SUBROUTINE STORE

DOS 3.3

Keep all your frequently used subroutines in one place, using this DOS 3.3 program to add them to other programs. You can change the starting numbers to conform to the new program.

by Ron Sjolander, 1249 West Ave., Hilton, NY 14468

For the subroutine collector, programming magazines are the mother lode. They are full of clever subroutines for you to input, test and save for future use. But once you type them in, you are left with your favorite subroutines scattered on several disks under short file names. Which routine does what? It's easier to retype a subroutine into a new program than to find the one you want, renumber it, create a text file and merge it.

Subroutine Store changes all that. This Applesoft program can store up to 50 subroutines. The first line of each subroutine is a REMark that describes the routine. These descriptive REM statements can be displayed three at a time, so you can choose the subroutine you want.

As you make your choices, you can LIST a subroutine to see just what it does. You can number all the subroutines consecutively starting with the number you supply, or you can select a different number for each. After you select a subroutine, you are prompted for the name of the program to which it will be added. You can SAVE the resulting program under the same name or a new one. Then Subroutine Store combines the selected subroutines with the program you named and saves the result.

USING THE PROGRAM: A SAMPLE SESSION

Subroutine Store has two primary functions:

1. It allows you to select from subroutines you have already stored in the system, and to add them to your own program.
2. It allows you to add new routines to those already stored.

These two options are presented when you first run SUBROUTINE .STORE. To demonstrate the first option, we will use a subroutine that is included in the printed listing of SUBROUTINE.STORE (**Listing 1**), the "PRESS ANY KEY TO CONTINUE" routine in **lines 5000-5020**. First, we must have a program to which the routine can be added. Type in the short program shown in **Example 1** and save it on the same disk as SUBROUTINE.STORE under the name TEST.PROGRAM. Then run SUBROUTINE.STORE and select option 1. After a brief delay, it will display:

```
5000 REM ANY KEY TO CONTINUE ROUTINE
```

If more subroutines were stored in SUBROUTINE.STORE, two more REMs would be displayed at this time, and you could display others on subsequent screens in groups of three. You may choose one of three options:

1. Press < RETURN > to display the next group of three subroutines. In this case, only the one routine (starting at **line 5000**) is stored in the program, so pressing < RETURN > will warn you of this fact and allow you to either view the previous screen or return to the menu.
2. Press S to stop the display of subroutine titles and prepare to add the routines you have selected to your program. If you have not yet selected any subroutines, you will be returned to the main menu.
3. Type a line number to select a subroutine from the store. You may then either list the routine to the screen or merge it into your program.

For our example, type **5000** to select the PRESS ANY KEY TO CONTINUE subroutine. Then type **M** to add the subroutine to your Applesoft program. You are then asked if you want to select a different starting line number for each subroutine you have selected. (It doesn't matter what you answer in this case, since only one routine is available.) Next, you are asked for the starting line number. Enter 1000. The program will take a while to renumber and relocate the subroutine, and then returns you to the display of subroutine files.

EXAMPLE 1: Test Program

```
10  REM  TEST PROGRAM
20  HOME
30  FOR X = 10 TO 0 STEP  - 1
40  PRINT X
50  FOR I = 1 TO 1000: NEXT I
60  NEXT X
70  END
```

This time press S <**RETURN**> to indicate you have finished selecting subroutines. Next you are asked the name of your Applesoft program. Enter **TEST.PROGRAM**. Subroutine Store will verify that the program you named exists. Finally, you are asked if you want the modified program to be saved under the same name. If you press Y, you will replace the original program, so instead press N and enter **TEST.PROGRAM.B** for the new name. The rest of the process is automatic. If all goes well, you will end up with your new program in memory, and with a copy saved on disk. All that remains is to link the subroutine into the body of the program. This can be done by replacing **line 50** of TEST.PROGRAM.B with the following:

```
50 GOSUB 1000: REM ANY KEY TO CONTINUE
```

Save this new program under the name TEST.PROGRAM.B. If any of the subroutines you added had included internal line number references, you would have an additional step — reconciling these references with their new locations.

To add new subroutines to SUBROUTINE.STORE, just select option 2 from the first menu. The program will find the next available starting line number for your new subroutine and then stop. To add a text screen dump subroutine to your store, type the lines shown in **Example 2**. Just type in the subroutine as you would any Applesoft program, starting at the line number given by SUBROUTINE.STORE (in this case, **line 5100**, as shown in the listing), and save the result with:

SAVE SUBROUTINE.STORE

SUBROUTINE.STORE now has two subroutines available to add to your Applesoft programs. Note that the length of this subroutine causes it to use **line 5200**, which would normally be used for the next subroutine. As long as you don't begin the statement in **line 5200** with a REM, SUBROUTINE.STORE will skip this line and use **line 5300** for the next subroutine.

EXAMPLE 2: Screen Dump Subroutine

```
5100  REM  TEXT SCREEN DUMP
5110  GOTO 5200
5120  FOR J = I TO I + 39
5130 A =  PEEK (J):A = A + (A < 32) * 192
5140 A = A + (A < 64) * 128:A = A + (A < 96) *
     64
5150 A = A + (A < 128) * 64:A = A + (A < 160)
      * 64
5160  PRINT  CHR$ (A);
5170  NEXT J
5180  PRINT  CHR$ (13);:
5190  RETURN
5200  PRINT  CHR$ (4)"PR#1": PRINT  CHR$ (9)"
     80N"
5210  FOR I = 1024 TO 1920 STEP 128: GOSUB 51
     20: NEXT I
5220  FOR I = 1064 TO 1960 STEP 128: GOSUB 51
     20: NEXT I
5230  FOR I = 1104 TO 2000 STEP 128: GOSUB 51
     20: NEXT I
5240  PRINT  CHR$ (4)"PR#0"
5250  RETURN
```

ENTERING THE PROGRAM

To key in SUBROUTINE.STORE, enter the Applesoft program shown in **Listing 1**. It is important to preserve the line numbers, since the program is self-modifying. Note that there are breaks in the numbering sequence between **lines 1520 and 5000**, and between **lines 5020 and 15000**. Save the program with the command:

SAVE SUBROUTINE.STORE

For help in entering *Nibble* listings, see "A Welcome to New *Nibble* Readers" at the beginning of this issue.

KEY ROUTINES

SUBROUTINE.STORE is hardly more than a collection of subroutines itself. Some of the routines are used more than once with minor modifications to fit a particular purpose. The key routines do the following:

1. Locate the starting address in memory for a selected line number or check to see if a line number exists (subroutine at **line 140** plus other versions starting at **lines 1120, 1270, 15880 and 16140**).
2. Use the starting address of a line to change its contents while the program is running (subroutines at **lines 350 and 440** plus a variation at **line 1370**).
3. Move a block of memory (subroutine at **line 410**).
4. Save a portion of a program to be loaded into memory along with another program and then run it by executing a text file (the "temporary" program referred to consists of **lines 16110-16280** of SUBROUTINE.STORE, located and saved by **lines 15720-15780**, and run (and deleted) by text files created in **lines 15430-15620**).

Since these subroutines are central to program functioning, a brief explanation of each follows.

LOCATING A LINE NUMBER IN MEMORY

The key is knowing how Applesoft stores program lines. For example, let's look at a two-line program starting at the normal starting address of $800 (2048 decimal):

```
100 HOME
110 PRINT X
```

Table 1 shows how the program is actually stored in memory. The address of the next line and the current line number are stored in typical Applesoft, two-byte format. The first byte is the low-order byte and the second byte is the high-order byte. Multiplying the high-order byte by 256 and adding the low-order byte to the sum converts it to a decimal number (the subroutine at **line 250** does the reverse).

The subroutine at **line 140** finds a particular line number by starting at the beginning of the program and PEEKing ahead from line to line, using the address of the next line, until it matches the line

TABLE 1: Applesoft Program Storage

Address			
Hex	**Decimal**	**Contents**	**Function**
0800	2048	00	Start of program
0801	2049	07	Address of next line in memory
0802	2050	08	(2055 decimal)
0803	2051	64	Line number (100 decimal)
0804	2052	00	
0805	2053	97	Applesoft token for HOME
0806	2054	00	End-of-line marker
0807	2055	0E	Address of next line in memory
0808	2056	08	(2062 decimal)
0809	2057	6E	Line number (110 decimal)
080A	2058	00	
080B	2059	BA	Applesoft token for PRINT
080C	2060	58	ASCII code for "X"
080D	2061	00	End-of-line marker
080E	2062	00	End-of-program marker 1
080F	2063	00	End-of-program marker 2

number or until the address of the next line equals zero, indicating the end of the program. **Line 150** finds the address of the first line (ADR) by PEEKing the start-of-program pointer at decimal 103 and 104, and converts the contents to a decimal number. **Line 170** is the start of a loop that gets the address of the next line (NXADR), gets the line number (LN), and compares the line number to the one being sought (CN). If the line number is less than the one being sought, it moves on to the next line. If the line number is greater, or the end of the program has been reached (NXADR = 0), the search ends with the "found" indicator (FD) set to zero. If the line number is found, FD is set to one (1), ADR is set equal to the starting address of the line, and the subroutine ends.

CHANGING THE CONTENTS OF A LINE

The subroutines at **lines 350 and 440** contain LIST commands. The first lists the beginning line of a stored subroutine (REM statement); the second lists the entire routine.

The line numbers within the LIST statement are changed by finding the starting address of the line (using the subroutine at **line 140**) and POKEing the desired line number into the LIST statement before it is executed. The desired line number is converted to a string, which is read a character at a time with the MID$ function, converted to ASCII code, and POKEd directly into memory. Note the comma instead of a dash at **line 500**. Applesoft would misinterpret a dash in this case.

MOVING THE CONTENTS OF MEMORY

The Monitor contains a MOVE routine for moving a specified segment of memory to a new location. It would perhaps be more accurate to say it copies a segment of memory to a new location, since the original segment stays where it is. Semantics aside, the Monitor MOVE routine is readily accessible from an Applesoft program. The MOVE routine gets the appropriate address from three specific pairs of memory locations:

Location		
Hex	Decimal	Description
3C,3D	60,61	Low/high byte of start-of-memory to move
3E,3F	62,63	Low/high byte of end-of-memory to move
42,43	66,67	Low/high byte of destination of memory being moved

The appropriate addresses are POKEd into these locations before the MOVE routine is executed. Since the Monitor was not designed to be CALLed directly from an Applesoft program, an additional user action to provide a RETURN to the Applesoft program is required. The subroutine at **line 410** accomplishes this indirectly by CALLing a routine (CALL 65209) that goes to the address stored in decimal 58 and 59. The address of the MOVE routine (65068), is POKEd into these locations and program flow RETURNs to the Applesoft program after the MOVE routine is complete. This is the approach used by Eric E. Goez in his program Real Variable Study (published in *Call-A.P.P.L.E. In Depth*, 1981).

SUBROUTINE.STORE "moves" subroutines, as you select them, to a location just above HIMEM (set initially in **line 100**). The appropriate addresses are found and calculated in **lines 1200-1250** before branching to the subroutine at **line 410**. The line numbers for the subroutine are changed after being moved using the technique described in the section CHANGING THE CONTENTS OF A LINE (above).

The selected subroutines are LISTed to a temporary text file by changing the start-of-program pointer to its new starting address before the LIST command (**line 15440**). To ensure that only the subroutines are LISTed, the subroutine at **line 680** first puts zeros in high memory where the subroutines will go. This ensures that the LIST routine will find a next line number address of zero and stop at the last line.

TEXT FILE ROUTINES

When a text file is EXECed, it is as if a series of direct commands were entered from the keyboard. Checking for existing line numbers, as in **lines 15830-15960**, would be difficult at best via a text file. Instead, **lines 15720-15780** save a subset of SUBROUTINE.STORE under a separate name, SUB.TEMP. After loading the Applesoft program with which to merge the subroutines, this program is loaded beyond the address indicated by the end-of-program pointer (decimal 175 and 176). Before it is run (by changing the start-of-program pointer), a zero is POKEd into memory just ahead of it (**line 15600**). If you look again at the two-line program described in **Table 1**, you'll see that the first byte is a zero used as a start-of-program indicator.

In this program, two text files are created. **Lines 15430-15530** create a text file (TEMP.TEMP). It contains the subroutines that:

1. Add a subroutine to your program.
2. Save the result.
3. Restore things to normal before deleting itself. If all goes well, the program ends here.

The second text file (EXEC.TEMP) is created in **lines 15560-15620** for the times when things don't go well. This text file loads your program and loads and runs the temporary program, SUB .TEMP. Besides checking for line number conflicts, SUB.TEMP checks that a program to which the subroutines will be added has, in fact, been loaded. If you somehow managed to make a typo entering the program name, you'd probably get an error message, but EXEC.TEMP wouldn't stop. It would continue as if you had entered another direct command. If you missed the error message, things would continue and you could end up adding subroutines to SUBROUTINE.STORE and saving the result in place of your program. To avoid that, SUB.TEMP adds four of the values of the ASCII codes in the last line of the program in memory. The funny looking REM statement at the end of SUBROUTINE.STORE adds up to 337. If the sum of the ASCII codes for the last line of the program in memory isn't 337, then a program was loaded.

MODIFICATIONS

Since SUBROUTINE.STORE is used to add routines during program development, it is assumed that a line editor will occupy HIMEM when the program is run. **Line 80** checks for the current value of HIMEM. **Line 100** resets HIMEM and **line 110** sets a memory limit (HM) to prevent overwriting the line editor. If you're not using a line editor, HIMEM will simply be higher in memory. The program should work without modification with any line editor that resides above HIMEM on a 48K or larger machine.

To allow room for customizing, line numbers for the subroutines added to a program are incremented in tens. To change this value, adjust the variable LI in **line 120**, which sets the increment. The variable LS in **line 120** holds the increment for continuously numbered subroutines. You might want to add an input routine to set these values each time the program is run or even as each subroutine is selected.

A few of the values in this program were selected rather arbitrarily. The maximum of 50 subroutines could be changed. If you want a different value, change the DIMension of SB in **line 120** to your new value plus two, change the FOR statements containing "1 to 50," and change "(50-I)" in **line 15240**.

Line 1250 assumes that the last line number in the last subroutine stored is no more than 200 greater than the first line number (20 lines numbered by 10, for example). If you regularly use long subroutines, you may want a larger value (500 or 1000). Since this only applies when you pick the last subroutine you have stored, be conservative and pick a number large enough to fit the largest subroutine you're likely to store.

Caution: Since this program is used to save other programs, test your typing adequately before you use it to save a routine using its original name. It is best to use different names at first. A bit of caution may save you a few hours of work.

LISTING 1: SUBROUTINE.STORE

```
10  REM **********************
20  REM * SUBROUTINE.STORE    *
30  REM * BY RON SJOLANDER    *
40  REM * COPYRIGHT (C) 1985 *
50  REM * BY MICROSPARC, INC *
60  REM * CONCORD, MA  01742 *
70  REM **********************
80 HM =  PEEK (115) +  PEEK (116) * 256:MEM =
    HM: IF  PEEK (782) <  > 255 - ( PEEK (78
    1) +  PEEK (780) - 256 * ( PEEK (780) +
     PEEK (781) > 255)) THEN MEM = HM - 25 *
    256
90  POKE 780, PEEK (115): POKE 781, PEEK (116
    ): POKE 782,255 - ( PEEK (780) +  PEEK (
    781) - 256 * ( PEEK (780) +  PEEK (781) >
    255)): REM  STORE HIMEM
100  HIMEM: MEM
110 HM = MEM + 24 * 256
120  DIM SB(52):LI = 10:LS = 10
130  GOTO 15000
140  REM  CHECK FOR LINE NO./ADDRESS
150 ADR =  PEEK (103) +  PEEK (104) * 256:FD =
    0
160  GOSUB 170:ADR = NXADR: GOTO 160
170 NXADR =  PEEK (ADR) +  PEEK (ADR + 1) * 2
    56
180 LN =  PEEK (ADR + 2) +  PEEK (ADR + 3) *
    256
190  IF LN < CN THEN  RETURN
200  IF LN > CN OR NXADR = 0 THEN  POP : RETURN

210 FD = 1: POP : RETURN
220  REM  BAR TO CONTINUE ROUTINE
230  VTAB 23: PRINT "PRESS THE ";: INVERSE : PRINT
     "  SPACE BAR  ";: NORMAL : PRINT " TO CO
    NTINUE. ";: GET X$
240  HOME : RETURN
250  REM  COMPUTE HI/LO BYTE FOR NUMBERS
260 HB =  INT (NUM / 256):LB = NUM - HB * 256
    : RETURN
270  REM  GET LINE NUMBERS FOR SUBROUTINES
280 CN = 5000: GOSUB 140:CN = 4900
290  FOR II = 1 TO 50
300 FD = 0:CN = CN + 100: GOSUB 160
310  IF FD = 0 THEN  RETURN
320  IF  PEEK (ADR + 4) <  > 178 THEN 300: REM
     CHECK FOR REM TOKEN (178)
330 SB(II) = CN: NEXT II
340  RETURN
350  REM  CHANGEABLE LIST ROUTINE ONE
360 CN = 400: GOSUB 140
370 PN$ =  STR$ (PN): IF  LEN (PN$) < 5 THEN
    PN$ = PN$ +  LEFT$ ("    ",5 -  LEN (PN$
    ))
380  FOR II = 1 TO  LEN (PN$): POKE ADR + 4 +
    II, ASC ( MID$ (PN$,II,1))
390  NEXT
400  LIST 5000:: RETURN : REM  SELF-MODIFYING
     LINE
410  REM  MOVE MEMORY ROUTINE
420  POKE 71,0: POKE 66,D1: POKE 67,D2: POKE
    60,S1: POKE 61,S2: POKE 62,E1: POKE 63,E
    2
430  POKE 58,44: POKE 59,254: CALL 65209: RETURN

440  REM  CHANGEABLE LIST ROUTINE TWO
450 CN = 500: GOSUB 140
460 PN$ =  STR$ (S1) + "," +  STR$ (S2): IF  LEN
    (PN$) < 11 THEN PN$ = PN$ +  LEFT$ ("
    ",11 -  LEN (PN$))
470  FOR II = 1 TO  LEN (PN$): POKE ADR + 4 +
    II, ASC ( MID$ (PN$,II,1)): NEXT
480  VTAB 23: PRINT "CTRL S / ANY KEY STOPS/S
    TARTS LIST": VTAB 1
490  POKE 35,21: SPEED= 125
500  LIST 5000,5099:: REM SELF-MODIFYING LIN
    E
510  POKE 35,24: SPEED= 255: RETURN
520  REM  CHECK FOR SUBROUTINE
530  FOR II = 1 TO 50
540  IF SB = SB(II) THEN  RETURN
550  NEXT : HOME
560  PRINT "YOU HAVE ENTERED A LINE NUMBER TH
    AT": PRINT "DOES NOT START A SUBROUTINE.
     TRY AGAIN.": GOSUB 220
570  POP : TEXT : HOME : GOTO 870
580  REM  GET CHOICE
590  POKE 34,18: HOME : PRINT "PLEASE ENTER T
    HE ";: INVERSE : PRINT "LINE NUMBER";: NORMAL
    : PRINT " OF YOUR": PRINT
600  PRINT "CHOICE,  PRESS ";: INVERSE : PRINT
    "RETURN";: NORMAL : PRINT " TO CONTINUE,
    ": PRINT
610  PRINT "OR ";: INVERSE : PRINT "S";: NORMAL
    : INPUT " <RETURN> TO STOP LOOKING. ";X$
    : IF X$ = "S" THEN 1500
620 SB =  VAL (X$): IF SB = 0 THEN  RETURN
630  GOSUB 520: HOME : INVERSE : PRINT "ENTER
    :";: NORMAL : PRINT "  M - MERGE WITH EX
    ISTING APPLE-"
640  HTAB 13: PRINT "SOFT PROGRAM.": HTAB 9: PRINT
    "L - TO LIST SUBROUTINE SELECTED.": PRINT

650  PRINT "PRESS ANY OTHER KEY TO CONTINUE.
    ";: GET X$: IF X$ = "L" THEN  POP : GOTO
    750
660  IF X$ = "M" THEN  POP : GOTO 970
670  RETURN
680  REM  CLEAR MEMORY
690  FOR I = MEM TO MEM + 256: POKE I,0: NEXT

700 NUM = MEM: GOSUB 250:S1 = LB:S2 = HB
710 NUM = MEM + 256: GOSUB 250:E1 = LB:E2 = H
    B
720 D1 = S1: FOR II = S2 + 1 TO S2 + 23
730 D2 = II: GOSUB 410: NEXT
740  RETURN
750  REM  LIST A SUBROUTINE
760  TEXT : HOME : FOR I = 1 TO 50
770  IF SB(I) = SB THEN 790
780  NEXT
790 S1 = SB(I): IF SB(I + 1) > 0 THEN S2 = SB
    (I + 1) - 1
800  IF SB(I + 1) = 0 THEN S2 = 14990
810  GOSUB 440: GOSUB 220: GOTO 870
820  HOME : IF SB(1) > 0 THEN 860
830  HOME : VTAB 10: HTAB 13: INVERSE : PRINT
    "PLEASE STAND BY": NORMAL : PRINT
840  GOSUB 680: REM  CLEAR MEMORY
850  HTAB 10: PRINT "FINDING SUBROUTINES.": GOSUB
    270
860 J = 1:JJ = 3
870  HOME : FOR Q = J TO JJ
880  IF SB(J) <  > 0 THEN 940
890  TEXT : HOME : VTAB 8: PRINT "THAT'S ALL
    OF THE SUBROUTINES.  IF YOU": PRINT
900  PRINT "HAVE PICKED ALL YOU WANT TO MERGE
     OR": PRINT : PRINT "ARE THROUGH REVIEWI
    NG THEM, PRESS ANY": PRINT
910  PRINT "KEY EXCEPT 'A'.  PRESSING 'A' WIL
    L": PRINT
920  PRINT "LIST THE SUBROUTINES AGAIN. ": GET
    X$: IF X$ = "A" THEN 860
930  GOTO 1500
940  IF SB(Q) = 0 THEN  GOSUB 580: GOTO 960
950 PN = SB(Q): GOSUB 350: NEXT : GOSUB 580: REM
     LIST SUBROUTINES AND GET CHOICES
960 J = J + 3:JJ = JJ + 3: TEXT : GOTO 870
970  IF PK > 0 THEN 1030
```

```
980  HOME : PRINT "DO YOU WANT TO ENTER A SEP
     ARATE LINE": PRINT : PRINT "NUMBER FOR E
     ACH SUBROUTINE AS YOU": PRINT : PRINT "S
     ELECT IT? ";: GET X$
990  IF X$ = "Y" THEN  HOME :NB = 1: GOTO 103
     0
1000  IF X$ = "N" THEN  HOME :NB = 0: GOTO 10
     20
1010  GOTO 980
1020  PRINT "PLEASE ENTER THE STARTING LINE N
     UMBER": PRINT : INPUT "FOR THE SUBROUTIN
     E(S). ";NN
1030  IF NB = 1 THEN  HOME : PRINT "PLEASE EN
     TER THE STARTING LINE NUMBER": PRINT : INPUT
     "FOR THIS SUBROUTINE. ";NN
1040  POKE 820,NB
1050  HOME : INVERSE : PRINT "STAND BY";: NORMAL
     : PRINT "  - RENUMBERING AND MOVING": PRINT

1060  PRINT "SUBROUTINE AT LINE ";SB;"."
1070  IF PK = 0 THEN 1170
1080  FOR I = 1 TO 50: IF SB = SB(I) THEN 110
     0
1090  NEXT I
1100  IF SB(I + 1) <  > 0 THEN 1150
1110 CN = SB: GOSUB 140:X = ADR:ADR = NXADR
1120  GOSUB 1130:ADR = NXADR: GOTO 1120
1130 NXADR =  PEEK (ADR) +  PEEK (ADR + 1) *
     256:LN =  PEEK (ADR + 2) +  PEEK (ADR +
     3) * 256: IF LN = 15000 THEN  POP :Y = A
     DR - 1: GOTO 1160
1140  RETURN
1150 CN = SB: GOSUB 140:X = ADR:CN = SB(I + 1
     ): GOSUB 140:Y = ADR
1160  IF (D1 + D2 * 256) + (Y - X) > HM - 5 THEN
     1430
1170  IF NB = 0 AND PK = 0 THEN NUM = NN: GOSUB
     250: POKE 824,LB: POKE 825,HB
1180  IF NB = 1 THEN NUM = NN: GOSUB 250: POKE
     820 + (PK + 1) * 4,LB: POKE 821 + (PK +
     1) * 4,HB
1190  IF PK > 0 THEN 1210
1200 NUM = MEM + 1: GOSUB 250:D1 = LB:D2 = HB
1210 CN = SB: GOSUB 140
1220  FOR I = 1 TO 50: IF SB = SB(I) THEN 124
     0
1230  NEXT I
1240  IF SB(I + 1) > 0 THEN NSB = SB(I + 1) -
     1
1250  IF SB(I + 1) = 0 THEN NSB = SB + 200
1260  FOR I = SB TO NSB
1270  GOSUB 1280:ADR = NXADR: GOTO 1270
1280 NXADR =  PEEK (ADR) +  PEEK (ADR + 1) *
     256
1290 LN =  PEEK (ADR + 2) +  PEEK (ADR + 3) *
     256
1300  IF LN <  > I THEN  POP : NEXT I: GOTO 1
     390
1310 NUM = ADR: GOSUB 250:S1 = LB:S2 = HB
1320 NUM = NXADR - 1: GOSUB 250:E1 = LB:E2 =
     HB
1330  GOSUB 410: REM  MOVE MEMORY
1340 X = D1 + D2 * 256:Y = NXADR - ADR
1350 NUM = X + Y: GOSUB 250: POKE X,LB: POKE
     X + 1,HB
1360 D1 = LB:D2 = HB
1370 NUM = NN: GOSUB 250: POKE X + 2,LB: POKE
     X + 3,HB
1380 NN = NN + LI: RETURN
1390 PK = PK + 1: IF NB = 0 THEN NUM = NN - L
     I: GOSUB 250: POKE 826,LB: POKE 827,HB
1400  IF NB = 1 THEN NUM = NN - LI: GOSUB 250
     : POKE 820,PK: POKE 822 + PK * 4,LB: POKE
     823 + PK * 4,HB
1410  IF NB = 0 THEN NN =  INT (NN / LS) * LS
     + LS
1420  TEXT : HOME : GOTO 870
1430  TEXT : HOME : VTAB 4: PRINT "SORRY - YO
     U'VE PICKED ONE MORE SUB-": PRINT
1440  PRINT "ROUTINE THAN WILL FIT IN MEMORY.
     ": PRINT : PRINT
1450  PRINT "WHEN YOU CONTINUE, ALL SUBROUTIN
     ES UP": PRINT
1460  PRINT "TO ";SB;" WILL BE ADDED TO THE P
     ROGRAM": PRINT : PRINT "YOU CHOOSE.": PRINT
1470  PRINT : PRINT "YOU'LL HAVE TO RUN SUBRO
     UTINE STORE": PRINT
1480  PRINT "AGAIN TO ADD THIS ONE (AND OTHER
     S).": GOSUB 220: GOTO 15270
1490  REM  CHECK FOR SUBROUTINE TO SAVE
1500  IF  PEEK (MEM + 1) +  PEEK (MEM + 2) =
     0 THEN  TEXT : HOME : GOTO 15050
1510  GOTO 15270
1520  TEXT : HOME : POKE 115, PEEK (780): POKE
     116, PEEK (781): END
5000  REM  ANY KEY TO CONTINUE ROUTINE
5010  VTAB 23: PRINT "PRESS ANY KEY TO CONTIN
     UE. ";: GET X$
5020  RETURN
15000  TEXT : HOME : HTAB 11: INVERSE : PRINT
     "SUBROUTINE STORAGE": PRINT : PRINT
15010  NORMAL : PRINT "THIS PROGRAM PROVIDES
     A PLACE TO STORE": PRINT : PRINT "SUBROU
     TINES TO MERGE LATER WITH AN": PRINT
15020  PRINT "APPLESOFT PROGRAM YOU'RE DEVELO
     PING.": PRINT : PRINT : PRINT "THE SUBRO
     UTINES BEGIN AT LINE 5000 AND": PRINT
15030  PRINT "AT EVEN INCREMENTS OF 100 AFTER
      THAT.": PRINT : PRINT
15040  PRINT "THIS PROGRAM SHOULD BE ON THE S
     AME DISK": PRINT : PRINT "WITH YOUR APPL
     ESOFT PROGRAM.": PRINT : PRINT "* COPYRI
     GHT 1985 MICROSPARC, INC. *": GOSUB 220
15050  INVERSE : HTAB 11: PRINT "SUBROUTINE S
     TORAGE": NORMAL : PRINT : PRINT
15060  PRINT "PLEASE ENTER THE NUMBER OF YOUR
      CHOICE:": PRINT : PRINT
15070  PRINT "   1 - REVIEW EXISTING SUBROUTI
     NES AND": PRINT
15080  PRINT "       PICK ONE OR MORE TO MERG
     E WITH": PRINT
15090  PRINT "       AN EXISTING APPLESOFT PR
     OGRAM.": PRINT : PRINT
15100  PRINT "   2 - ADD A NEW SUBROUTINE(S).
     ": PRINT : PRINT "   3 - QUIT.": PRINT
15110  GET X$:X =  VAL (X$): IF X < 1 OR X >
     3 THEN  HOME : GOTO 15050
15120  ON X GOTO 820,15130,1520
15130  HOME : VTAB 12: HTAB 13: INVERSE : PRINT
     "PLEASE STAND BY": NORMAL : GOSUB 270: REM
      FIND SUBROUTINE LINE NUMBERS
15140  IF SB(1) = 0 THEN  HOME : VTAB 8: PRINT
     "SINCE THIS PROGRAM CONTAINS NO": PRINT
     : PRINT "SUBROUTINE AT LINE 5000, BEGIN"
     : PRINT : PRINT "THE FIRST ONE AT LINE 5
     000."
15150  POKE 115, PEEK (780): POKE 116, PEEK (
     781): REM  CHANGE HIMEM BACK
15160  IF SB(1) = 0 THEN  END
15170  FOR I = 1 TO 50: IF SB(I) = 0 THEN 151
     90
15180  NEXT
15190  HOME : VTAB 4: PRINT "THE LAST SUBROUT
     INE STORED IN THIS": PRINT
15200  PRINT "PROGRAM ENDS BEFORE ";SB(I - 1)
     + 100: PRINT
15210  PRINT "BEGIN THE NEXT ONE AT ";SB(I -
     1) + 100
15220  PRINT : PRINT
15230  PRINT "YOU HAVE ";I - 1;" SUBROUTINES
     STORED IN": PRINT
15240  PRINT "THIS PROGRAM AND ROOM FOR ";51 -
     I;" MORE."
15250  POKE 115, PEEK (780): POKE 116, PEEK (
     781): REM  CHANGE HIMEM BACK
15260  END
15270 D$ =  CHR$ (13) +  CHR$ (4): REM  RETUR
     N + CONTROL D
15280  TEXT : HOME
15290  PRINT D$;"MON C,I,O"
15300  TEXT : HOME : VTAB 4: PRINT "PLEASE EN
     TER THE NAME OF THE APPLESOFT": PRINT
15310  PRINT "PROGRAM TO ADD SUBROUTINE(S) TO
     :": PRINT : INPUT F$: PRINT
15320  ONERR  GOTO 15650
15330  PRINT D$"RENAME"F$","F$
15340  POKE 216,0
15350  PRINT "AFTER THE SUBROUTINE(S) ARE ADD
     ED, DO": PRINT
```

continued on next page

LISTING 1: SUBROUTINE.STORE *(continued)*

```
15360  PRINT "YOU WANT TO CALL THE PROGRAM: "
     : PRINT : PRINT F$: PRINT
15370  PRINT "ENTER Y FOR YES, N FOR NO.": PRINT

15380  GET X$: IF X$ = "Y" THEN S$ = F$: HOME
     : GOTO 15430
15390  IF X$ <  > "N" THEN  HOME : VTAB 6: GOTO
     15350
15400  HOME : VTAB 6: PRINT "PLEASE ENTER THE
     NEW NAME YOU'D LIKE:": PRINT
15410  INPUT S$
15420  HOME
15430  PRINT D$;"OPEN TEMP.TEMP": PRINT D$;"D
     ELETE TEMP.TEMP": PRINT D$;"OPEN TEMP.TE
     MP": PRINT D$;"WRITE TEMP.TEMP"
15440  POKE 33,30:NUM = MEM + 1: GOSUB 250: POKE
     103,LB: POKE 104,HB: LIST
15450  TEXT : POKE 103,1: POKE 104,8
15460  PRINT "POKE 103,1:POKE 104,8"
15470  PRINT "DELETE SUB.TEMP"
15480  PRINT "SAVE ";S$
15490  PRINT "NOMON C,I,O"
15500  PRINT "POKE 115,PEEK(780):POKE 116,PEE
     K(781)"
15510  PRINT "TEXT:HOME"
15520  PRINT "DELETE TEMP.TEMP"
15530  PRINT D$;"CLOSE TEMP.TEMP"
15540  PRINT : HTAB 5: INVERSE : PRINT "YES -
     SOMETHING IS HAPPENING!": NORMAL
15550  GOSUB 15720: REM  CREATE TEMP PROGRAM
15560  PRINT D$;"OPEN EXEC.TEMP": PRINT D$;"D
     ELETE EXEC.TEMP": PRINT D$;"OPEN EXEC.TE
     MP": PRINT D$;"WRITE EXEC.TEMP"
15570  PRINT "LOAD ";F$
15580  PRINT "POKE 770,PEEK(175):POKE 771,PEE
     K(176)"
15590  PRINT "POKE 103,PEEK(175):POKE 104,PEE
     K(176)+1"
15600  PRINT "P=PEEK(103)+PEEK(104)*256:POKE
     P-1,0"
15610  PRINT "RUN SUB.TEMP"
15620  PRINT D$;"CLOSE EXEC.TEMP"
15630  PRINT D$;"EXEC EXEC.TEMP"
15640  END
15650 E =  PEEK (222): POKE 216,0: HOME : VTAB
     12
15660  IF E = 10 THEN  PRINT "FILE LOCKED."
15670  IF E = 6 THEN  PRINT "FILE NOT FOUND."
15680  IF E = 8 THEN  PRINT "I/O ERROR."
15690  IF E = 4 THEN  PRINT "DISK WRITE PROTE
     CTED."
15700  IF E <  > 4 AND E <  > 8 AND E <  > 6 AND
     E <  > 10 THEN  PRINT "ERROR NUMBER "E".
     "

15710  PRINT : PRINT "PRESS ANY KEY TO CONTIN
     UE.": GET Z$: PRINT : GOTO 15300
15720  REM  CREATE A TEMPORARY PROGRAM TO CHE
     CK LINE NUMBERS
15730 CN = 15790: GOSUB 140: REM   GET ADDRES
     S
15740 NUM = ADR: GOSUB 250: REM  GET LO/HI BY
     TE
15750  POKE 103,LB: POKE 104,HB: REM  RESET S
     TART OF PROGRAM POINTERS TEMPORARILY
15760  PRINT D$;"SAVE SUB.TEMP"
15770  POKE 103,1: POKE 104,8
15780  RETURN
15790  REM  TEMPORARY PROGRAM
15800 D$ =  CHR$ (13) +  CHR$ (4): PRINT D$;"
     DELETE EXEC.TEMP"
15810  HOME : VTAB 12: PRINT "PLEASE STAND BY
     - MAKING SOME CHECKS."
15820  GOSUB 16110: REM  CHECK FOR PROPER LOA
     D
15830 II =  PEEK (820): IF II < 2 THEN II = 1
15840  FOR I = 1 TO II
15850 X =  PEEK (820 + I * 4) +  PEEK (821 +
     I * 4) * 256
15860 Y =  PEEK (822 + I * 4) +  PEEK (823 +
     I * 4) * 256
15870 ADR = 2049
15880  GOSUB 15890:ADR = NXADR: GOTO 15880
15890 NXADR =  PEEK (ADR) +  PEEK (ADR + 1) *
     256
15900 LN =  PEEK (ADR + 2) +  PEEK (ADR + 3) *
     256
15910  IF NXADR = 0 THEN  POP : GOTO 15960
15920  IF LN < X THEN  RETURN
15930  IF LN = X OR LN = Y THEN 15970
15940  IF LN > X AND LN < Y THEN  POP : GOTO
     15970
15950  IF LN > Y THEN  POP
15960  NEXT I: GOTO 16070
15970  HOME : VTAB 8: PRINT "SORRY - THERE IS
     A LINE NUMBER CONFLICT": PRINT
15980  PRINT "BETWEEN THE APPLESOFT PROGRAM A
     ND THE": PRINT
15990  PRINT "LINE NUMBERS YOU PICKED FOR AT
     LEAST": PRINT
16000  PRINT "ONE SUBROUTINE AT ";LN;"."
16010  FOR I = 1 TO 3000: NEXT I
16020 D$ =  CHR$ (13) +  CHR$ (4)
16030  PRINT D$;"DELETE TEMP.TEMP"
16040  PRINT D$;"DELETE SUB.TEMP"
16050  POKE 103,1: POKE 104,8
16060  PRINT D$;"RUN SUBROUTINE.STORE"
16070  HOME
16080  POKE 103,1: POKE 104,8: POKE 175, PEEK
     (770): POKE 176, PEEK (771)
16090  PRINT D$;"EXEC TEMP.TEMP"
16100  END
16110  REM  THIS ROUTINE CHECKS TO BE SURE A
     PROGRAM WAS LOADED TO ADD SUBROUTINES TO
     .
16120 X = 0:T$ = ""
16130 ADR = 2049
16140  GOSUB 16150:ADR = NXADR: GOTO 16140
16150 NXADR =  PEEK (ADR) +  PEEK (ADR + 1) *
     256
16160  IF NXADR = 0 THEN  POP : GOTO 16180
16170  RETURN
16180  FOR I = 2 TO 6
16190 X = X +  PEEK (ADR - I)
16200  NEXT I
16210  IF X <  > 337 THEN  RETURN
16220  POP : HOME
16230  VTAB 8: PRINT "SORRY - THE PROGRAM NAM
     E YOU ENTERED": PRINT
16240  PRINT "MUST NOT BE ON THIS DISK.": PRINT
     : PRINT
16250  PRINT "PLEASE TRY AGAIN."
16260  GOTO 16010
16270  REM    THE FOLLOWING LINE MUST BE THE
     LAST LINE OF THIS PROGRAM FOR THE ROUTIN
     E AT 16010 TO WORK.
16280  REM  *+*
```

END OF LISTING 1

```
  KEY PERFECT 4.0
      RUN ON
  SUBROUTINE.STORE
=======================
 CODE       LINE# - LINE#
------      -------------
  AC9C        10 -   100
  6941       110 -   200
  7BA4       210 -   300
  7259       310 -   400
  9BB3       410 -   500
  891C       510 -   600
  A299       610 -   700
  5194       710 -   800
  82D2       810 -   900
  A263       910 -  1000
  910E      1010 -  1100
  A544      1110 -  1200
  5F77      1210 -  1300
  8B2A      1310 -  1400
  B267      1410 -  1500
  DCB3      1510 - 15040
  E0E0     15050 - 15140
  862E     15150 - 15240
  7162     15250 - 15340
  9F4B     15350 - 15440
  6A68     15450 - 15540
  9DF1     15550 - 15640
  966C     15650 - 15740
  78EB     15750 - 15840
  6A9A     15850 - 15940
  785E     15950 - 16040
  75A0     16050 - 16140
  5196     16150 - 16240
  3809     16250 - 16280
PROGRAM CHECK IS : 2100
```