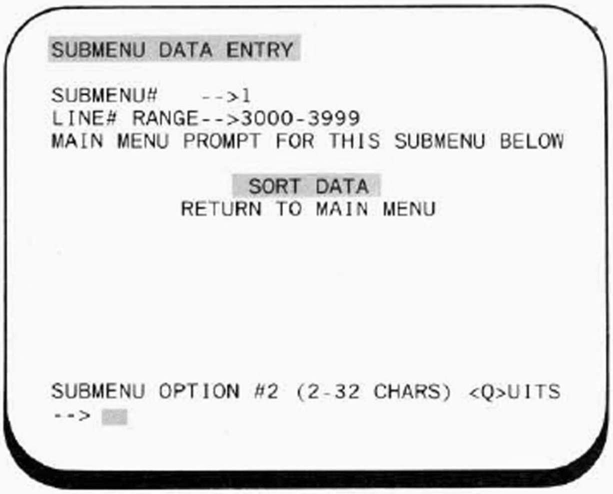
```
MAIN MENU DATA ENTRY

              MAIN MENU
    ADD DATA TO FILE (1000-1999)N
    CHANGE/EDIT DATA (2000-2999)N
        SORT DATA (3000-3999)Y
       PRINT DATA (4000-4999)Y
     DISK OPTIONS (5000-5999)Y




<CR>CONTINUE,<SP>MENU COMPLETED,<Q>UIT?
```

FIGURE 4: Submenu Data Entry



```
SUBMENU DATA ENTRY

SUBMENU#    -->1
LINE# RANGE-->3000-3999
MAIN MENU PROMPT FOR THIS SUBMENU BELOW

              SORT DATA
         RETURN TO MAIN MENU




SUBMENU OPTION #2 (2-32 CHARS) <Q>UITS
-->
```

you scroll through the program lines that have been assembled using the left and right arrow keys. If you decide to change any of the line numbers, press <RETURN>. This puts you into an edit mode that uses control key functions to edit program lines. The special function keys are displayed at the bottom of the screen. This useful Applesoft routine is similar to a routine by Robert J. Beck first published in Creative Computing (December, 1983). **Table 1** lists the control key functions. Notice the special use of the <RETURN> key.

When you have changed the line to your satisfaction, press <RETURN> (regardless of cursor position) to enter the modified line into memory. You are passed directly back to the scroll mode, as indicated by a message at the bottom of the screen. You may want to use this routine for viewing lines rather than for editing them, at least until you know what you would like to change. After you've had your fill of scrolling and editing, press Q to return to the main menu.

Saving Your Program

Choosing the DISK OPERATIONS option from the main menu takes you to a submenu. Choose the SAVE CREATED MENU TEXT FILE option. (Other options in this submenu will be explained later.) You are now prompted for the name of the file you want to save. Type DATABASE and press <RETURN>. The file DATABASE.TEXT is saved to disk. The .TEXT suffix is added regardless of whether you type it in or not to indicate that this file is a text file. When the file has been saved, you are returned to the DISK OPERATIONS submenu.

Testing Your Menu

You're finally ready for the moment of truth. Choose TEST RUN CREATED MENU from the DISK OPERATIONS submenu and respond to the ENTER FILE NAME TO TEST prompt by typing the file name, DATABASE. (You don't have to enter the .TEXT suffix to test run the file, although you can if you want.) After you press <RETURN>, the lines being read into memory from disk scroll across a window on the bottom of the screen. After the text file has been read, the Applesoft program DATABASE is saved to disk, and then is automatically run.

A menu for DATABASE that conforms to your specifications is displayed. The DATABASE program is now organized and all routines and subroutines have been defined. If you don't like the menu, you can load the text file back in and edit it. Now it's up to you to implement the menu selections by adding the necessary program lines. Follow problem-solving steps 5-7.

Loading Your Menu Text File — The LOAD CREATED MENU TEXT FILE option allows you to read any menu text file (like DATABASE.TEXT) into memory, view it, edit it and resave it to disk. In order to use this option, choose DISK OPERATIONS from the main menu, and choose LOAD CREATED MENU TEXT FILE from the submenu. The load process may seem lengthy. This is because AUTO.MENU is discarded while the file is loaded with a separate LOADER program (**Listing 2**). Then AUTO.MENU is reloaded.

RESTART and QUIT

The RESTART option reinitializes variables so that data can be entered for a new menu. Whenever data is entered for a menu and the operation is aborted, choose the RESTART option before entering data for a new menu.

The QUIT option checks to see whether you have saved a menu file for which data has been entered. If you haven't, you are prompted to do so. If you have saved a file, the program exits to Applesoft.

ENTERING THE PROGRAM

To key in Automatic Menu Generator, enter AUTO.MENU (**Listing 1**) and save it with the command:

```
SAVE AUTO.MENU
```

TABLE 1: Control Keys for Editing

Keypress	Function
<CTRL>B	Moves the cursor to the beginning of the entry
<CTRL>D	Deletes the character at the cursor position
<CTRL>FX	Finds the first occurrence of X (or any indicated character) to the right of the cursor
<CTRL>I	Inserts text at the cursor position
<CTRL>N	Jumps to the end of the entry
<CTRL>Q	Cuts off the entry at the cursor position
<CTRL>ZX	Zaps (deletes) all characters between the cursor position and the first X (or any indicated character) to the right of the cursor
<RETURN>	Enters the entire entry into memory (it does not cut it off at the cursor)

Next, type in **Listing 2** and save it to the same disk with the command:

```
SAVE LOADER
```

If you are using ProDOS, these two listings are all you need. If you're using DOS 3.3, insert your DOS 3.3 System Master and use FID to copy the file named CHAIN onto the disk that holds AUTO.MENU and LOADER. For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

HOW IT WORKS

Data Entry

The major variables for Automatic Menu Generator are listed in the program REM statements in **lines 50-150**. The most important ones are the arrays MM$(12,3) and SB$(12,2,6). MM$(12,3) contains information on the main menu options. The options themselves are found in the array MM$(N,0), where N is any number between 1 and 12. The other three sections of the array contain the beginning and ending line numbers and a Y or N to indicate whether a subroutine is required for that particular menu item. SB$(12,2,6) is a three-dimensional array that allows room for up to twelve subroutines with six options each, one for each main menu string. However, the program was written so that only six subroutines can be entered because of memory limitations. The array SB$(12,2,6) also holds the beginning and ending line numbers, which are accessed by the middle subscripts, 1 and 2, respectively. The actual submenu strings are found in SB$(N,0,M), where N specifies the submenu number and M specifies the option number. The menu and submenu data is stored in these two arrays during the data entry process.

Error Checking

Extensive error checking takes place during data entry. Most of these error routines are located in **lines 3250-3680**. **Table 2** gives the rules that Automatic Menu Generator enforces.

The array LN$(600) is key to the program's error checking. It holds all the legal line numbers up to 60000 that are divisible by 100. As soon as a line number is chosen and it passes tests 1-4 of **Table 2**, the array element with the subscript equal to the line number divided by 100 is set to null. Elements corresponding to all lines in the ranges specified by the user are also set to null.

Program Assembly

The first pass of the program assembly basically entails stringing together the data that has been entered and integrating the constructed program lines with lines that already exist in data statements within Automatic Menu Generator. In **lines 1270-1300**, the main menu data line is constructed with a maximum of seven data items (the main menu options) in each line before the line number counter, LN, is incremented by 10. The process repeats in **lines 1310-1390** where all of the menu strings are placed in one data line for each subroutine. Notice that RETURN TO MAIN MENU is inserted in each subroutine data statement. Next, the GOSUB command lines for the menu and submenus are assembled. The construction of these lines depends on the starting and ending lines supplied by the user for each submenu. The REM statements for each subroutine and the ending RETURN and REM statements are constructed in **lines 1620-1690**.

During the second pass, the already assembled lines that have been stored in LN$(600) (to save memory) are integrated with lines read from data statements found in **lines 1890-2250**. This routine is constructed so that the line numbers are assembled more or less in order and are eventually placed in the array PG$(214).

TABLE 2: Rules Used in Menu Construction

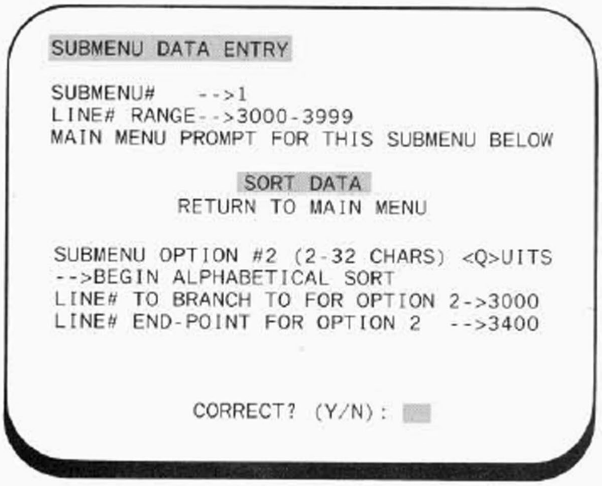
1. Number of main menu options: 2-12.
2. Number of submenus: 0-6.
3. Number of submenu options: 3-7.
4. Character length of menu options: 2-32.
5. Menu options must be in upper-case only.
6. Menu options must be distinct within a given menu.
7. Line number range for user's code: 700-59999.
8. Line number range for main menu routines: 100-9999.
9. A line number separation of 299 is necessary for a submenu to be possible.
10. The beginning line numbers for the main menu and the subroutines must be divisible by 100.
11. No two subroutines can occupy the same line number range.
12. The ending line number may be any number (subject to conditions 7-11 above), but if it is divisible by 100, then 1 is subtracted from it.

Saving and Testing

Since the program exists in the form of a string array when assembly is finished, the easiest way to create an Applesoft file is to save the array as a text file and then EXEC it into memory. This is exactly what is done. Before the program can be test run, a text file version of the program is saved to disk. When you choose the TEST RUN submenu option, the following steps occur (**lines 2980-3110**):

1. A small text file called IF is opened.
2. The appropriate commands to NEW and EXEC the previously saved text file version of the program are inserted into it.
3. AUTO.MENU EXECs IF.
4. The text file version of your menu program is entered into memory. As each line is entered it becomes part of the Applesoft version of your program.
5. When the text file version of the file was originally saved to disk, the commands HOME, DELETE IF, SAVE and RUN were inserted at the end. Consequently, after your program is EXECed into memory, the screen clears and the little EXEC file is deleted. The program then saves itself under the proper name and runs.

FIGURE 5: Submenu Data Entry Confirmation



```
SUBMENU DATA ENTRY

SUBMENU#     -->1
LINE# RANGE-->3000-3999
MAIN MENU PROMPT FOR THIS SUBMENU BELOW

                  SORT DATA
          RETURN TO MAIN MENU

SUBMENU OPTION #2 (2-32 CHARS) <Q>UITS
-->BEGIN ALPHABETICAL SORT
LINE# TO BRANCH TO FOR OPTION 2->3000
LINE# END-POINT FOR OPTION 2   -->3400


           CORRECT? (Y/N):
```

The Menu

The menu program created by Automatic Menu Generator is in itself a program worthy of note. Besides the nicely organized REM statements it creates to guide the placement of your code, it does an admirable job of handling the flow from one menu to another.

The key lines to this menu are the main menu and submenu GOSUB command lines starting at line 560. The variable ST identifies at what position in the data stack a particular menu's strings lie. This information is processed by line 290, which advances the DATA pointer to the proper item, if necessary. (The data statements containing the menu and submenu strings are the first ones in the program. You must take this into consideration if your own program has data statements in it. Simply perform a priming READ similar to the one in line 290 to get the data pointer to the proper position to read your own data.) SN contains the number of strings to read into MENU$(12) in line 310. Within the READ loop the center tab positions are located and stored in TB(12). VE passes to all subsequent PRINT statements the vertical tab position for the menu items. When these variables have been set, the menu is printed on the screen for the first time (line 330) and the first menu item is highlighted. The keyboard is then checked for one of several characters (line 370) and the appropriate action is taken. If the character is a RETURN, then the vertical position of the cursor is determined, stored in X, and control returns to the command line where the ON X GOSUB statement is executed. Note that the command line ends with a GOTO statement, which sends control back to itself. If one of the main menu options has a submenu option, the ON X GOSUB statement sends control to the command line for the subroutine. Each subroutine has its own command line. The first item on each subroutine menu is, of course, RETURN TO MAIN MENU. To accomplish this, each subroutine command line contains ON X GOSUB 690 as a first option because this line contains the necessary POP and RETURN.

Menus created by Automatic Menu Generator have a common structure. Lines 100-160 are documentation. Lines 170-280 (depending on the number of subroutines) contain data statements. Lines 290-550 are the body of the menu program. Line 560 displays the main menu command line. Lines 570-680 handle submenu commands if necessary. Line 690 POPs and RETURNs to the main menu command line. Lines 700 and up contain your implementation of the menu options.

The Chaining Mechanism

In order to avoid excessive delays due to garbage collection, the program uses a chain to a separate, short LOADER program. First the name is saved in a short text file so that a CLEAR can be performed before the loader is chained in using the CHAIN program from the DOS 3.3 System Master Disk. The loader program reads the file name from the text file, and then loads the text file into the array PG$( ). This array is preserved when the main program is chained back in.

Under ProDOS, the separate CHAIN program is not necessary, since the ProDOS CHAIN command works so well. Where necessary, ProDOS is checked for by testing the contents of address 48896. If a value of 76 is found, the program is operating under ProDOS and the CHAIN command is issued instead of the LOAD command.

Auto Menu, Imagewriter and Scribe Control, Double Lo-Res Graphics From BASIC and Double Hi-Res Characters are available on diskette for an introductory price of $17.95 plus $1.50 shipping/handling ($2.50 outside the U.S.) from Nibble, 45 Winthrop St., Concord, MA 01742. Introductory price expires 4/30/86.

LISTING 1: AUTO.MENU

```
1   REM ************************
2   REM *       AUTO.MENU      *
3   REM * BY KENNETH PENNER    *
4   REM * COPYRIGHT (C) 1986   *
5   REM * BY MICROSPARC, INC   *
6   REM * CONCORD, MA   01742  *
7   REM ************************
50  TEXT : HOME : VTAB 10: HTAB 11: PRINT "AU
    TO MENU GENERATOR": VTAB 12: HTAB 12: PRINT
    "BY KENNETH PENNER": VTAB 14: PRINT "**
    COPYRIGHT 1986 BY MICROSPARC, INC **": VTAB
    22: HTAB 7: PRINT "PRESS <RETURN> TO CON
    TINUE";: GET Z$: PRINT
60  ONERR  GOTO 3770
70  IF C1 > 1 THEN 480: REM  INITIALIZATION A
    LREADY TAKEN PLACE
80  HOME : GOTO 3700: REM   INITIALIZE
90  REM  THROW-AWAY STRING AND COUNTER VARIAB
    LE -A$,B$,C$,F$,C,C1,C2
100  REM   FLAGS ER=ERROR OCCURED, DA=DATA EN
     TERED, AS=ASSEMBLY COMPLETED, SV=FILE SA
     VED, LD=FILE LOADED
110  REM  D$=CHR$(4),NS=NUMBER OF SUBROUTINES
     , MD$=MODE INDICATOR STRING, PR$=PROMPT
     MESSAGE STRING
120  REM   N1+NS=COUNTERS FOR # OF SUBROUTINE
     S PROCESSED, N=COUNTER FOR NUMBER OF STR
     INGS IN MENU
```

LISTING 1: AUTO.MENU *(continued)*

```
130  REM     SB$(12,2,7)-HOLDS SUBROUTINE DAT
     A, MM$(12,3)-HOLDS MAIN MENU DATA, PG$(1
     5Ø)-HOLDS FINAL PROGRAM LINES FROM PASS
     2
140  REM   LN$(6ØØ) HOLDS ALL LEGAL LINE NUMB
     ERS POSSIBLE FOR ERROR CHECKING PURPOSES
      AND ALSO ASSEMBLED DATA LINES FROM PASS
      1
150  REM   LL$ AND LL HOLD LOWER LINE# LIMIT
     FOR GIVEN SUBROUTINE,UL$ AND UL HOLD UPP
     ER LINE# LIMIT FOR GIVEN SUBROUTINES
160  REM  MENU VARIABLES- T$=TITLE:SD=SOUNDFL
     AG:MENUS$(N)=MENUSTRINGS:TB(N)=TABS:ST=ST
     ART POSITION OF MENUSTRINGS IN DATA LIST
     :VE=VERTICALTABPOSITION:SN=# OF MENU CHO
     ICES:X=VERTICAL CURSOR POSITION
170 SD = 1: GOTO 48Ø: REM   MAIN MENU CONTROL
     LINE
180  DATA        AUTO MENU GENERATOR,ENTER PRO
     GRAM DATA,CHANGE PROGRAM ASSEMBLY PARAME
     TERS,ASSEMBLE PROGRAM LINES, VIEW/EDIT A
     SSEMBLED PROGRAM LINES,DISK OPERATIONS,R
     ESTART,QUIT
190  DATA        RETURN TO MAIN MENU, CATALOG,
     LOAD CREATED MENU TEXT FILE,SAVE CREATED
      MENU TO TEXT FILE, TEST RUN CREATED MEN
     U
200  DATA     RETURN TO MAIN MENU, DISPLAY MOD
     E STRING,INCLUDE ERROR HANDLING,EQUALIZE
      MENU STRING LENGTHS
210  RESTORE : IF ST > 1 THEN  FOR C = 1 TO S
     T: READ A$: NEXT : REM   PRIMING READ TO
      GET TO PROPER DATA
220  READ T$
230  FOR C = 1 TO SN: READ MENUS$(C):TB(C) =  INT
     (2Ø -  LEN (MENUS$(C)) / 2): NEXT : RETURN

240  REM   INITIAL PRINTING OF MENU ROUTINE
250  HOME : HTAB  INT (17 - ( LEN (T$) / 2)):
      PRINT "** ";: INVERSE : PRINT T$;: NORMAL
     : PRINT " **";
260  NORMAL : VTAB 24: HTAB 7: INVERSE : PRINT
     "->";: NORMAL : PRINT "MOVE";: INVERSE :
      PRINT "<-";: NORMAL : PRINT "    ";: INVERSE
     : PRINT "<RETURN>";: NORMAL : PRINT " SE
     LECTS";
270  HTAB TB(1): VTAB VE: FOR C = 1 TO SN: HTAB
     TB(C): PRINT MENUS$(C): NEXT : VTAB VE: HTAB
     TB(1): INVERSE : PRINT MENUS$(1);: NORMAL

280  REM  GET AND PROCESS KEY PRESS
290  POKE  - 16368,Ø: GET A$: GOSUB 44Ø:X =  PEEK
     (37) + 1
300  IF A$ =  CHR$ (8) OR A$ =  CHR$ (11) THEN
      GOSUB 35Ø: REM    BACKWARD OR UP ARROW
     REVERSES MENU SELECTION
310  IF A$ =  CHR$ (21) OR A$ =  CHR$ (32) OR
     A$ =  CHR$ (1Ø) THEN  GOSUB 37Ø: REM
     FORWARD OR DOWN ARROW OR SPACE BAR ADVAN
     CES MENU SELECTION
320  IF A$ =  CHR$ (13) THEN X =  PEEK (37) -
     VE + 2: PRINT : RETURN : REM   DETERMINE
      CHOICE BY CHECKING VERTICAL TAB POSITIO
     N
330  GOTO 29Ø
340  REM  TAB CONTROL ROUTINE
350  IF X = VE THEN  VTAB X: GOSUB 4ØØ:X = VE
      + SN - 1: GOSUB 41Ø: INVERSE : GOSUB 42
     Ø: RETURN
360  VTAB X: GOSUB 4ØØ:X = X - 1: INVERSE : GOSUB
     41Ø: GOSUB 42Ø: RETURN
370  IF X = SN + VE - 1 THEN  VTAB SN + VE -
     1: GOSUB 4ØØ:X = VE: GOSUB 41Ø: INVERSE
     : GOSUB 42Ø: RETURN
380  VTAB X: GOSUB 4ØØ:X = X + 1: INVERSE : GOSUB
     41Ø: GOSUB 42Ø: RETURN
390  REM    SUBROUTINE WHICH PRINTS PROPER ME
     NU STRING IN INVERSE DEPENDING ON KEY PR
     ESS
400  HTAB TB(X - VE + 1): PRINT MENUS$(X - VE +
     1);: RETURN
410  VTAB X: HTAB TB(X - VE + 1): RETURN
420  PRINT MENUS$(X - VE + 1);: NORMAL : RETURN

430  REM      ROUTINE TO PRODUCE SOUND
440  IF A$ =  CHR$ (83) THEN SD = (SD = Ø)
450  IF SD THEN S =  - 16336:A =  PEEK (S) -
      PEEK (S) +  PEEK (S) -  PEEK (S) +  PEEK
     (S)
460  RETURN
470  REM    MAIN MENU
480  POKE 33,4Ø:ST = 1:SN = 7:VE = 3: GOSUB 2
     1Ø: GOSUB 25Ø: ON X GOSUB 54Ø,5ØØ,122Ø,2
     27Ø,49Ø,4Ø2Ø,314Ø: POKE 34,Ø: HOME : GOTO
     48Ø
490 ST = 7:SN = 5:VE = 16: GOSUB 21Ø: GOSUB 2
     6Ø: ON X GOSUB 52Ø,266Ø,286Ø,27ØØ,299Ø: POKE
     34,Ø: HOME : GOTO 49Ø
500 ST = 12:SN = 4:VE = 16: POKE 33,38: GOSUB
     399Ø: GOSUB 21Ø: GOSUB 26Ø: ON X GOSUB 5
     2Ø,398Ø,398Ø,398Ø: POKE 34,Ø
510  GOSUB 29Ø: GOSUB 398Ø: ON X GOTO 48Ø,51Ø
     ,51Ø,51Ø
520  POP : POKE 33,4Ø: RETURN
530  REM   ENTER PROGRAM DATA ROUTINE
540  IF N > Ø OR N1 > Ø THEN PR$ = "MUST  RES
     TART FIRST: ": GOSUB 322Ø: POKE  - 16368
     ,Ø: GET A$: RETURN
550 N = 1:N1 = Ø
560  HOME :MD$ = "MAIN MENU DATA ENTRY": GOSUB
     318Ø: POKE 34,16
570  HOME : VTAB 17: PRINT "MAIN MENU TITLE (
     2-32 CHARS) (Q)UITS": INPUT "-->";B$: IF
     B$ = "Q" THEN  RETURN
580  GOSUB 354Ø: ON  LEN (B$) > Ø GOSUB 365Ø:
      IF ER THEN ER = Ø: GOTO 57Ø
590  GOSUB 321Ø: IF A$ = "N" THEN 57Ø
600  IF A$ <  > "Y" THEN 59Ø
610  INVERSE :MM$(N,Ø) = B$: GOSUB 319Ø: NORMAL

620 UL = Ø:LL = Ø: HOME : VTAB 17: PRINT "MAI
     N MENU OPTION #"N" (2-32 CHARS):": INPUT
     "-->";B$
630  GOSUB 352Ø: GOSUB 365Ø: IF ER THEN ER =
     Ø: GOTO 62Ø
640  HTAB 1: VTAB 19: PRINT "LINE# TO GOSUB T
     O FOR OPTION "N;: INPUT "-->";LL$:LL =  VAL
     (LL$)
650  GOSUB 326Ø: IF ER THEN ER = Ø: GOTO 64Ø:
      REM    ERROR CHECKING
660  HTAB 1: VTAB 2Ø: PRINT "LINE# END-POINT
     FOR OPTION "N;: INPUT "  -->";UL$:UL =  VAL
     (UL$): IF UL / 1ØØ =  INT (UL / 1ØØ) THEN
     UL = UL - 1:UL$ =  STR$ (UL)
670  GOSUB 326Ø: IF ER THEN ER = Ø: GOTO 66Ø:
      REM  ERROR CHECK
680  GOSUB 336Ø: IF ER THEN ER = Ø:C$ = "N": GOTO
     7ØØ
690  IF  NOT ER THEN  HTAB 1: VTAB 21: PRINT
     "SUBMENU FOR OPTION "N;: INPUT "      (Y/
     N)-->";C$: IF C$ <  > "Y" AND C$ <  > "N
     " THEN 69Ø
700  GOSUB 321Ø: IF A$ = "N" THEN 62Ø
710  IF A$ <  > "Y" THEN 7ØØ
720 N = N + 1:MM$(N,Ø) = B$:MM$(N,1) = LL$:MM
     $(N,2) = UL$: IF C$ = "Y" THEN MM$(N,3) =
     "SB":NS = NS + 1
730 B$ = B$ + " (" + LL$ + "-" + UL$ + ")" +
     C$: IF  LEN (B$) > 36 THEN B$ = ".." +  RIGHT$
     (B$,36)
740  GOSUB 319Ø
750  FOR C = LL TO UL STEP 1ØØ:LN$(C / 1ØØ) =
     "": NEXT : REM       CLEAR OUT LINE # PO
     SSIBILITIES
760  HOME :PR$ = "<CR>CONTINUE,<SP>MENU COMPL
     ETED,(Q)UIT": GOSUB 322Ø
770  POKE  - 16368,Ø: GET A$:MM$(Ø,Ø) =  STR$
     (N)
780  IF A$ = "Q" OR (A$ =  CHR$ (32) AND N =
     2) THEN  GOSUB 338Ø: GOTO 76Ø
790  IF A$ =  CHR$ (32) THEN  GOTO 82Ø
800  IF A$ =  CHR$ (13) AND N < 13 THEN  GOTO
     62Ø
810  GOTO 76Ø
820 PR$ = "MAIN MENU COMPLETED, PRESS A KEY:
     ": GOSUB 322Ø: POKE  - 16368,Ø: GET A$
830  IF NS = Ø THEN 119Ø
840 N = 1:SB = 1:N1 = 1: REM      N=SUBMENU#,
      N1=# OF STRINGS IN SUBMENU
```

LISTING 1: AUTO.MENU *(continued)*

```
850  POKE 34,0: HOME :MD$ = "SUBMENU DATA ENT
     RY": GOSUB 3180
860  POKE 34,1: HOME :SB = SB + 1: FOR C = SB
      TO 13: IF MM$(C,3) = "SB" THEN SB = C:C
      = 13
870  NEXT
880 L1 =  VAL (MM$(SB,1)):L2 =  VAL (MM$(SB,2
     ))
890  FOR C = L1 TO L2 STEP 100:LN$(C / 100) =
      STR$ (C): NEXT
900  HTAB 1: VTAB 3: PRINT "SUBMENU#    -->"N:
      PRINT "LINE# RANGE-->"MM$(SB,1)"-"MM$(S
     B,2): PRINT "MAIN MENU PROMPT FOR THIS S
     UBMENU BELOW": VTAB 7: INVERSE :B$ = MM$
     (SB,0): GOSUB 3230: NORMAL
910  HTAB 1: VTAB 8:B$ = "RETURN TO MAIN MENU
     ": GOSUB 3230
920  POKE 34,16:UL = 0:LL = 0: HOME : VTAB 17
     : PRINT "SUBMENU OPTION #"N1 + 1" (2-32)
      CHARS (Q)UITS": INPUT "-->";B$: IF B$ =
     "Q" THEN PR$ = "SUBMENU INCOMPLETE,QUIT
     ANYWAY? (Y/N)": GOSUB 3220: POKE  - 1636
     8,0: GET A$: IF A$ = "Y" THEN NS = N - (
     N1 = 1): GOTO 1190
930  GOSUB 3540: IF ER THEN ER = 0: GOTO 920
940  GOSUB 3590: GOSUB 3650: IF ER THEN ER =
     0: GOTO 920
950  HTAB 1: VTAB 19: PRINT "LINE# TO BRANCH
     TO FOR OPTION "N1 + 1;: INPUT "->";LL$:L
     L =  VAL (LL$)
960  GOSUB 3440: IF ER THEN ER = 0: GOTO 950:
      REM      ERROR CHECKING
970  HTAB 1: VTAB 20: PRINT "LINE# END-POINT
     FOR OPTION "N1 + 1;: INPUT "  -->";UL$:U
     L =  VAL (UL$): IF UL / 100 =  INT (UL /
     100) THEN UL = UL - 1:UL$ =  STR$ (UL)
980  GOSUB 3440: IF ER THEN ER = 0: GOTO 970:
      REM    ERROR CHECK
990  GOSUB 3210: IF A$ = "N" THEN 920
1000  IF A$ <  > "Y" THEN 990
1010 SB$(N,0,N1) = B$:SB$(N,1,N1) = LL$:SB$(N
     ,2,N1) = UL$:SB$(N,0,0) =  STR$ (N1):N1 =
     N1 + 1
1020 B$ = B$ + " (" + LL$ + "-" + UL$ + ")": IF
      LEN (B$) > 36 THEN B$ = ".." +  RIGHT$
     (B$,36)
1030  GOSUB 3240
1040  FOR C = LL TO UL STEP 100:LN$(C / 100) =
     "": NEXT : REM           CLEAR OUT LINE #
      POSSIBLITIES
1050 C1 = 0: FOR C = L1 TO L2 STEP 100: IF LN
     $(C / 100) <  > "" THEN C1 = C1 + 1: REM
      CHECK TO SEE IF ANY LINE NUMBERS LEFT
1060  NEXT : IF C1 = 0 THEN A$ =  CHR$ (32): GOTO
     1090
1070  HOME :PR$ = "<CR>CONTINUES,<SP>MENU COM
     PLETED,(Q)UIT": GOSUB 3220
1080  POKE  - 16368,0: GET A$
1090  IF A$ =  CHR$ (32) AND N1 > 1 THEN N =
     N + 1:N1 = 1: GOTO 1140
1100  IF A$ = "Q" THEN  GOSUB 3380: GOTO 1070
1110  IF A$ =  CHR$ (13) THEN 1140
1120  GOTO 1080
1130  REM  CHECK TO SEE IF ANY MORE SUBMENUS
1140  IF N =  > NS + 1 OR (A$ =  CHR$ (13) AND
     N = NS AND N1 > 6) THEN 1190
1150  IF N > 13 THEN 1190
1160  IF N1 > 6 THEN N = N + 1:N1 = 1: GOTO 1
     180
1170  IF A$ =  CHR$ (13) THEN 920
1180 PR$ = "ADVANCING TO NEXT SUBMENU-PRESS <
     CR>: ": GOSUB 3220: POKE  - 16368,0: GET
     A$: GOTO 860
1190 PR$ = "SUBMENUS COMPLETE-PRESS <CR>: ": GOSUB
     3220: POKE  - 16368,0: GET A$:DA = 1:SB$
     (0,0,0) =  STR$ (NS): RETURN
1200  REM    ASSEMBLE BASIC LINES ROUTINE
1210  REM    ASSEMBLE DATA LINES FOR MAIN MEN
     U FIRST
1220  GOSUB 3610: IF ER THEN ER = 0: RETURN
1230  IF AS OR LD THEN PR$ = "ALREADY ASSEMBL
     ED-PRESS <CR>: ": GOSUB 3220: POKE  - 16
     368,0: GET A$: RETURN
1240  HOME :MD$ = "ASSEMBLING BASIC PROGRAM L
     INES": GOSUB 3180:PR$ = "...PASS 1...": INVERSE
     : GOSUB 3220: NORMAL : POKE 34,3: POKE 3
     5,22:LN = 160: HTAB 1: VTAB 4
1250  IF AP(3) THEN  GOSUB 3830
1260  FOR C = 1 TO 600:LN$(C) = "": NEXT
1270  FOR C = 1 TO  VAL (MM$(0,0))
1280 F$ = F$ +  CHR$ (34) + MM$(C,0) +  CHR$
     (34) + ", "
1290  IF C / 7 =  INT (C / 7) OR C =  VAL (MM
     $(0,0)) THEN LN$((LN - 60) / 10 - 9) =  STR$
     (LN) + " DATA " +  LEFT$ (F$, LEN (F$) -
     2): PRINT LN$((LN - 60) / 10 - 9):F$ = "
     ":LN = LN + 10
1300  NEXT
1310  REM  ASSEMBLE DATA LINES FOR SUBROUTINE
     S IF ANY
1320  IF NS = 0 THEN 1410: REM   SKIP TO MENU
      GOSUB LINE ASSEMBLY ROUTINE
1330  FOR C = 1 TO NS
1340  GOSUB 4040
1350  FOR C1 = 1 TO  VAL (SB$(C,0,0))
1360 F$ = F$ +  CHR$ (34) + SB$(C,0,C1) +  CHR$
     (34) + ", "
1370  NEXT C1
1380 LN$((LN - 60) / 10 - 9) =  STR$ (LN) + "
      DATA " +  LEFT$ (F$, LEN (F$) - 2) + ":
     REM MENU DATA LINE FOR SUBROUTINE #" +  STR$
     (C): PRINT LN$((LN - 60) / 10 - 9):F$ =
     "RETURN TO MAIN MENU, ":LN = LN + 10
1390  NEXT C
1400  REM  MAIN MENU GOSUB LINE ASSEMBLY
1410 A$ = ":GOSUB290:GOSUB330:ON X GOSUB":SB =
     1
1420 LN = 560:F$ =  STR$ (LN) + " ST=1:SN=" +
      STR$ ( VAL (MM$(0,0)) - 1) + ":VE=3" +
     A$
1430  FOR C = 2 TO  VAL (MM$(0,0))
1440  IF MM$(C,3) = "SB" THEN F$ = F$ +  STR$
     (560 + 10 * SB) + ",":SB = SB + 1: GOTO
     1460
1450 F$ = F$ + MM$(C,1) + ","
1460  NEXT
1470 F$ =  LEFT$ (F$, LEN (F$) - 1) + "
1480 LN$(20) = F$ + ":POKE34,0:HOME:GOTO560":
      PRINT LN$(20):LN = LN + 10:F$ = ""
1490  IF NS = 0 THEN 1620: REM     START INTE
     GRATING PROGRAM LINES
1500  REM  SUB-MENU GOSUB LINE ASSEMBLY ROUTI
     NE
1510 A$ = ":VE=16:GOSUB290:GOSUB340:ON X GOSU
     B 690,":ST$ =  STR$ ( VAL (MM$(0,0)) - 1
     )
1520  FOR C = 1 TO NS: REM  # OF SUBROUTINES
1530 F$ = " ST=" + ST$ + ":SN=" +  STR$ ( VAL
     (SB$(C,0,0)) + 1) + A$
1540  FOR C1 = 1 TO  VAL (SB$(C,0,0))
1550 F$ = F$ + SB$(C,1,C1) + ","
1560  NEXT
1570 F$ =  LEFT$ (F$, LEN (F$) - 1) + ":POKE3
     4,0:HOME:GOTO" +  STR$ (LN)
1580 LN$(20 + C) =  STR$ (LN) + " " + F$: PRINT
     LN$(20 + C):LN = LN + 10
1590 ST$ =  STR$ ( VAL (ST$) +  VAL (SB$(C,0,
     0)) + 1)
1600  NEXT C
1610 LN$(49) = "690 POP:RETURN": PRINT LN$(49
     )
1620  REM  ASSEMBLE MAIN MENU SUBROUTINE STAR
     T AND END POINT LINES
1630 C1 = 0: FOR C = 2 TO  VAL (MM$(0,0))
1640  IF MM$(C,3) = "SB" THEN 1690
1650 LN$(50 + C1) = MM$(C,1): IF AP(1) THEN L
     N$(50 + C1) = LN$(50 + C1) + "POKE34,0:H
     OME:INVERSE:PRINT" +  CHR$ (34) + MM$(C,
     0) +  CHR$ (34) + ":NORMAL:GETA$:"
1660 LN$(50 + C1) = LN$(50 + C1) + "REM **" +
     MM$(C,0) + "** MAIN MENU ROUTINE BEGINS
     HERE AND ENDS AT LINE " + MM$(C,2): PRINT
     LN$(50 + C1)
1670 LN$(50 + C1 + 1) = MM$(C,2) + " RETURN:R
     EM END OF **" + MM$(C,0) + "** MAIN MENU
      ROUTINE": PRINT LN$(50 + C1 + 1)
1680 C1 = C1 + 2
1690  NEXT C:C2 = 0: IF NS = 0 THEN 1770
1700  REM  ASSEMBLE SUBROUTINE START AND END
     POINT LINES
```

LISTING 1: AUTO.MENU *(continued)*

```
1710  FOR C = 1 TO NS: FOR C1 = 1 TO  VAL (SB
     $(C,Ø,Ø))
1720 LN$(7Ø + C2) = SB$(C,1,C1): IF AP(1) THEN
     LN$(7Ø + C2) = LN$(7Ø + C2) + "POKE34,Ø:
     HOME:INVERSE:PRINT" +  CHR$ (34) + SB$(C
     ,Ø,C1) +  CHR$ (34) + ":NORMAL:POKE-1636
     8,Ø:GET A$:"
1730 LN$(7Ø + C2) = LN$(7Ø + C2) + "REM **" +
     SB$(C,Ø,C1) + "** SUBMENU ROUTINE BEGINS
      HERE AND ENDS AT LINE " + SB$(C,2,C1): PRINT
     LN$(7Ø + C2)
1740 LN$(7Ø + C2 + 1) = SB$(C,2,C1) + "RETURN
     : REM END OF **" + SB$(C,Ø,C1) + "** SUB
     MENU ROUTINE": PRINT LN$(7Ø + C2 + 1)
1750 C2 = C2 + 2: NEXT C1: NEXT C
1760  REM     INTEGRATE LINES FROM DATA AND GE
     NERATED PROGRAM LINES
1770 PR$ = "...PASS 2...": INVERSE : GOSUB 32
     2Ø: NORMAL : HOME : HTAB 1: VTAB 4
1780  RESTORE : FOR C = 1 TO 17: READ A$: NEXT
     : REM                  PRIMING READ
1790  FOR C1 = 1 TO 6: READ PG$(C1): PRINT PG
     $(C1): NEXT :C1 = 7: REM   READ LINES 1Ø
     Ø-15Ø OF AUTO MENU GENERATOR AND PLACE I
     N PG$(1-6)
1800  IF AP(2) THEN PG$(1) = "1ØØ ONERR GOTO6
     2Ø1Ø:" +  RIGHT$ (PG$(1), LEN (PG$(1)) -
     3)
1810  FOR C = 1 TO 19: IF  LEN (LN$(C)) > 1 THEN
     PG$(C1) = LN$(C):LN$(C) = "": PRINT PG$(
     C1):C1 = C1 + 1: REM          PLACE ASSEM
     BLED DATA LINE) IN PG$(7-N)
1820  NEXT
1830  FOR C = 1 TO 27: READ PG$(C1):C1 = C1 +
     1: IF C = 6 OR C = 5 THEN A$ = PG$(C1 -
     1): GOSUB 4ØØØ:PG$(C1 - 1) = A$
1840  PRINT PG$(C1 - 1): NEXT : REM        READ
      MAIN PROGRAM BODY INTO PG$(N)
1850  IF AP(2) THEN  FOR C = 215 TO 221: READ
     LN$(C):A$ = LN$(C): GOSUB 4ØØØ:LN$(C) =
     A$: NEXT
1860  FOR C = 2Ø TO 225: IF  LEN (LN$(C)) > 1
      THEN PG$(C1) = LN$(C):LN$(C) = "": PRINT
     PG$(C1):C1 = C1 + 1: REM                P
     LACE ASSEMBLE GOSUB LINES AND BEGIN END
     POINTS INTO PG$(N)
1870  NEXT
1880 AS = 1:PR$ = "PROGRAM ASSEMBLED IN MEMOR
     Y-PRESS<CR>:": GOSUB 322Ø: POKE  - 16368
     ,Ø: GET A$:NL = C1 - 1: POKE 35,24: RETURN

1890  DATA       "1ØØ REM ******************
     ********","11Ø REM * CREATED USING AUTO.
     MENU *","12Ø REM *      BY KEN PENNER
        *","13Ø REM ************************
     **"
1900  DATA  "14Ø REM T$=TITLE:SD=SOUNDFLAG:ME
     NU$(N)=MENUSTRINGS:TB(N)=TABS:ST=START P
     OSITION OF MENUSTRINGS IN DATA LIST:VE=V
     ERTICALTABPOSITION:SN=# OF MENU CHOICES:
     X=VERTICAL CURSOR POSITION"
1910  DATA       "15Ø DIM TB(12),ME$(12):SD=1:
     D$=CHR$(4): GOTO 56Ø: REM MAIN MENU GOSU
     B LINE IS LOCATED AT 56Ø"
1920  DATA  "29Ø RESTORE : IF ST > 1 THEN  FO
     R C = 1 TO ST: READ A$: NEXT : REM  PRIM
     ING READ TO GET TO PROPER DATA"
1930  DATA  "3ØØ  READ T$"
1940  DATA "31Ø FOR C = 1 TO SN: READ MENU$(C
     ):TB(C) =  INT (2Ø -  LEN (MENU$(C)) / 2
     ): NEXT : RETURN "
1950  DATA  "32Ø REM   INITIAL PRINTING OF ME
     NU ROUTINE"
1960  DATA     "33Ø HOME : HTAB  INT (17 - (
     LEN (T$) / 2)): PRINT [** [;: INVERSE :
     PRINT T$;: NORMAL : PRINT [ **[;"
1970  DATA      "34Ø NORMAL : VTAB 24: HTAB 7:
      INVERSE : PRINT[->[;: NORMAL : PRINT[MO
     VE[;: INVERSE : PRINT[<-[;: NORMAL : PRI
     NT[   [;: INVERSE : PRINT[<RETURN>[;: N
     ORMAL : PRINT[ SELECTS[;"
1980  DATA "35Ø HTAB TB(1): VTAB VE: FOR C =
     1 TO SN: HTAB TB(C): PRINT MENU$(C): NEX
     T : VTAB VE: HTAB TB(1): INVERSE : PRINT
      MENU$(1);: NORMAL "
1990  DATA  "36Ø REM  GET AND PROCESS KEY PRE
     SS"
2000  DATA  "37Ø GET A$: GOSUB 51Ø:X =  PEEK
     (37) + 1"
2010  DATA    "38Ø IF A$ =  CHR$ (8)OR A$=CHR
     $(11) THEN  GOSUB 43Ø: REM  BACKWARD OR
     UP ARROW REVERSES MENU SELECTION"
2020  DATA    "39Ø IF A$ =  CHR$ (21) OR A$ =
      CHR$ (32) OR A$=CHR$(1Ø)THEN  GOSUB 45Ø
      : REM  FORWARD OR DOWN ARROW OR SPACE BA
     R ADVANCES MENU SELECTION"
2030  DATA "4ØØ IF A$ =  CHR$ (13) THEN X =
     PEEK (37) - VE + 2: PRINT : RETURN : REM
        DETERMINE CHOICE BY CHECKING VERTICAL
      TAB POSITION"
2040  DATA    "41Ø GOTO 37Ø"
2050  DATA  "42Ø REM  TAB CONTROL ROUTINE"
2060  DATA "43Ø IF X = VE THEN  VTAB X: GOSUB
     48Ø:X = VE + SN - 1: GOSUB 49Ø: INVERSE
     : GOSUB 5ØØ: RETURN "
2070  DATA  "44Ø VTAB X: GOSUB 48Ø:X = X - 1:
      INVERSE : GOSUB 49Ø: GOSUB 5ØØ: RETURN
     "
2080  DATA "45Ø IF X = SN + VE - 1 THEN  VTAB
      SN + VE - 1: GOSUB 48Ø:X = VE: GOSUB 49
     Ø: INVERSE : GOSUB 5ØØ: RETURN "
2090  DATA "46Ø VTAB X: GOSUB 48Ø:X = X + 1:
     INVERSE : GOSUB 49Ø: GOSUB 5ØØ: RETURN "
2100  DATA   "47Ø REM   SUBROUTINE WHICH PRIN
     TS PROPER MENU STRING IN INVERSE DEPENDI
     NG ON KEY PRESS"
2110  DATA "48Ø HTAB TB(X - VE + 1): PRINT ME
     NU$(X - VE + 1);: RETURN "
2120  DATA "49Ø VTAB X: HTAB TB(X - VE + 1):
     RETURN "
2130  DATA "5ØØ PRINT MENU$(X - VE + 1);: NOR
     MAL : RETURN "
2140  DATA  "51Ø REM       ROUTINE TO PRODUCE
     SOUND"
2150  DATA  "52Ø IF A$ =  CHR$ (83) THEN SD =
     (SD = Ø)"

2160  DATA "53Ø IF SD THEN S =  - 16336:A =
     PEEK (S) -  PEEK (S) +  PEEK (S) -  PEEK
      (S) +  PEEK (S)"
2170  DATA "54Ø RETURN"
2180  DATA  "55Ø REM     MAIN AND SUBMENU CON
     TROL LINES"
2190  DATA      62ØØØ REM ERROR HANDLING ROUTI
     NE"
2200  DATA               "62Ø1Ø CALL -3288:PRI
     NTD$:PRINTD$[CLOSE[:PRINT D$[PR#Ø[:PRINT
      CHR$(7):X=PEEK(222):HTAB1:VTAB23:POKE35
     ,24:CALL-958:HTAB1:VTAB23
2210  DATA      "62Ø2Ø IF X=6 OR X=5 THEN PRINT
      [FILE NOT FOUND[;:X=257"
2220  DATA       "62Ø3Ø  IF X = 8 OR X = 9 OR
     X = 4 THEN  PRINT [I/O ERROR[;:X = 257"
2230  DATA       "62Ø4Ø IF X = 254 OR X = 255
     OR X = 53 OR X = 176 THEN  PRINT [BAD IN
     PUT ERROR[;:X = 257
2240  DATA        "62Ø5Ø IF X <  > 257 THEN
      VTAB 23: PRINT [ERROR NUMBER [PEEK (222
     )[ IN LINE [PEEK (218) +  PEEK (219) * 2
     56
2250  DATA         "62Ø6Ø PRINT [-PRESS <CR>
     : [;: POKE-16368,Ø:GET A$: PRINT : RETUR
     N
2260  REM    VIEW ASSEMBLED PROGRAMMED LINES
     ROUTINE
2270  GOSUB 361Ø: IF ER THEN ER = Ø: RETURN
2280  GOSUB 363Ø: IF ER THEN ER = Ø: RETURN
2290  HOME :MD$ = "VIEW/EDIT PROGRAM LINES": GOSUB
     318Ø
2300  GOSUB 239Ø
2310  POKE 34,2: POKE 35,23: HTAB 1: VTAB 12:
     PRINT "-->"PG$(1);: HTAB 39: VTAB 24:C2
     = 1
2320  POKE  - 16368,Ø: GET A$
2330  IF A$ =  CHR$ (32) OR A$ =  CHR$ (21) THEN
     C2 = C2 + 1: IF C2 > NL THEN C2 = 1
2340  IF A$ =  CHR$ (8) THEN C2 = C2 - 1: IF
     C2 = Ø THEN C2 = NL
```

LISTING 1: AUTO.MENU *(continued)*

```
2350  IF A$ = "Q" THEN  POKE 35,24: RETURN
2360  IF A$ =  CHR$ (13) THEN  GOSUB 2400: GOSUB
     2390
2370  HTAB 1: VTAB 12: PRINT "-->"PG$(C2);: CALL
     - 958: HTAB 39: VTAB 24
2380  GOTO 2320
2390 PR$ = "<-ARROWS SCROLL-> (Q)UITS <CR>TO
     EDIT:": POKE 35,23: GOSUB 3220: RETURN
2400 PR$ = "<RETURN> ACCEPTS,<CTRL>BDFINQZ TO
      EDIT:": GOSUB 3220: HTAB 1: VTAB 12:T =
     12
2410 M$ = " " + PG$(C2): PRINT M$"  ";: GOSUB
     2420:PG$(C2) = R$ + " ": RETURN
2420 L = 2
2430 E = 1
2440  VTAB T: HTAB L: POKE  - 16368,Ø: POKE  -
     16368,Ø: GET Z1$:X =  PEEK ( - 16384): VTAB
     T
2450  IF X < 32 THEN  ON X GOTO 2440,2420,244
     Ø,2550,2440,2610,2440,2560,2600,2440,244
     Ø,2440,2520,2640,2440,2440,2630,2440,244
     Ø,2440,2580,2440,2440,2440,2440,2620,244
     Ø,2440,2440,2440,2440: GOTO 2440
2460  IF E = 1 THEN M$ =  LEFT$ (M$,L - 1) +
     Z1$ +  MID$ (M$,L + 1): HTAB L: PRINT Z1
     $;:L = L + 1: GOTO 2440
2470  IF E = 2 THEN M$ =  LEFT$ (M$,L - 1) +
     Z1$ +  MID$ (M$,L): HTAB L: PRINT  MID$
     (M$,L);:L = L + 1: GOTO 2440
2480 F = W:W = X: IF F <  > X AND F THEN E =
     1: GOTO 2460
2490  FOR J3 = L + 1 TO  LEN (M$): IF Z1$ <  >
      MID$ (M$,J3,1) THEN  NEXT : GOTO 2440
2500  IF E THEN L = J3: GOTO 2440
2510 M$ =  LEFT$ (M$,L - 1) +  MID$ (M$,J3): HTAB
     L: CALL  - 958: PRINT  MID$ (M$,L): GOTO
     2440
2520 R$ =  MID$ (M$,2): IF  LEN (R$) = 1 THEN
     R$ = " " + R$
2530  RETURN
2540  REM    EDIT ROUTINE
2550 M$ =  LEFT$ (M$,L - 1) +  MID$ (M$,L + 1
     ): HTAB L: PRINT  MID$ (M$,L)" ";: GOTO
     2430
2560  IF L = 2 THEN 2430
2570 L = L - 1: GOTO 2430
2580  IF L < 1 +  LEN (M$) THEN L = L + 1
2590  GOTO 2430
2600 E = 2: GOTO 2440
2610 E = 3:W = Ø: GOTO 2440
2620 E = Ø:W = Ø: GOTO 2440
2630 M$ =  LEFT$ (M$,L - 1): HTAB L: CALL  -
     958: GOTO 2430
2640 L =  LEN (M$) + 1: GOTO 2430
2650  REM  CATALOG ROUTINE
2660  HOME :PR$ = "GETTING CATALOG...": GOSUB
     3220: PRINT : PRINT D$"CATALOG"
2670  PRINT :PR$ = "PRESS <RETURN>": GOSUB 32
     20: POKE  - 16368,Ø: GET A$: PRINT
2680  RETURN
2690  REM   SAVE MENU PROGRAM ROUTINE
2700  GOSUB 3610: IF ER THEN ER = Ø: RETURN
2710  GOSUB 3630: IF ER THEN ER = Ø: RETURN
2720  HOME : POKE 34,1:MD$ = "SAVE ASSEMBLED
     LINES TO DISK TEXT FILE": GOSUB 3180
2730  HOME : HTAB 1: VTAB 4: PRINT "ENTER FIL
     E NAME (Q)UITS": INPUT "-->";B$: IF B$ =
     "Q" THEN  RETURN
2740  GOSUB 3560: IF ER THEN ER = Ø: GOTO 273
     Ø
2750  IF  RIGHT$ (B$,5) <  > ".TEXT" THEN B$ =
     B$ + ".TEXT"
2760  PRINT D$"OPEN"B$: PRINT D$"CLOSE"B$: PRINT
     D$"DELETE"B$
2770  PRINT : PRINT "SAVING " LEFT$ (B$, LEN
     (B$) - 5)" TO DISK: "
2780  PRINT D$"OPEN"B$: PRINT D$"WRITE"B$
2790  PRINT NL: FOR C = 1 TO NL: IF PG$(C) <
     > "" THEN  PRINT PG$(C)
2800  NEXT
2810  PRINT "POKE34,Ø": PRINT "HOME": PRINT "
     SAVE " LEFT$ (B$, LEN (B$) - 5): PRINT "
     DELETE IF"
2820  PRINT "RUN"
2830  PRINT D$"CLOSE"B$
```

```
2840 PR$ = "FILE SAVED TO DISK-PRESS <CR>: ":
      GOSUB 3220: POKE  - 16368,Ø: GET A$:SV =
     1: RETURN
2850  REM     LOAD CREATED MENU PROGRAM ROUTI
     NE
2860  IF N > Ø OR N1 > Ø THEN PR$ = "MUST RES
     TART FIRST: ": GOSUB 3220: POKE  - 16368
     ,Ø: GET A$: POP : RETURN
2870  HOME :MD$ = "LOAD CREATED MENU TEXT FIL
     E": GOSUB 3180
2880  HTAB 1: VTAB 4: PRINT "ENTER FILE NAME
     (Q)UITS";: INPUT ": ";B$: IF B$ = "Q" THEN
      RETURN
2890  GOSUB 3560: IF ER THEN ER = Ø: GOTO 288
     Ø
2900  IF  RIGHT$ (B$,5) <  > ".TEXT" THEN B$ =
     B$ + ".TEXT"
2910  PRINT D$"VERIFY"B$
2920  PRINT D$"OPEN FN": PRINT D$"WRITE FN": PRINT
     B$: PRINT D$"CLOSE FN": REM  SAVE FILE N
     AME TO DISK SO CLEAR IS POSSIBLE
2930  POKE 34,12
2940  CLEAR :D$ =  CHR$ (4)
2950  HTAB 1: VTAB 7: PRINT "WAIT WHILE PREPA
     RING TO READ FILE"
2960  IF  PEEK (48896) <  > 76 THEN  PRINT D$
     "BLOAD CHAIN,A520": CALL 520"LOADER": REM
       RUN LOADER PROGRAM TO READ THE FILE WI
     TH DOS 3.3 CHAIN PROGRAM FROM SYSTEM MAS
     TER DISK
2965  PRINT D$"CHAIN LOADER": REM  FOR PRODOS
2970  REM  NO RETURN NEEDED HERE BECAUSE LOAD
     ER RETURNS CONTROL TO FIRST LINES
2980  REM    TEST RUN CREATED MENU ROUTINE
2990  HOME : POKE 34,1:MD$ = "TEST RUN CREATE
     D MENU ": GOSUB 3180
3000  HOME : HTAB 1: VTAB 4: PRINT "ENTER FIL
     E NAME TO TEST (Q)UITS": INPUT "-->";B$:
      IF B$ = "Q" THEN  RETURN
3010  GOSUB 3560: IF ER THEN ER = Ø: GOTO 299
     Ø
3020  IF  RIGHT$ (B$,5) <  > ".TEXT" THEN B$ =
     B$ + ".TEXT"
3030  PRINT D$"VERIFY"B$
3040  HOME : POKE 34,16: HTAB 1: VTAB 3: PRINT
     "PREPARING TO READ FILE":
3050  PRINT D$: PRINT D$"OPEN IF": PRINT D$"W
     RITE IF"
3060  PRINT "NEW": PRINT "EXEC "B$
3070  PRINT D$"CLOSE IF"
3080  HTAB 1: VTAB 3: PRINT "READING IN "B$"
     FILE.."
3090  PRINT : PRINT "REMEMBER...": PRINT : PRINT
     "1. "B$" IS A TEXT FILE": PRINT : PRINT
     "2. " LEFT$ (B$, LEN (B$) - 5)" IS AN AP
     PLESOFT FILE"
3100  VTAB 17: PRINT
3110  PRINT D$"EXEC IF"
3120  END
3130  REM   QUIT ROUTINE
3140  IF  NOT SV THEN PR$ = "PROGRAM NOT SAVE
     D! QUIT? (Y/N): ": FLASH : GOSUB 3220: NORMAL
     : POKE  - 16368,Ø: GET A$: IF A$ <  > "Y
     " AND A$ <  > "N" THEN 3140
3150  IF A$ = "Y" THEN  TEXT : HOME : END
3160  RETURN
3170  REM   UTILITY ROUTINES
3180  HTAB 1: VTAB 1: INVERSE : PRINT MD$: NORMAL
     : RETURN
3190  VTAB (N + 2): HTAB (20 -  INT ( LEN (B$
     ) / 2)): PRINT B$:B$ = "": RETURN
3200  HTAB 1: VTAB 24: FOR C = 1 TO 39: PRINT
     " ";: NEXT : RETURN
3210 PR$ = "CORRECT? (Y/N): ": GOSUB 3220: POKE
      - 16368,Ø: GET A$: GOSUB 3200: RETURN
3220  GOSUB 3200: VTAB 24: HTAB 20 -  INT ( LEN
     (PR$) / 2): PRINT PR$;: RETURN
3230  HTAB (20 -  INT ( LEN (B$) / 2)): PRINT
     B$:B$ = "": RETURN
3240  VTAB (N1 + 7): HTAB (20 -  INT ( LEN (B
     $) / 2)): PRINT B$:B$ = "": RETURN
3250  REM  LINE# AND OTHER CHECKS FOR MAIN ME
     NU
3260  IF LL > 60000 OR UL > 60000 THEN ER$ =
     "CHOOSE LINE# < 60000": GOSUB 3410
```

LISTING 1: AUTO.MENU *(continued)*

```
3270  IF UL > Ø AND UL - LL > 10000 THEN ER$ =
     "SUBROUTINE TOO LARGE": GOSUB 341Ø
3280  IF LL / 1ØØ <  >  INT (LL / 1ØØ) THEN E
     R$ = "CHOOSE MULTIPLES OF 1ØØ": GOSUB 34
     1Ø
329Ø A = (LL < 7ØØ) * 699 + (UL > Ø AND UL <
     LL) * (LL + 99) + (UL =  < Ø) * (LL < 7Ø
     Ø) * (LL > 7ØØ) * (LL + 99): IF LL < 7ØØ
      OR (UL > Ø AND UL < LL + 99) OR (UL < Ø
     ) THEN ER$ = "CHOOSE LINE# >" +  STR$ (A
     ):A = Ø: GOSUB 341Ø
33ØØ A = (LL < 7ØØ) * 699 + (UL > Ø AND UL <
     LL) * (LL + 99) + (UL =  < Ø) * (LL < 7Ø
     Ø) * (LL > 7ØØ) * (LL + 99): IF LL < 7ØØ
      OR (UL > Ø AND UL < LL + 99) OR (UL < Ø
     ) THEN ER$ = "CHOOSE LINE# >" +  STR$ (A
     ):A = Ø: GOSUB 341Ø
331Ø  FOR C = LL TO UL STEP 1ØØ: IF LN$(C / 1
     ØØ) = "" THEN ER$ = "LINE NUMBERS ASSIGN
     ED": GOSUB 341Ø
332Ø  FOR C = LL TO UL STEP 1ØØ: IF LN$(C / 1
     ØØ) = "" THEN ER$ = "LINE NUMBERS ASSIGN
     ED": GOSUB 341Ø
333Ø  NEXT
334Ø C = LL + 1ØØ: IF LN$(C / 1ØØ) = "" THEN
     ER$ = "NO ROOM FOR SUBROUTINE": GOSUB 34
     1Ø
335Ø  GOSUB 32ØØ: RETURN
336Ø  IF LL > Ø AND UL > Ø AND UL - LL < 299 OR
     NS > 5 THEN ER = 1
337Ø  RETURN
338Ø PR$ = "ABORT DATA ENTRY AND RESTART? (Y/
     N): ": GOSUB 322Ø: POKE  - 16368,Ø: GET
     A$: IF A$ = "Y" THEN  GOTO 8Ø
339Ø  RETURN
34ØØ  REM  ERROR DISPLAY ROUTINES
341Ø ER$ =  CHR$ (7) + "ERROR! " + ER$ + " <C
     R>: "
342Ø HT = 2Ø -  INT ( LEN (ER$) / 2):ER = 1
343Ø  GOSUB 32ØØ: HTAB HT: PRINT ER$;: POKE  -
     16368,Ø: GET A$: GOSUB 32ØØ: POP : RETURN

344Ø  REM    LINE # AND OTHER CHECKS FOR SUBR
     OUTINE MENUS
345Ø A = (LL < L1) * (L1 - 1) + (UL > Ø AND U
     L < LL + 99) * (LL + 99) + (UL =  < Ø) *
     (L1 < LL) * (LL > Ø) * (LL): IF LL < L1 OR
     (UL > Ø AND UL < LL + 99) OR (UL < Ø) THEN
     ER$ = "CHOOSE LINE# >" +  STR$ (A):A = Ø
     : GOSUB 341Ø
346Ø A = (UL > L2) * L2 + ((LL > L2 - 99) * L
     2 - 98): IF UL > L2 OR LL > L2 - 99 THEN
     ER$ = "CHOOSE LINE# <" +  STR$ (A):A = Ø
     : GOSUB 341Ø
347Ø  FOR C = LL TO UL STEP 1ØØ: IF LN$(C / 1
     ØØ) = "" THEN ER$ = "LINE NUMBERS ASSIGN
     ED": GOSUB 341Ø
348Ø  NEXT
349Ø  IF LL / 1ØØ <  >  INT (LL / 1ØØ) THEN E
     R$ = "NOT MULTIPLE OF 1ØØ": GOSUB 341Ø
35ØØ  IF N1 = 1 AND UL > Ø AND LL > Ø AND UL -
     LL > L2 - L1 - 99 THEN ER$ = "NO ROOM FO
     R 2 ROUTINES": GOSUB 341Ø
351Ø  GOSUB 32ØØ: RETURN
352Ø  FOR C = 1 TO N: IF B$ = MM$(C,Ø) THEN E
     R$ = "OPTION NOT DISTINCT": GOSUB 341Ø
353Ø  NEXT
354Ø  IF  LEN (B$) > 33 OR  LEN (B$) < 2 THEN
     ER$ = "IMPROPER ENTRY LENGTH: ": GOSUB 3
     41Ø
355Ø  RETURN
356Ø A = 15: IF  RIGHT$ (B$,5) = ".TEXT" THEN
     A = 1Ø
357Ø  IF  LEN (B$) > A OR  LEN (B$) < 1 THEN
     ER$ = "IMPROPER ENTRY LENGTH": GOSUB 342
     Ø
358Ø  RETURN
359Ø  FOR C = 1 TO N1: IF B$ = SB$(N,Ø,C) THEN
     ER$ = "OPTION NOT DISTINCT": GOSUB 341Ø
36ØØ  NEXT : RETURN
361Ø  IF DA = Ø AND LD = Ø THEN ER$ = "NO DAT
     A AVAILABLE": GOSUB 341Ø
362Ø  RETURN
363Ø  IF AS = Ø THEN ER$ = "ASSEMBLE LINES FI
     RST": GOSUB 341Ø
364Ø  RETURN
365Ø  FOR C = 1 TO  LEN (B$):A =  ASC ( MID$
     (B$,C,1)): IF A < 32 OR A > 96 THEN 367Ø
366Ø  NEXT : GOTO 368Ø
367Ø ER$ = "UPPER CASE ONLY": GOSUB 341Ø
368Ø  RETURN
369Ø  REM  INITIALIZATION ROUTINE
37ØØ  CLEAR : POKE 34,Ø: HOME
371Ø PR$ = "INITIALIZING": INVERSE : GOSUB 32
     2Ø: NORMAL
372Ø  DIM MM$(13,3),SB$(12,2,6),LN$(6Ø1),PG$(
     225),AP(4),TB(12),ME$(12)
373Ø  FOR C = 7ØØ TO 6ØØØØ STEP 1ØØ:LN$(C / 1
     ØØ) =  STR$ (C): NEXT
374Ø D$ =  CHR$ (4): ONERR  GOTO 377Ø
375Ø  GOTO 17Ø
376Ø  REM  ERROR HANDLER ROUTINE
377Ø  CALL  - 3288: PRINT D$: PRINT D$"CLOSE"
     : PRINT D$"PR#Ø": PRINT  CHR$ (7):X =  PEEK
     (222): HTAB 1: VTAB 23: POKE 35,24: CALL
      - 958: HTAB 1: VTAB 23
378Ø  IF X = 6 OR X = 5 THEN  PRINT "FILE NOT
      FOUND";:X = 257
379Ø  IF X = 8 OR X = 9 OR X = 4 THEN  PRINT
     "I/O ERROR";:X = 257
38ØØ  IF X = 254 OR X = 255 OR X = 53 OR X =
     176 THEN  PRINT "BAD INPUT ERROR";:X = 2
     57
381Ø  IF X <  > 257 THEN  PRINT "ERROR NUMBER
     " PEEK (222)" IN LINE " PEEK (218) +  PEEK
     (219) * 256
382Ø  PRINT "-PRESS <CR>: ";: GET A$: PRINT :
      RETURN
383Ø A$ = " ":L = Ø: FOR C = 2 TO  VAL (MM$(Ø
     ,Ø)): REM  DETERMINE LONGEST STRING
384Ø  IF  LEN (MM$(C,Ø)) > L THEN L =  LEN (M
     M$(C,Ø))
385Ø  NEXT : IF L / 2 =  INT (L / 2) THEN L =
     L + 1: REM  MAKE SURE L IS ODD FOR CENTE
     RING PURPOSES
386Ø  FOR C = 1 TO  VAL (MM$(Ø,Ø)): REM  PAD
     WITH SPACES
387Ø  IF  LEN (MM$(C,Ø)) < L THEN MM$(C,Ø) =
     MM$(C,Ø) + A$:MM$(C,Ø) =  MID$ (A$,1,( LEN
     (MM$(C,Ø)) < L)) + MM$(C,Ø): GOTO 387Ø
388Ø  NEXT : IF  VAL (SB$(Ø,Ø,Ø)) = Ø THEN 39
     7Ø
389Ø  FOR C = 1 TO  VAL (SB$(Ø,Ø,Ø)):L = 19: REM
      LENGTH OF RETURN TO MAIN MENU
39ØØ  FOR C1 = 1 TO  VAL (SB$(C,Ø,Ø)): REM  F
     IND LONGEST SUBMENU STRING
391Ø  IF  LEN (SB$(C,Ø,C1)) > L THEN L =  LEN
     (SB$(C,Ø,C1))
392Ø  NEXT
393Ø  FOR C1 = 1 TO  VAL (SB$(C,Ø,Ø))
394Ø  IF  LEN (SB$(C,Ø,C1)) < L THEN SB$(C,Ø,
     C1) = SB$(C,Ø,C1) + A$:SB$(C,Ø,C1) =  MID$
     (A$,1, LEN (SB$(C,Ø,C1)) < L) + SB$(C,Ø,
     C1): GOTO 394Ø
395Ø  NEXT
396Ø  NEXT
397Ø  RETURN
398Ø A = X - 1:AP(A) = (AP(A) = Ø)
399Ø A$ = "YN": FOR C = 1 TO 3: HTAB 35: VTAB
     16 + C: PRINT  MID$ (A$,(AP(C) = Ø) + 1,
     1);: NEXT : VTAB (X) + VE - 1: RETURN
4ØØØ  FOR C2 = 1 TO  LEN (A$): IF  MID$ (A$,C
     2,1) = "[" THEN A$ =  LEFT$ (A$,C2 - 1) +
     CHR$ (34) +  MID$ (A$,C2 + 1)
4Ø1Ø  NEXT : RETURN
4Ø2Ø  HOME : VTAB 12: PRINT "RESTART AND CLEA
     R MEMORY (Y/N): ";: GET A$: IF A$ = "Y" THEN
      RUN 5Ø
4Ø3Ø  RETURN
4Ø4Ø F$ = "RETURN TO MAIN MENU, ":L = AP(3) *
     ( LEN (SB$(C,Ø,1)) - 19): IF L > 1 THEN
     F$ =  LEFT$ (F$,19): FOR C1 = 1 TO L / 2
     :F$ = " " + F$ + " ": NEXT :F$ =  CHR$ (
     34) + F$ +  CHR$ (34) + ", "
4Ø5Ø  RETURN
```

END OF LISTING 1

```
             KEY PERFECT 5.Ø
                 RUN ON
               AUTO.MENU
=====================================
 CODE-5.Ø    LINE# - LINE#    CODE-4.Ø
 --------    -------------    --------
 BEA2Ø9BD       1 -    7Ø        A7ØØ
 B4ØC4D92      8Ø -   17Ø      Ø19874
 16FC66A3     18Ø -   27Ø      Ø148DØ
 87AD2628     28Ø -   37Ø        CF9D
 2FEDA8FE     38Ø -   47Ø        871Ø
 794FØØ61     48Ø -   57Ø        E77D
 Ø7FE525Ø     58Ø -   67Ø        C4D9
 D6C4Ø399     68Ø -   77Ø        C871
 79F262B1     78Ø -   87Ø        8EA8
 D117A9BB     88Ø -   97Ø      Ø146F4
 82874C99     98Ø -  1Ø7Ø        C1D9
 91A99F2Ø    1Ø8Ø -  117Ø        6B14
 55629DF4    118Ø -  127Ø        DDC4
 69654461    128Ø -  137Ø        95C7
 88B274E7    138Ø -  147Ø        BØF5
 EDA917F2    148Ø -  157Ø        B66E
 38D37ØE5    158Ø -  167Ø        FFFC
 3DD34179    168Ø -  177Ø      Ø11262
 63E12B57    178Ø -  187Ø      Ø1171D
 1F3C5ØCD    188Ø -  197Ø      Ø1B8A9
 BC6DØAE3    198Ø -  2Ø7Ø      Ø148Ø7
 96718100    2Ø8Ø -  217Ø        F928
 5CCFEF8C    218Ø -  227Ø      Ø11DCB
 5FF59595    228Ø -  237Ø        8BA4
 A2F676Ø5    238Ø -  247Ø      Ø1Ø1CE
 9126B13E    248Ø -  257Ø        79F6
 6637DBC1    258Ø -  267Ø        6DA7
 98F74163    268Ø -  277Ø        9495
 DF4EØ273    278Ø -  287Ø        AC59
 83Ø1CD1D    288Ø -  2965        D166
 74829904    297Ø -  3Ø6Ø        B4C3
 9DD57Ø5B    3Ø7Ø -  316Ø        8469
 43Ø93B27    317Ø -  326Ø        B262
 FB9BCC63    327Ø -  336Ø      Ø12F7F
 ØE649Ø3Ø    337Ø -  346Ø        E5E8
 FB876AF8    347Ø -  356Ø        AØ86
 16EBD37F    357Ø -  366Ø        76ØE
 2CØ2F5CC    367Ø -  376Ø        757F
 BØE87B31    377Ø -  386Ø        D75A
 AF36E4F1    387Ø -  396Ø        B559
 4C35B379    397Ø -  4Ø5Ø        9768
 ØF46FC52 = PROGRAM TOTAL =      46AD
```

LISTING 2: LOADER

```
10  REM ************************
20  REM *        LOADER        *
30  REM * BY KENNETH PENNER    *
40  REM * COPYRIGHT (C) 1986   *
50  REM * BY MICROSPARC, INC   *
60  REM * CONCORD, MA  Ø1742   *
70  REM ************************
80  DIM ME$(12),TB$(12),PG$(225):EF = Ø
90 D$ =  CHR$ (4):SD = 1:DA = 1:LD = 1:N = 1:
     AS = 1: REM   SOUND,DATA,LOAD,ASSEMBLED
     FLAGS SET
100  ONERR  GOTO 19Ø
110  IF EF = Ø THEN  TEXT : HOME : PRINT : PRINT
     D$"OPENFN": PRINT D$"READFN": INPUT B$: PRINT
     D$"CLOSEFN": PRINT D$"DELETEFN": REM   R
     EAD FILE NAME
120  HTAB 1: VTAB 7: PRINT "NOW READING FILE
     INTO MEMORY...";: CALL  - 958: REM  CLEA
     R TO END OF LINE
130  PRINT : HTAB 1: VTAB 13: PRINT D$"OPEN"B
     $: PRINT D$"READ"B$
140 C1 = 1:PG$(1) = ""
150  INPUT NL
160  GET A$: PRINT A$;: IF A$ <  >  CHR$ (13)
      THEN PG$(C1) = PG$(C1) + A$: GOTO 16Ø: REM
      PLACE A BASIC LINE IN PG$(C1) BY GETTIN
     G ONE CHARACTER AT A TIME UNTIL <RETURN>
      ENCOUNTERED
170  IF PG$(C1) <  > "" THEN C1 = C1 + 1:PG$(
     C1) = "": REM   CHECK FOR TWO (OR MORE)
     RETURNS IN A ROW
180  GOTO 16Ø
190  PRINT : PRINT D$"CLOSE"B$
200  IF  PEEK (222) <  > 5 OR NL = Ø THEN  HOME
     : VTAB 12: POKE 216,Ø:EF = 1: PRINT "UNA
     BLE TO READ FILE": PRINT : INPUT "DO YOU
      WANT TO TRY AGAIN? ";YN$: ON YN$ = "Y" GOTO
     9Ø: END
210  PRINT : PRINT
220  POKE 34,Ø: VTAB 24: HTAB 1: PRINT B$" NO
     W IN MEMORY-<CR>: ";: GET A$: PRINT
230  HOME : VTAB 12: HTAB 1: INVERSE : PRINT
     "WAIT WHILE RELOADING AUTO.MENU": NORMAL
     : PRINT
240  IF  PEEK (48896) <  > 76 THEN  PRINT D$"
     BLOAD CHAIN,A52Ø": CALL 52Ø"AUTO.MENU"
250  PRINT D$"CHAIN AUTO.MENU"
```

END OF LISTING 2