● Program Development Philosophies

The IBM PC version of the PCDatagraph Data Manager program is the "flagship" version of the program. It is from this program that all other versions are developed. Development was carried out on the IBM PC, Columbia and Compaq computers using MSDOS and Microsoft's GWBASIC and IBM BASIC. By using these various "IBM compatible" computers, the confidence level of the "IBM-IBM compatibility of the code is very high.

The decision to use BASIC as the program language was determined by Hattori Corporation of America as it was felt that a finished product in this language would make the program easily transportable and easy to develop regardless of any other limitations. BASIC's interpretive mode of operation and the resultant loss of processing speed are minor limitations when transportability and the program's application are considered.

The program is written in a straightforward style which makes "following the code" fairly easy. The development version of the code is throughly commented and segmented into modules. The release version has virtually no comments, and the module "headers" have been removed. The removal of the comments for the release version is to conserve memory and not to make the code hard to read, comprehend or unintelligible.

Version Commonalty

A major goal of this computer program has been to develop the software in such a fashion that a single version of the user documentation would be common to all versions of the finished program. To this end, commonalty is preserved between the various versions by using code modules which have common functions and line numbering—that is to say, the function of each program module within any particular line number sequence, in any release version, corresponds to the code within the same sequence of line numbers in the IBM version. In many cases the code is identical. Where it is not, it is to accommodate a particular requirement of the hardware upon which the code must function or memory limitations of that particular computer.

In addition to the commonalty of the functions, line numbers and general program layout, a very high degree of variable commonalty is also preserved. Over 95% of the variables used on all versions use the same variable names and serve the same purposes. Exceptions to this scheme of functionality are rare.

Code Efficiency

In some instances, code efficiency is sacrificed for clarity. The purpose of which is to make it easier for persons other than the original author to maintain the programs should the need arise. Since this code has been transferred to computers running a version of BASIC which is not compatible with IBM's BASIC, it was necessary that very straightforward coding practices be followed, which in the long run, would be more efficient in the overall development and maintenance of the program on many different machines, with different operating systems, running different versions and types of BASIC.

## Program Version Differences

The IBM PC, MSDOS and Microsoft BASIC 1.1 are a particularly "powerful" combination of computer, operating system and language. The computer hardware and BASIC language resources of other machines, in the main, are not as "powerful." This means that these other machines do not, have all of the editing features of the IBM PC/MS-DOS version of the program. The documentation is written in such a way that program features which are not included in some versions, is noted and accounted for.

In some cases a program feature is present, but will require a different key combination to operate it. For instance, some computers do not have an escape key and a different key is used. In the case of the Commodore computer, the "Home" key is used in place of the escape key. Most computers do not have "programmable function keys," or they have fewer function keys or their function keys operate differently. The documentation covers these differences.

## User Interface

Essentially the PCDatagraph Data Manager is a mini-word processor and data base whose function is to generate and maintain user files which are loaded into the RC1000 Wrist Terminal.

The program will only permit the entry of data which will be accepted by the RC1000 Wrist Terminal. This is accomplished by extensive data checking and processing during data entry, and during file transmission to the RC1000 Wrist Terminal.

In addition, the program is "user friendly" in that it is "menu driven" and will not permit the user to dispose of or overwrite a data file unless the user explicitly confirms his intentions.

The menu choices are self-explanatory, and there is "help" when entering or creating data. Sound and flashing messages are used to alert the user to various conditions. The file's "status" (number of labels, lines free, etc.) are constantly updated and displayed as is the program's current mode of operation.

There are no commands to remember. All choices are displayed either in menus or in a "help" presentation when creating and editing the watch data file.

The program manages every aspect of the data input; "illegal" characters may not be typed; no more than 12 labels may exist in a single file; schedule alarm, weekly alarm and world time may only have 1 label each and in the event a label is deleted, all the data associated with that label is also deleted. At no time can the user enter data which is in a form unacceptable to the RC1000 Wrist Terminal or which will not be properly interpreted and displayed by the RC1000.

## General Program Operation

In order to process keystrokes in real time and maintain a formatted video display using BASIC (as opposed to machine language), certain types of programming coding practices had to be avoided: The use INPUT statements which would cause program halts, the use of extensive string handling during data input and the extensive use of program statements which are peculiar to only one type of computer or one version of BASIC.

In those instances where a BASIC statement is used which is peculiar to a particular computer, a careful study of the program module construct was made to determine which alternative statements could be used on other computers. As an example, the WHILE-WEND statements used in the IBM version are duplicated with IF-THEN statements on computers which do not have WHILE-WEND as program statements. (See Shell sort routines, lines 8600 to 8699). Although the Apple and Commodore versions are physically different than the IBM version, they are logically identical. If a logical problem occurs in one, it will occur in the others and any fix made to one will apply to all versions.

Where possible GOTOs were used instead of GOSUBs to make use of the generally faster processing times possible with GOTOs. The use of GOTOs also saved additional houskeeping processing in "bail-out" situations since RETURNs accidentally left on the stack did not have to be cleared.

The video screen is used as the input buffer when editing and creating data for the RC1000 watch file. Each time the user moves the to another line or exits the input data routine, for any reason, the input buffer is transferred to the file array. This method eliminates extensive string handling and the attendant internal system delays associated with string handling (i.e., Microsoft's "garbage collection").

Disk and cassette I/O differ substantially between the IBM and the Apple and Commodore machines. These routines, although located within the same line number sequences, differ greatly between machines and necessitated different approaches to accommodate the various I/O methods employed by each machine. On machines using a "standard" Microsoft BASIC (i.e., OPEN, CLOSE, GET, PUT, etc.), the version used on the IBM will be very similar.

Cassette I/O is only used on a few of the versions and is not a major consideration.

The user's watch data file is maintained in memory as a string array. When the watch data file is saved to disk or cassette, it is saved substantially in the same format as it is used within the RC1000. Schedule Alarms and Weekly Alarms are not sorted in year, month and time order when written to the disk or cassette file—sorting only occurs when data is sent to the watch as the continual sorting of data would impose too great a time delay and the user would perceive this as unacceptable.

The error handling capabilities of the BASICs employed on the various machines ranks from virtually useless to excellent. The IBM and Radio Shack computers have excellent error handling capabilities. The Apple has poor error handling and the Commodore's error handling is virtually useless. Because of these error handling restraints, some versions of the program are unable to detect certain classes of errors. When this situation was encountered, the user's options were limited, where possible, to inputs which could only produce results which would not have to be checked.

User files sent to the RC1000 are processed only at the time they are sent. The processing consists building the watch data directory, organizing the Schedule Alarms and Weekly Alarms into sorted order, calculating World Time Zone offsets between the user's time zone and the target time and adding the correct data headers to each block of transmitted data.

Each file sent to the RC1000 consists of a "header byte," and 80 blocks of data 25 bytes long regardless of the user's active file length. This is necessitated by the RC1000's specifications. When transmission to the watch is complete, control of the program is returned to the user.

## Machine Peculiarities

IBM PC -- This computer cannot be considered as "one machine," it is rather, a "class" of machines that includes the Columbia, the Compaq the IBM PCjr, IBM PC/XT and a host of "compatibles." Each of these machines is a close copy of the IBM PC, however there are differences in hardware and in the implementation of the DOS and some of the BASIC statements. Within this "class" of machines there are statements which behave in one way on one machine and in an entirely different way on another.

Because of this, there are code sequences which seem overly cumbersome. A casual observer may conclude that a simple statement could be substituted that would involve a considerable reduction in code and simplification of the logic employed. Although that statement may function in one way on an IBM computer running under IBM's version of Microsoft BASIC, it will function differently on a Columbia running under Microsoft's GWBASIC. Any changes to the code involving the use of different statements other than those already used, should be tested on all of the computers mentioned.

APPLE II, II+, IIe and IIc — The Apple computers, although widely hailed as "game" and "leaning" machines capable of fantastic "graphics," have a weak implementation of BASIC as an applications language. Most applications developed for this machine are written in machine language to overcome the limitations imposed upon the BASIC supplied with the machine. This machine also has some disconcerning quirks that make applications programming difficult.

It will not send "pure" ASCII characters, it uses its own "screen codes" to represent ASCII characters on the video display, I/O is not device oriented—it is, instead "slot" oriented, numerous bugs exist in its DOS which have to be patched or simply programmed around. The Apple's disk I/O is very weak and cumbersome to program.

To overcome some of these deficiencies, some machine language patches are employed. Some functions have been simplified or eliminated. Some conditions which occur and cannot be adequately handled have been left alone as they are familiar to the Apple user as "normal conditions" and are not viewed with particular alarm.

Commodore 64 -- The Commodore is several steps below the Apple in terms of functional reliability and BASIC language utility. This machine is fundamentally a "learning" machine which is extremely weak in error handling and I/O.

Most of the comments about the Apple (above) doubly apply to this machine.

## ● PCDatagraph --Variable Naming Conventions and Use

| Variable | | Name Convention |
|---|---|---|
| AP | - | A.M.-P.M. - AM/PM represented as a value: AM: AP=0 PM: AP=12. |
| AP$ | - | A.M.-P.M. - AM/PM represented as "A" or "P". |
| BL | - | Beginning Line # - used in sort routine. |
| BL$ | - | BLank string - represents a non-existent watch data file line. |
| C% | - | Counter - used in sort routine. Has no mnemonic significance. |
| CC | - | Current Character - contains the ASCII value of the current char. |
| CL | - | Current Line # - temp value for LN variable. |
| CM$( | - | CoMmand string - used to decode command (^x) key presses. |
| CN | - | Current eNd of file - temp value for EN variable. |
| CR | - | CuRsor character - contains ASCII value of the cursor character. |
| CT | - | CuT & paste buffer - contains the data type of the "cut" data. |
| CT$ | - | CuT & paste buffer string - contains "cut" string. |
| DA$( | - | DAta string - contains actual watch file data line. |
| DA( | - | Data Attribute - contains label/data attribute - 5=label, 4=data. |
| DD$( | - | Default Data string - used when inserting a new data line. |
| DL$( | - | Default Label string - used when inserting a new label line. |
| DR$ | - | DiRectory - contains the watch directory when the file is sent to the watch. |
| DT$( | - | Data Type string - used for print information associated w/each data line. |
| DT( | - | Data Type - contains actual watch file data type: 0=memo, 1=Schedule Alarm, 2=Weekly Alarm, 3=World Time. |
| DU$ | - | DUmmy string - used to FIELD World Time data records for random file I/O input. |
| DY | - | DaYs in a month - contains the number of days for each month. |
| DY$ | - | DaYs displayed - used as a temp variable in which the "days" are picked-off the screen and inserted into. |
| DY$( | - | DaY data - used in ROLL DAYS for Weekly Alarm data lines. |
| EF | - | End File - used as maximum number of lines in a watch data file. By changing this value a watch data file of any size may be created. |
| EL | - | End data Line - the maximum numer of chars which may be in a watch data line. |
| EN | - | ENd of active file - the total number of "active" watch data file lines. |
| F% | - | Counter used in sort routine. Has no mnemonic significance. |
| F1 | - | Flag 1 - used in Edit/Create - 0=type mode 1=insert mode. |
| F2 | - | Flag 2 - indicates whether a data line has been altered. 0=data unaltered 1=data altered. |
| F3 | - | Flag 3 - indicates whether the file has been altered. 0=file un-altered 1=file altered. Reset to 0 by LOAD and SAVE routines. |
| FI$ | - | FIle name string - contains the name of the current active file name. |
| G% | - | Counter - used in sort routine. Has no mnemonic significance. |
| HL$( | - | HeLp strings - used to print "help" in Edit/Create. |
| HR | - | HouR value - contains the value derived from HR$ |
| HR$ | - | HouR string - used when picking-off data from data line |

| | | |
|---|---|---|
| IK | - | InKey value - the ASCII value of IK$ |
| IK$ | | InKey string - used in keyboard scan - IK$ contains the user's last key press. |
| IL | - | Input Length - maximum length of a data line. |
| IP$ | - | InPut string - used for user input. |
| J | - | Counter - used in sort routine. Has no mnemonic significance. |
| LA | - | LAbel "found" - used when searching for a label in the file list. Contains the line number of the label when a label is "found." |
| LA( | - | LAbel - contains the number of labels for each data type. LA(0) contains the number of memo labels; LA(1) the number of Schedule Alarm labels, LA(2) the number of Weekly Alarm labels, and LA(3) the number of World Time labels. |
| LL | - | List Length - used in WRITE TO WATCH routine. |
| LN | - | Line Number - contains the currently "active" line number. |
| LN$( | - | Line Number string - used when printing text area display. |
| LO | - | LOcation - used to define the position of a PRINT variable or the position of data to be picked-off from a data line. |
| MG | - | MessaGe number - contains the number of a message to be printed. Always used with MG$(n). |
| MG$( | - | MessaGe string - contains a message or error message. |
| MI$ | - | MInutes string - contains results of "minutes" pick-off from the screen or watch data file. |
| MO | - | MOnth - contains the value of the month derived from the system date string. |
| MO$ | - | MOnth string - contains the results of "month" pick-off from the watch data file. |
| MU | - | MenU number value - contains the value of the menu to be printed. 0=Main Menu 1=System Menu 2=Insert Menu 3=Load Watch |
| MU$( | - | MenU string - contains the menu displays. |
| N% | - | Counter - used in sort routine. Has no mnemonic significance. |
| OS | - | Off-Set - used in sort routine. |
| PO | - | POsition - contains the current cursor position in the active data data line in Edit/Create mode. |
| PR | - | PRint utility variable - contains value of picked-off data which will later be printed. Also used as a utility variable and has no mnemonic significance when used for this purpose. |
| PR$ | - | PRint utility string - used to construct strings which will later be PRINTed or LPRINTed. |
| PR$( | - | PRint stings - used when LPRINTing the watch data file to the line printer. |
| PZ | - | PoZition - contains the absoulute screen address of the first character of the active data line in text area display. |
| Q1 | - | Question 1 - used for testing - 0=No 1=Yes. |
| RQ | - | ReQuest - used in INSERT routine to define the data type of the requested insert. 4=data 5=label |

| | | |
|---|---|---|
| SA | - | Screen Attribute - used to define attribute (hi-intensity, blink, etc.) of a message to be printed. |
| SD( | - | Sort Data - used in sort routine. |
| SEG | - | SEGment - NOT A PROGRAM VARIABLE - this is a portion of a BASIC program statement and is used "like" a variable. It is used to define the location of the video display memory for BASIC's BLOAD, PEEK, and POKE statements. |
| SN( | - | Sort Number - used in sort routine. |
| SP | - | SPace - used to define the number of spaces which are to be printed to the video display. |
| TZ | - | Time Zone - the value of the current time zone - input by user. default value = 16. |
| V1$ | - | Validation string 1 - used to validate user input. |
| V2$ | - | Validation string 2 - used to validate user input. |
| V3$ | - | Validation string 3 - used to validate user input. |
| V4$ | - | Validation string 4 - used to validate user input. |
| WH$ | - | World time Hours string - contains the "hours" data read from the file: "TIMEZONE.DAT". |
| WT | - | World Time counter - used in ROLL WORLD TIME FROM LIST and FIND CITY routines. |
| WT$ | - | World Time string - contains the data read from the file: "TIMEZONE.DAT". |
| X | - | Counter - has no mnemonic significance. |
| Y | - | Counter - has no mnemonic significance. |
| YN$ | - | Yes/No validation string - used to validate user "yes/no" responses to prompts. |
| YR | - | YeaR - contains the value of the year portion of the system date date string. |
| Z | - | Counter - has no mnemonic significance. |

## Variable Names Specific To Commodore 64 Version (1.00 c64)

| | | |
|---|---|---|
| ST | - | Reserved Commodore variable - I/O error status |
| TI$ | - | Reserved Commodore variable - Time Functions |
| CM | - | CoMmand string length - the length of the command string |
| V0 | - | Validation variable 0 - a defined function (DEFFN) |
| V1 | - | Validation variable 1 - a defined function (DEFFN) |
| V2 | - | Validation variable 2 - a defined function (DEFFN) |
| VT$ | - | Vertical Tab string - contains chrs used to print line feeds to position the cursor vertically |
| TP | - | TyPe of device used for I/O: 8=Disk 1=Tape 4=Printer |

## Variable Names Specific to Apple II Version (1.00 ap)

| | | |
|---|---|---|
| HI | - | High Memory address |
| HS | - | Help Screen number - used to toggle help screens |
| HT | - | Horizontal Tab position - used to position cursor prior to printing |
| IP | - | InPut value - also used as a utility variable |
| SP$ | - | SPace string - a string containg ASCII space characters |
| SLOT | - | Currently active I/O slot |
| YEAR$ | - | A string containing the current year value |

## • PCDatagraph — Data Manager Program Routine Locations

### • RC1000.BAS

| Routine | Line Number |
|---|---|
| First Program Line | 10 |
| Program Title Line | 51 |
| Initialization | 1005 |
| Initialize Command Strings for 'Edit' Module | 1300 |
| Transfer Program Control to Mainline | 1501 |
| Draw Screen | 15010 |
| Message Strings | 15805 |
| Menu Data Initialization | 16510 |
| Misc Strings | 17160 |
| Help Screen | 17405 |
| Last Program Line | 17999 |

### • RC1000.SYS

| Routine | Line Number |
|---|---|
| First Program Line | 10 |
| Program Title Line | 51 |
| Second Stage Initialization | 1501 |
| Main Line Program | 2001 |
| Transmit to Watch | 2101 |
| Set Time Zone | 2201 |
| Print Watch Data | 2610 |
| System Menu | 2910 |
| Exit Program | 3110 |
| Print Menus | 3310 |
| Menu Input - "Choose a number" | 3510 |
| Text Entry -- Edit/Create Watch Data | 3710 |
| Keyboard Scan | 3740 |
| Cursor Left -- Left Arrow Key | 4010 |
| Cursor Right -- Right Arrow Key | 4090 |
| Cursor to Beginning of Line -- <Home> Key | 4170 |
| Cursor to End of Line -- End Key | 4230 |
| Backspace and Erase -- Backspace Key | 4290 |
| Delete Character -- Delete Key | 4380 |
| Erase From Cursor Position to EOL -- ^E | 4480 |
| Insert Character -- Insert Mode On | 4570 |
| Toggle Insert Mode On/Off -- Insert Key | 4690 |
| Insert Off -- Subroutine | 4760 |
| Scroll Up -- Up Arrow Key Pressed | 4830 |
| Scroll Down -- Down Arrow/CR Key | 4950 |
| Current Line (CL$) Into the Array | 5070 |
| Page Up -- PgUp Key | 5180 |
| Page Down -- PgDn Key | 5290 |
| Jump Next Label -- ^J | 5400 |
| Restore Changes to DATA -- ^R | 5530 |
| Exit to Main Menu -- <Esc> Key | 5610 |
| Print Text Area | 5690 |
| Delete Label/Data Line(s) -- ^D | 6070 |
| Insert Label/Data Line -- ^I | 6680 |
| Find a City in World Time List -- ^F | 7301 |

● PCDatagraph — Data Manager Program Listing — RC1000.BAS

```
10    GOTO 1020
20    SAVE"A:RC1000.BAS" : END
29  '
50   REM *******************************************************************
51   REM                 RC1000.BAS    Version 1.00i/ms
52   REM *******************************************************************
53   '
54   '        By H.C. Pennington
55   '        10/01/84
56   '
57   '        Copyright (c) 1984, Hattori Corporation of America, Inc.
58   '        1330 West Walnut Parkway, Compton, CA 90220
59   '
60   '
999 '
1000 REM *******************************************************************
1005 REM INITIALIZATION
1010 REM *******************************************************************
1015 '
1020 CLS                                'Clear the screen
1025 DIM DA$(81), LN$(81), MG$(35)      'Dimension array variables
1026 DIM MU$(40),HL$(41)                'Dimension array variables
1027 DIM DT(85), DA(85), SD(85), SN(85) 'Dimension array variables
1029                                    '(Below...) Determine video type
1030 PRINT"A":DEF SEG=&HB000:IF PEEK(0)<>65 THEN X=1:DEF SEG=&HB800
1031 IF X=1 THEN FI$="GRAPHICA.PIX" ELSE FI$="MONOCHRM.PIX" 'Set bboard file
1032 '- - - - - - - - - - - - -
1033 LA(3) = 0                          'Num of WORLD TIME lbls (0-1)
1035 BLOAD FI$                          'Load billboard from disk
1036 '
1037 LOCATE 12,39:PRINT LEFT$(TIME$,5); 'Print hrs/minutes on watch
1038 LOCATE 12,46:PRINT RIGHT$(TIME$,2);'Print seconds on watch
1039 IK$=INKEY$:IF IK$="" THEN 1037     'Scan keyboard for key press
1040 LOCATE 21,50:COLOR 31:PRINT"Initilizing program ...     "; 'Print init msg
1041 COLOR 7                            'Restore video to normal
```

```
1042 '- - - - - - - - - - - - - - -
1043 KEY OFF                             'Turn off line 25 display
1045 CR = 220:CR$ = CHR$(219)            'Cursor chars: type, insert
1046 CC = 0                              'Current character
1047 PZ = 2776                           'Current line position
1048 PO = PZ                             'Current screen position
1049 IL = PZ + 46                        'Maximum string length
1050 EN = 80                             'Watch: end of data
1051 AP=0                                'AM/PM value
1052 AP$=""                              'AM/PM string
1053 BL=0                                'Beginning Line # (for sort routine)
1054 CZ=0                                'Sort variable
1055 FZ=0                                'Sort variable
1056 GZ=0                                'Sort variable
1057 NZ=0                                'Sort variable
1058 X=0:J=0:Y=0                         'Loop counters
1059 LO=1                                'Print locater (used in LOCATE stmnts)
1060 LL=0                                'List length-used in WRITE watch data
1061 MG=0                                'Message number-[ used with MU$(MU) ]
1062 MU=0                                'Menu number-[ used with MU$(mu) ]
1063 PR=0                                'Print utility variable
1064 'PR$(                               'Print utility-used for LPRINTing file
1065 EL=0                                'End of list
1076 EF = 80                             'End file-the max # of lines in a file
1077 MU = 0                              'Set menu to Main Menu
1078 LN = 1                              'Current line# of data
1080 CL = 0                              'Temp line# to print text area
1085 'DA(                                 Data attribute: 4=labl  5=dat
1090 'DT(                                 Data type: 0=Me 1=ShA  2=W'kA 3=WTime
1095 F1 = 0                              'Flag: 0=nrml input  1=insert
1100 F2 = 0                              'Flag: 0=no chng to data 1=data altd
1105 F3 = 0                              'Flag: file altered & NOT saved
1120 'FR(                                 0=lines free 1=labels free
1125 PR$=""                              'Set prnt string length
1130 IP$=""                              'INPUT string
1135 'DL$(                                Deflt labels; data types 0-3
1140 'DD$(                                Deflt data; data types 0-3
1145 'IK$=""                              Inkey input string
1150 IK = 0                              'Val of IK$, if any
1155 LA=0                                '"Found" LAbel list address
1160 '
1161 WT=1                                'World time record counter
1162 'WT$                                'Wrld time string - init when FIELDed
1163 'WX$                                'Wrld time offsets - init when FIELDed
1166 CN=0                                'Cur line # used for wrking storage
1168 DD=0                                'Day-date validation [not used on IBM]
1170 DY=0                                'Days-utility var [ROLL DAYS]
1172 DY$=""                              'Days-utility var [WRITE WATCH DATA]
```

```
1174 HH=0                                ´Hours-utility var
1176 HR=0                                ´Hours-utility var
1178 HR$=""                              ´Hours-utility var
1180 OS=0                                ´Off-set [used in DELETE]
1182 MI$=""                              ´Minutes-utility var
1184 MM=0                                ´Minutes-utility var
1186 MO=0                                ´Months-utility var
1188 MO$=""                              ´Months-utility var
1190 Q1=0                                ´Question?  1=YES  0=NO
1192 RQ=0                                ´Request! #=request number
1194 SA=0                                ´Scrn attrib [ used with COLOR stmnt ]
1196 SP=0                                ´Def# of spcs [ used w/SPC(SP) stmt ]
1198 SS=0                                ´Seconds-utility variable
1200 YR=0                                ´Year-utility variable
1202 YY=0                                ´Year-utility variable
1204 DR$=""                              ´Watch directory
1205 CT$=""                              ´Cut & Paste buffer
1206 CT=0                                ´Data type of "cut" buffer contents
1280 ´
1285 V1$="EATONSHIRDLU BCDFGJKMPQVWXYZ1234567890:*/#&+-=?." ´Validation string
1286 V2$=V1$+" "                        ´Validation string
1287 V4$=CHR$(8)+CHR$(13)+CHR$(27)       ´Validation string
1289 YN$="YyNn"+CHR$(27)                 ´"YES/NO" validation string
1290 ´
1295 REM ----------------------
1300 REM INITIALIZE COMMAND STRINGS FOR ´EDIT´ MODULE
1305 ´
1310 RESTORE 1330                        ´Set next data to be read
1315 FOR X=1 TO 13                       ´Build command string
1320  READ Y : CM$(1)=CM$(1)+CHR$(Y)     ´Put val into command string
1325 NEXT X                              ´Loop until done
1330 DATA 13, 08, 27           :         ´CR  BS  Esc
1335 DATA 04, 05, 09, 10       :         ´^D  ^E  ^I  ^J
1340 DATA 18, 20, 6, 19, 2, 14 :         ´^R  ^T  ^F ^S ^B ^N
1345 ´- - - - - - - - - - - - -
1350 RESTORE 1370                        ´Set next data to be read
1355 FOR X=1 TO 10                       ´Build command string
1360  READ Y : CM$(2)=CM$(2)+CHR$(Y)     ´Put val into command string
1365 NEXT X                              ´Loop until done
1370 DATA 75, 77, 72, 80       :         ´Lt  Rt  Up  Dn
1375 DATA 82,83                :         ´Insert, Delete
1380 DATA 73,81                :         ´PgUp, PgDn
1385 DATA 71,79                :         ´Home, End,
1390 ´- - - - - - - - - - - - -
1395 FOR X=0 TO 9                        ´Set loop
1400  KEY X+1, MID$("[]{}<>()|^",X+1,1) ´Program funct keys
1405  CM$(3)=CM$(3)+MID$("[]{}<>()|^",X+1,1) ´Build command string
1410 NEXT X                              ´Loop until done
1415 ´
```

```
1500 REM ***************************************************************
1501 REM TRANSFRER PROGRAM CONTROL TO MAINLINE
1502 REM ***************************************************************
1503 '
1505 GOSUB 15820                    'Initialize all strings
1506 GOSUB 15030                    'Draw the screen
1510 CHAIN "RC1000.SYS",1505,ALL    'Chain mainline @ 2000 w/vars intact
1520 '
15000 REM **************************************************************
15010 REM DRAW SCREEN
15011 REM **************************************************************
15020 '
15030  COLOR 15:CLS                 'Set display to hi-intensity
15040 LOCATE 1,1:PRINT CHR$(218)+STRING$(78,196)+CHR$(191) 'Top of big box
15050 FOR X = 2 TO 10               'Set loop
15060   LOCATE X,1                  'Position & print graphics
15061   PRINT CHR$(179)+STRING$(78, 32)+CHR$(179) 'Sides of big box
15070 NEXT X                        'Loop until done
15080 LOCATE 11,1                   'Set position & print graphics
15081 PRINT CHR$(192)+STRING$(28,196)+" Press  <Esc> to exit "+
            STRING$(28,196)+CHR$(217)
15090  COLOR 7                      'Turn off hi-intensity
15100 '
15110 LOCATE 12,1                   'Set position & print graphics
15111 PRINT SPACE$(27)+CHR$(218)+STRING$(24,196)+CHR$(191)
15120 FOR X = 13 TO 23              'Set loop
15130   LOCATE X,1                  'Position & print graphics
15131   PRINT SPACE$(27)+CHR$(179)+STRING$(24, 32)+CHR$(179)
15140 NEXT X                        'Loop until done
15150 LOCATE 24,1                   'Set position & print graphics
15151 PRINT SPACE$(27)+CHR$(192)+STRING$(24,196)+CHR$(217);
15160 RETURN                        'Return to caller
15170 '
```

```
15500 REM *******************************************************************
15805 REM MESSAGE STRINGS
15810 REM *******************************************************************
15815 '
15820 MG$(0)  = "          Typing Mode"
15825 MG$(1)  = "          Insert Mode"
15830 MG$(2)  = "  Time Zone:"
15835 MG$(3)  = " Lines Free:"
15840 MG$(4)  = "Labels Free:"
15845 MG$(5)  = "Active File:"
15850 MG$(6)  = "Inserting label/data"
15855 MG$(7)  = "Watch full"
15860 MG$(8)  = "Already have 12 labels - unable to insert"
15865 MG$(9)  = "Enter new time zone: "
15870 MG$(10) = "Current file not saved - continue (Y/N)?"
15875 MG$(11) = "Press a number key."
15880 MG$(12) = "Data below label will be lost-continue (Y/N)?"
15885 MG$(13) = "Deleting label/data"
15890 MG$(14) = "Are you sure (Y/N)?"
15895 MG$(15) = "Saving data file"
15900 MG$(16) = "Loading data file"
15905 MG$(17) = "I/O error -- correct and retry"
15910 MG$(18) = "Enter time (HH:MM:SS):"
15915 MG$(19) = "Enter date (MM/DD/YY):"
15920 MG$(20) = "     Invalid input - re-enter     "
15925 MG$(21) = "Writing data to PCDatagraph..."
15930 MG$(22) = "Ready printer-press <Enter> when ready"
15935 MG$(23) = "Press <Esc> to exit"
15940 MG$(24) = "SEIKO PCDatagraph Watch Data File: "
15945 MG$(25) = "Enter file name: "
15950 MG$(26) = "File already exists -- use it anyway (Y/N)?"
15955 MG$(27) = "Can't LOAD/SAVE -- no file name"
15960 MG$(28) = " Find city: "
15965 MG$(29) = "Search for: "
15970 MG$(30) = "Can't find target"
15975 MG$(31) = "Searching ..."
15980 MG$(32) = "Set PCDatagraph to RECEIVE mode"
15985 MG$(33) = "Ready Cassette -- Press <Enter> when ready"
16000 BL$     = "   3     --- --- ---      3                          "
16010 DL$( 0) = "---------- --- MEMO ---"
16020 DL$( 1) = "SCHEDULE    -- ALARM ---"
16030 DL$( 2) = "WEEKLY      -- ALARM ---"
16040 DL$( 3) = "------------ WORLD  TIME"
16050 DD$( 0) = " - - - - -             "
16060 DD$( 1) = " - - - - -  01/01 A12:00"
16070 DD$( 2) = " - - - - -  0 SUN A12:00"
16080 DD$( 3) = " - - - - -   01:00 = TZ "
16090 '
```

```
16500 REM ***************************************************************
16510 REM MENU DATA INITIALIZATION
16520 REM ***************************************************************
16530 '
16540 RESTORE 16620                      'Set next data to read
16550 FOR X = 0 TO 23                    'Set read loop
16560  READ MU$(X)                       'Load array
16570 NEXT X                             'Loop until done
16580 '
16590 REM ======================
16600 REM #0   MAIN MENU
16610 '
16620 DATA "      Main Menu                               "
16630 DATA "1.  Edit Create Watch Data                    "
16640 DATA "2.  Load PCDatagraph (Load Watch)             "
16650 DATA "3.  Print Watch Data                          "
16660 DATA "4.  System Menu                               "
16670 DATA "5.  Quit Program                              "
16720 '
16730 REM ======================
16740 REM #1   SYSTEM MENU
16750 '
16760 DATA "      System Menu                     "
16770 DATA "1.  Name File                         "
16780 DATA "2.  Change Time/Date                  "
16790 DATA "3.  Load File                         "
16800 DATA "4.  Save File                         "
16810 DATA "5.  Change Time Zone                  "
16860 '
16870 REM =====================
16880 REM #2   INSERT LABEL/DATA MENU -- ^I
16890 '
16900 DATA "      Insert Menu (^I)                        "
16910 DATA "1.  Add memo Label                            "
16920 DATA "2.  Add Memo Data                             "
16930 DATA "3.  Add Schedule Alarm Data/Label             "
16940 DATA "4.  Add Weekly Alarm Data/Label               "
16950 DATA "5.  Add World Time Data/Label                 "
17000 '
17010 REM ======================
17020 REM #3   LOAD WATCH
17030 '
17040 DATA "             Load Watch              "
17050 DATA "Make sure  Seiko RS-1000 Wrist Terminal"
17060 DATA "is properly connected  to the computer.
17070 DATA "                                     "
17080 DATA " Press <Enter> to begin loading watch. "
17090 DATA "                                     "
17140 '
```

```
17150 REM ****************************************************************
17160 REM MISC STRINGS
17161 REM ****************************************************************
17170 '
17180 DT$(0)= "Memo        "
17190 DT$(1)= "Sched. Alarm"
17200 DT$(2)= "Weekly Alarm"
17210 DT$(3)= "World Time  "
17220 DT$(5)= "Label: "
17230 DT$(4)= " Data: "
17240 PR$(0)= "Me"
17250 PR$(1)= "SA"
17260 PR$(2)= "WA"
17270 PR$(3)= "WT"
17280 PR$(5)= "L: "
17290 PR$(4)= "D: "
17300 '
17310 DY$(0)= "0 SUN "
17320 DY$(1)= "1 MON "
17330 DY$(2)= "2 TUE "
17340 DY$(3)= "3 WED "
17350 DY$(4)= "4 THU "
17360 DY$(5)= "5 FRI "
17370 DY$(6)= "6 SAT "
17380 '
```

```
17400 REM *****************************************************************
17405 REM HELP SCREEN
17406 REM *****************************************************************
17410 '
17420 '
17425 HL$(0)="  Home":HL$(1)="-Beg Line"
17430 HL$(2)= "  End":HL$(3)="-End Line "
17435 HL$(4)= "  Ins":HL$(5)="-Ins Char "
17440 HL$(6)= "  Del":HL$(7)="-Del Char "
17445 HL$(8)= "  Lt/Rt":HL$(9)="-Lft/Rgt"
17450 '---------
17455 HL$(10)="  Up":HL$(11)="-Up 1 Line "
17460 HL$(12)="  Dn":HL$(13)="-Dwn 1 Line"
17465 HL$(14)="  CR":HL$(15)="-Dwn 1 Line"
17470 HL$(16)="  PgUp":HL$(17)="-Up Page "
17475 HL$(18)="  PgDn":HL$(19)="-Down Pge"
17480 '---------
17485 HL$(20)="  ^I":HL$(21)="-ns Line   "
17490 HL$(22)="  ^D":HL$(23)="-el Line   "
17495 HL$(24)="  ^E":HL$(25)="-rase Line "
17500 HL$(26)="  ^J":HL$(27)="-mp to Lbl "
17505 HL$(28)="  ^R":HL$(29)="-estre Line"
17510 '---------
17515 HL$(30)="  ^F":HL$(31)="-ind City  "
17520 HL$(32)="  ^S":HL$(33)="-earch Data"
17525 HL$(34)="  ^B":HL$(35)="-eg of Data"
17530 HL$(36)="  ^N":HL$(37)="-End File  "
17535 HL$(38)="  ^T":HL$(39)="-ab Rt/Lt  "
17540 '---------
17545 HL$(40)=" AM-PM  AM-PM   Hours  Mins    Month  Days    "
17550 HL$(41)=" Cut    Paste   WTUp   WTDn "
17555 '---------
17800 RETURN
17999 '----------------------- END PROGRAM -----------------------------
```

● PCDatagraph — Data Manager Program Listing — RC1000.SYS

```
10   RUN"RC1000.BAS
20   SAVE"A:RC1000.sys":RUN 29  '
50   REM ************************************************************
51   REM              RC1000.SYS   Version 1.00i/ms
52   REM ************************************************************
53   '
54   '        By H.C. Pennington
55   '        10/01/84
56   '
57   '        Copyright (c) 1984, Hattori Corporation of America, Inc.
58   '        1330 West Walnut Parkway, Compton, CA 90220
59   '
60   '
99   '
1500 REM ******************************************************************
1501 REM SECOND STAGE INITIALIZATION
1502 REM ******************************************************************
1503 '
1505 DEF SEG=&HB000:IF PEEK(0)<>218 THEN DEF SEG=&HB800 'Def seg for PC type
1510 OPEN"R",3,"TIMEZONE.DAT",20          'Open world time file
1520 FIELD #3,18 AS WT$:WT=1              'Field world time record
1525 FIELD #3,13 AS DU$,2 AS WH$          'Field WT hours
1610 GOSUB 10130                          'Create an "empty" file
1620 GOSUB 5710                           'Display file in text area
1999 '
```

```
2000 REM ***********************************************************************
2001 REM MAIN LINE PROGRAM
2002 REM ***********************************************************************
2003 '
2004 '- - - - - - - - - MAIN MENU
2010 MU=0:CLOSE 1,2                      'Set Menu=0 & clse files 1 & 2
2020 GOSUB 3340                          'Print the menu
2030 COLOR 31                            'Set display to hi-blink
2040 SA=31:MG=11:GOSUB 13430             'Print prompt
2050 COLOR 7                             'Hi-blink off
2060 GOSUB 3540                          'Get user input
2061 IF IK$=CHR$(27) THEN IK$="1"        'Esc was pressed-load IK$
2062 '
2070 ON VAL(IK$) GOTO 3760, 2105, 2640, 2940, 3140 'Jump on val of input
2080 '                 Edit  Xmit  Prnt  Sys   Exit
2081 '
2090 SOUND 37,1:GOTO 2060                'BLAT & rtn to KB scan
2099 '
2100 REM ***********************
2101 REM TRANSMIT TO WATCH
2102 REM ***********************
2103 '
2105 SP=20:MU=3:GOSUB 3340:SP=0          'Set menu psn & print menu
2106 MG=32:GOSUB 13430                   'Print "set receive" msg
2110 IK$=INKEY$:IF IK$="" THEN 2110      'Scan KB
2115 ON INSTR(CHR$(13)+CHR$(27),IK$) GOTO 7605, 2130 'Jump on input value
2120 SOUND 37,1:GOTO 2110                'BLAT & rtn to KB scan
2130 GOTO 2010                           'Vector from 2115-Jmp to XMIT
2131 '
2200 REM ***********************
2201 REM SET TIME ZONE
2202 REM ***********************
2203 '
2205 MG=9:GOSUB 13430:MG=20              'Print "TZ" msg & set err msg
2206 IP$="":LO=52:GOSUB 13930:GOSUB 14505'Wait for user input
2208 IF IP$="" THEN 2950                 'EXIT if <Enter> pressed
2210 IF VAL(IP$)<1 OR VAL(IP$)>24 THEN GOSUB 13230:GOTO 2205 'Invalid input
2212 TZ=VAL(IP$):GOSUB 11330             'Update status display
2214 GOTO 2950                           'EXIT to system menu input
2599 '
```

```
2600 REM ***********************
2610 REM PRINT WATCH DATA
2620 REM ***********************
2630 '
2640 OPEN"lpt1:" AS #2:WIDTH #2,132:MG=22:GOSUB 13430 'Opn Lptr & Prtr msg
2660 IK$=INKEY$:IF IK$="" THEN 2660      'Scan KB for user input
2670 ON INSTR(CHR$(13)+CHR$(27),IK$) GOTO 2700, 2800 'Jmp on user input
2680 GOTO 2640                           'Illegal input-rtn to KB scan
2700 PRINT #2, STRING$(45-(LEN(MG$(24)+FI$)/2),32)+MG$(24)+FI$ 'Print header
2710 PRINT #2, STRING$(10,32)+STRING$(70,"=") 'Finish header
2720 LPRINT                              'Print blank line
2730 X=1                                 'Set starting line#
2740 '- - - - - BUILD PRINT STRINGS
2745 '
2750 IF X<=EN THEN PR$=STRING$(10,32)+LN$(X)+">"+DA$(X)+"<"+PR$(DA(X))
               +PR$(DT(X))
2760 IF (X+40)<=EN THEN PR$=PR$+STRING$(3,32)+LN$(X+40)+">"+
                      DA$(X+40)+ "<"+PR$(DA(X+40))+PR$(DT(X+40))
2770 '- - - - - - - - - - - - -
2775 IF X>EN THEN 2810                   'EXIT: end of active file
2780 PRINT #2, PR$                       'Print data
2790 IK$=INKEY$:IF IK$=CHR$(27) THEN 2810'Bail out if Esc pressed
2800 X=X+1:IF X<=40 THEN 2750            'Loop until done
2810 LPRINT STRING$(63-X,13)             'Simulate form feed
2820 GOSUB 13630:CLOSE 2                 'Clear msg & close LPT1
2830 GOTO 2010                           'EXIT to menu input
2840 '
2900 REM ***********************
2910 REM SYSTEM MENU
2920 REM ***********************
2930 '
2940 MU=1:GOSUB 3340                     'Print menu
2950 SA=31:MG=11:GOSUB 13430             'Print prompt
2960 GOSUB 3540                          'Get user input
2970 IF IK$=CHR$(27) THEN 2010           'EXIT: M-Menu if Esc pressed
6980 ON IK GOTO 11530, 10330, 3030, 3040, 2205 'Jump on user input
2990 '          FNam   T&D ' 'Load  Save  Exit
3000 GOTO 2960                           'Scan again if invalid input
3010 ' - - - - - - - - - - - - -
3020 MU=0                                'Reset menu: M-Menu
3030 GOSUB 9070:GOTO 2950                'GSub LOAD & EXIT: S-Mnu input
3040 GOSUB 9330:GOTO 2950                'GSub SAVE & EXIT: S-Mnu input
3050 '
3100 REM ***********************
3110 REM EXIT PROGRAM
3120 REM ***********************
3130 '
3140 MG=14:GOSUB 13430                   'Print "RU sure?" prompt
3150 IK$=INKEY$:IF IK$="" THEN 3150      'Scan KB for response
3160 ON INSTR(YN$,IK$) GOTO 3180,3180,3190,3190,3190 'Jump on Y or N
3170 SOUND 37,1:GOTO 3150                'BLAT & rtn to KB scan
3180 CLS:NEW                             'Clear screen & NEW program
3190 GOSUB 13630                         'Clear msg area
3200 GOTO 2010                           'EXIT to S-Menu input
3210 '
```

```
3300 REM ***********************
3310 REM PRINT MENUS
3320 REM ***********************
3330 '
3340 LOCATE 2,2:COLOR 15
3341 IF SP=0 THEN SP=28                    'Set menu position if not already set
3345 IF F2>0 THEN GOSUB 5090               'If data altered, insert into array
3350 MU = MU*6                             'Set Mnu array to prnt menu requested
3355 PRINT SPC(SP)+MU$(MU);:COLOR 7        'Print title in Hi-intsty
3356 LOCATE 8,2:PRINT SPC(77);             'Clear line #8
3360 FOR X = 1 TO 5                        'Set loop to print menu
3370  LOCATE X+2,2:PRINT SPC(SP) MU$(MU+X); 'Print menu
3380 NEXT X                                'Loop until done
3390 MU = MU * .1                          'Reset val of MU
3400 RETURN                                'Return to caller
3410 '
3500 REM ***********************
3510 REM MENU INPUT - "Choose a number"
3520 REM ***********************
3530 '
3540 V3$ = "12345"                         'Input validation string
3550 '
3560 IK$=INKEY$:IF IK$="" THEN 3560        'Scan KB
3570 IF IK$=CHR$(27) THEN RETURN           'Return if Esc is pressed
3580 IF INSTR(V3$,IK$) THEN IK=VAL(IK$):RETURN 'Load IK w/val & rtn to caller
3590 SOUND 37,1                            'BLAT on invalid iput
3600 GOTO 3560                             'Rtn to scan KB
3610 '
3700 REM **********************************************************************
3710 REM TEXT ENTRY -- EDIT/CREATE WATCH DATA
3720 REM **********************************************************************
3730 '
3740 '--------KEYBOARD SCAN
3750 '
3760 LOCATE 18,1:PRINT MG$(F1);            'Print input status mode
3770 GOSUB 14210                           'Print help screen
3780 CC=PEEK(PO)                           'Get current char on screen
3790 POKE PO,CC:POKE PO+1,7                'Put current char on screen
3800 IK$=INKEY$                            'Scan KB
3810 ON LEN(IK$) GOTO 3890, 3840           'Jmp on length of inkey value
3820 IF IK$="" THEN POKE PO+1,15:POKE PO,CR:GOTO 3790 'IK$ empty; flash & scan
3830 '- - - - - - - - - - - - - -
3840 IK$=RIGHT$(IK$,1)                     'Get right most bye of IK$
3850 ON INSTR(CM$(2),IK$) GOTO 4030, 4110, 4970, 4850, 4710, 4400, 5310,
                               5200, 4190, 4250
3860 '                         Left, Rght, Up  , Down, Ins , Del , PgUp,
                               PgDn, Home, End
3870 SOUND 37,1:GOTO 3780                  'BLAT & return to KB scan
3880 '- - - - - - - - - - - - - -
3890 ON INSTR(CM$(1),IK$) GOTO 4850, 4310, 5630, 6090, 4500, 6700, 5420,
                               5550, 7575, 7305, 7405, 7485, 7495
3900 '                         CR  , BSpc, Esc , ^D  , ^E  , ^I  , ^J  ,
                               ^R  , ^T  , ^F  , ^S  , ^B  , ^N
3910 '
3920 ON INSTR(CM$(3),IK$) GOTO 12030,12030,12230,12430,12630,
                               12830, 8755, 8785, 7505, 7510
3930 '                         F1  , F2  , F3  , F4  , F5  ,
                               F6  , F7  , F8  , F9  , F10
```

```
3940 '
3950 IF INSTR(V1$,IK$) THEN 3955 ELSE 3980 'Test for valid character
3955 IF F1 THEN 4590                     'If insert is on, go do it
3960 POKE PO,ASC(IK$)                    'Put character on the screen
3965 F2=1:IF F2 THEN F3=1                'Set flag: file not SAVEd
3970 PO=PO+2:IF PO>IL THEN PO=PZ:GOSUB 4780 'Calc new scn pos'n & chk for len
3975 GOTO 3780                           'EXIT to KB scan
3980 IF ASC(IK$)>90 THEN IK$=CHR$(ASC(IK$)-32):GOTO 3950 'Conv to lower case
3982 SOUND 37,1:GOTO 3780                'BLAT & rtn to KB scan
3985 '
4000 REM ----------------------
4010 REM CURSOR LEFT -- LEFT ARROW KEY
4020 '
4030 PO=PO-2:IF PO<PZ THEN PO=IL:GOSUB 4780 'Move back 1 and chk position
4040 CC=PEEK(PO):POKE PO,CR              'Flash cursor
4050 SOUND 2000,1:POKE PO,CC             'Waste time & restore char
4060 GOTO 3780                           'EXIT to key scan
4070 '
4080 REM ----------------------
4090 REM CURSOR RIGHT -- RIGHT ARROW KEY
4100 '
4110 PO=PO+2:IF PO>IL THEN PO=PZ:GOSUB 4780 'Move fwd 1 & chk position
4120 CC=PEEK(PO):POKE PO,CR              'Flash cursor
4130 SOUND 2000,1:POKE PO,CC             'Waste time & restore character
4140 GOTO 3780                           'EXIT to KB scan
4150 '
4160 REM ----------------------
4170 REM CURSOR TO BEGINNING OF LINE -- HOME KEY
4180 '
4190 PO = PZ                             'Set position to beg of line
4200 GOTO 3780                           'EXIT to KB scan
4210 '
4220 REM ----------------------
4230 REM CURSOR TO END OF LINE -- END KEY
4240 '
4250 PO = IL                             'Set positon to end of line
4260 GOTO 3780                           'EXIT to KB scan
4270 '
4280 REM ----------------------
4290 REM BACKSPACE AND ERASE -- BACKSPACE KEY
4300 '
4310 PO=PO-2:IF PO<PZ THEN PO=IL:GOSUB 4780 'Back 1 & chk position
4320 CC=32:POKE PO,CR                    'Set cur char to space & flash cursor
4330 IF F1 THEN 4400                     'Insert is on-go to delete char
4340 SOUND 1000,1:POKE PO,CC             'BEEP & print a space
4350 GOTO 3780                           'EXIT to KB scan
4360 '
```

```
4370 REM ----------------------
4380 REM DELETE CHARACTER — DELETE KEY
4390 '
4400 FOR X = PO TO IL STEP 2             'Set limits of move
4410  POKE X,PEEK(X+2)                   'Move entire line left 1 char
4420 NEXT X                              'Loop until done
4430 POKE IL,32                          'Put space in position 24
4440 SOUND 1000,1                        'BEEP
4450 GOTO 3780                           'EXIT to KB scan
4460 '
4470 REM ----------------------
4480 REM ERASE FROM CURSOR POSITION TO EOL — ^E
4490 '
4500 FOR X=PO TO IL STEP 2               'Set loop to length of line
4510  POKE X, 32                         'Fill with spaces
4520 NEXT X                              'Loop until done
4530 F2 = 1                              'Set flag: data is altered!
4540 GOTO 3780                           'EXIT to KB scan
4550 '
4560 REM ----------------------
4570 REM INSERT CHARACTER — INSERT MODE ON
4580 '
4590 FOR X = IL TO PO STEP -2            'Set limits of move
4600         POKE X,PEEK(X-2)            'Move line right 1 character
4610 NEXT X                              'Loop until done
4620 POKE PO,ASC(IK$)                    'Put a pace under the cursor
4630 PO=PO+2:IF PO>IL THEN PO=PZ:GOSUB 4780 'Increment pos'n & chk for EOLine
4640 F2 = 1                              'Set flag: data is altered!
4660 GOTO 3780                           'EXIT to KB scan
4670 '
4680 REM ----------------------
4690 REM TOGGLE INSERT MODE ON/OFF — INSERT KEY
4700 '
4710 IF F1 THEN F1=0:CR=220 ELSE F1=1:CR=219 'Toggle F1 (insert flag)
4720 LOCATE 18,1:PRINT MG$(F1);          'Pint input status mode (type/insert)
4730 GOTO 3780                           'EXIT to KB scan
4740 '
4750 REM ----------------------
4760 REM INSERT OFF — SUBROUTINE
4770 '
4780 F1 = 0:CR = 220                     'Turn off insert mode
4790 LOCATE 18,1:PRINT MG$(F1);          'Prnt input status mode (type/insert)
4800 RETURN                              'Return to caller
4810 '
4820 REM ----------------------
4830 REM SCROLL UP — UP ARROW KEY PRESSED
4840 '
4850 IF EN=1 THEN 4920                   'Special case: EXIT if file is 1 line
4860 IF F2 >0 THEN GOSUB 5090            'Data altrd-insert cur line into file
4870 IF F1 >0 THEN GOSUB 4780            'If insert on, turn it off
4880 IF EN<EF THEN CN=EN                 'Set temporary end of list variable
4890 LN=LN+1:IF LN>CN THEN LN=1          'Increment current line#
4900 GOSUB 5710                          'Go display data in text area
4910 PO=PZ                               'Set cursor to beg of line
4920 GOTO 3780                           'EXIT to KB scan
4930 '
```

```
4940 REM ----------------------
4950 REM SCROLL DOWN -- DOWN ARROW/CR KEY
4960 '
4970 IF EN=1 THEN 5040                  'Spec case: Exit if file has 1 line
4980 IF F2 >0 THEN GOSUB 5090           'Data altered: insert into file
4990 IF F1 >0 THEN GOSUB 4780           'If insert on, turn it off
5000 IF EN<EF THEN CN=EN                'Set temporary end of list
5010 LN=LN-1:IF LN<1 THEN LN=CN         'Decrement current line#
5020 GOSUB 5710                         'Go display data in text area
5030 PO=PZ                              'Set cursor to beg of line
5040 GOTO 3780                          'EXIT to KB scan
5050 '
5060 REM ----------------------
5070 REM CURRENT LINE (CL$) INTO THE ARRAY
5080 '
5090 DA$(LN)=""                         'Clr string of current contents
5100 FOR X = 0 TO 23                    'Set loop to length of pick-off
5110 DA$(LN)=DA$(LN)+CHR$(SCREEN(18,29+X)) 'Put screen contents into file
5120 NEXT X                             'Loop until done
5130 IF F2=1 THEN F3=1                  'Set flag: FILE altered!
5140 F2 = 0                             'Reset data changed flag
5150 PO=PZ:RETURN                       'Cursor beg of line & EXIT to KB scan
5160 '
5170 REM ----------------------
5180 REM PAGE UP -- PgUp KEY
5190 '
5200 IF EN=1 THEN 5260                  'Spec case: EXIT if 1 line file
5210 IF EN <12 THEN 4850                'Can't page-up do up arrow instead
5220 IF F2 >0 THEN GOSUB 5090           'Data altered-insert into array
5230 IF F1 >0 THEN GOSUB 4780           'If insert on, then turn off
5240 LN=LN+11:IF LN>EN THEN LN=LN-EN    'Increment line# by 11
5250 GOSUB 5710                         'Gp print the data in text area
5255 PO=PZ                              'Reset cursor position
5260 GOTO 3780                          'EXIT to KB scan
5270 '
5280 REM ----------------------
5290 REM PAGE DOWN -- PgDn KEY
5300 '
5310 IF EN=1 THEN  5370                 'Spec case: EXIT if 1 line file
5320 IF EN <12 THEN  4970               'Can't PgDn - use dwn arrow instead
5330 IF F2 >0 THEN GOSUB 5090           'Data altered-insert into file
5340 IF F1 >0 THEN GOSUB 4780           'If insert on, turn it off
5350 LN=LN-11:IF LN<1 THEN LN=LN+EN     'Decrement line# by 11
5360 GOSUB 5710                         'Go print data in text area
5365 PO=PZ                              'Reset cursor position
5370 GOTO 3780                          'EXIT to KB scan
5380 '
5390 REM ----------------------
5400 REM JUMP NEXT LABEL  -- ^J
5410 '
5420 IF EN=1 THEN 5500                  'Spec case: EXIT if 1 line file
5430 IF F2 >0 THEN GOSUB 5090           'Data altered-insert into array
5440 IF F1 >0 THEN GOSUB 4780           'If insert on, turn it off
5450 GOSUB 13760                        'Search from LN to EN for label
5460 IF LA>0 THEN  5480                 'Label found [ LA >0 ], EXIT
5470 GOSUB 13840                        'Search from 1 to LN
5480 IF LA>0 THEN LN=LA ELSE GOTO 5500  'Set LN to found label line#
5490 GOSUB 5710                         'Display data in text area
5500 GOTO 3780                          'EXIT to KB scan
5510 '
```

```
5520 REM ------------------------
5530 REM RESTORE CHANGES TO DATA -- ^R
5540 '
5550 LOCATE 18,29                          'Set print positon
5560 PRINT DA$(LN);                        'Print data in active line
5570 F2 = 0                                'Reset changes to data flag
5580 GOTO 3780                             'EXIT to KB scan
5590 '
5600 REM ------------------------
5610 REM EXIT TO MAIN MENU -- <Esc> KEY
5620 '
5630 IF F2 >0 THEN GOSUB 5090              'Data altered - insert into file
5640 IF F1 >0 THEN GOSUB 4780              'If insert on, turn it off
5650 LOCATE 18,1:PRINT "
                                               " 'Locate & clear message area
5660 GOTO 2010                             'EXIT to menu routine
5670 '
5680 REM ------------------------
5690 REM PRINT TEXT AREA
5700 '
5710 IF EN>11 THEN 5920                    'Determine file size & jump
5720 ' - - - - -PRINT TOP OF SMALL ARRAY
5730 CL=LN-5:IF CL<0 THEN CL=LN-6          'Set cur line# & test for <0
5740 FOR X=1 TO 5                          'Sed loop to print top 5 lines
5750  LOCATE 12+X,25                       'Set print position
5760  IF CL<=0 THEN PRINT BL$:GOTO 5790    'If no active lines, print blanks
5770  PRINT LN$(CL)+" "+CHR$(179);         'Print line# & vertical bar of box
5780  PRINT DA$(CL)+CHR$(179)+" "+DT$(DA(CL))+DT$(DT(CL)) 'Finish prntg line
5790  CL = CL+1:IF CL=0 THEN CL=1          'Increment cur line# & test for 0
5800 NEXT X                                'Loop until done
5810 '- - - - - -PRINT BOTTOM OF SMALL ARRAY
5820 CL=LN                                 'Set cur line# to active line
5830 FOR X=6 TO 11                         'Set loop to print bottom 6 lines
5840  LOCATE 12+X,25                       'Set print position
5850  IF CL>EN THEN PRINT BL$:GOTO 5880    'If no active lines, print blanks
5860  PRINT LN$(CL)+" "+CHR$(179);         'Print line# & vert bar of box
5870  PRINT DA$(CL)+CHR$(179)+" "+DT$(DA(CL))+DT$(DT(CL)) 'Finish prntg line
5880  CL=CL+1                              'Increment current line#
5890 NEXT X                                'Loop until done
5900 GOTO 6020                             'Jump to routine EXIT
5910 ' - - - - -PRINT LARGE ARRAY
5920 CL=LN-5                               'Cur line# to top of text area
5930 IF EN=>EF THEN CN=EF ELSE CN=EN+1     'Set cur end of list = plus 1
5940 IF CL<=0 THEN CL=EN+CL                'If cur line <0, reset to 1
5950 IF CL=0 THEN CL=1                     'Test cur line# for 0
5960 FOR X=1 TO 11                         'Set loop to print 11 lines
5970  LOCATE 12+X,25                       'Set print positon
5980  PRINT LN$(CL)+" "+CHR$(179);         'Print line numbers & vert bar on box
5990  PRINT DA$(CL)+CHR$(179)+" "+DT$(DA(CL))+DT$(DT(CL)) 'Finish prntg line
6000  CL = CL+1:IF CL>EN THEN CL=1         'Inc cur line# & test for EOFile
6010 NEXT X                                'Loop until done
6020 GOSUB 11330                           'Go print file status
6030 IL=PZ+46:IF DA(LN)<5 AND DT(LN)>0 THEN IL=PZ+22 'Sen len of active line
6040 PO=PZ:RETURN '                        'Set cursor to beg of line & RETURN
6050 '
```

```
6060 REM ---------------------
6070 REM DELETE LABEL/DATA LINE(S) — ^D
6080 '
6090 IF DA(LN)=5 THEN  6100 ELSE 6280     'Get DT of LN & jump on result
6100 IK=DT(LN)                            'Preserve data type of current line
6110 IF DA(LN+1)=5 THEN GOSUB 6570:GOTO 6280 'If nxt line lbl: calc fre lbls &
6111                                        'jump to process the insert
6120 IF LN=EN THEN GOSUB 6570:GOTO 6280   'Jump if cur line is EOFile
6130 MG=12:GOSUB 13430                    'Print "will lose data" message
6140 '- - - - - - - - - - - - -
6150 IK$=INKEY$:IF IK$="" THEN 6150       'Wait for user input (Y or N)
6160 ON INSTR("YynN",IK$) GOTO 6190,6190,6400,6400 'Jump on valid input
6170 GOTO 6150                            'Rtn to KB scan if invalid input
6180 '- - - - - - - - - - - - -
6190 GOSUB 13630                          'Answer was YES-clr warning msg
6200 GOSUB 6570                           'Calc new number of free labels
6210 '- - - - - - - - - - - - -
6220 GOSUB 13760                          'Find next label in file, if any
6230 IF LA>0 THEN OS=LA-LN:GOTO 6300      'Jump if label found
6240 EN=LN-1:GOSUB 6510                   'LA=0, go clear the list
6250 IF DA(1)<5 THEN GOSUB 6620           'Fix spec case cond #1
6260 GOTO 6340                            'Jump to next section of code
6270 '- - - - - - - - - - - - -
6280 OS=1                                 'Set off-set =1
6290 '- - - - - - - - - - - - -
6300 MG=13:GOSUB 13430                    'Print "deleting" message
6310 GOSUB 6440                           'Move data up in list
6320 EN=EN-OS:GOSUB 6510                  'Set EOList & fill unused portion
6330 '- - - - - - - - - - - - -
6340 IF EN<2 THEN 6350 ELSE 6360          'Jump on spec case #2
6350 IF DA(1)<5 THEN GOSUB 6620           'Fix spec case conditon #1
6360 IF LN>EN THEN LN=EN                  'Make sure there's no bogus line
6370 IF LN<1 THEN LN=1                    'Make sure there's no bogus line
6380 IF EN=0 THEN EN=1:GOSUB 6620         'Fix spec case condition #1
6390 GOSUB 5710                           'Print the data in the text area
6400 GOSUB 13630                          'Clear message line
6410 F3=1:PO=PZ:GOTO 3770                 'Set flag: data altered! Set cursor
6411                                      'to beg of line & EXIT to KB scan
6420 '
6430 '- - - - -MOVE DATA UP - SBR
6431 '
6440 FOR X=LN TO EN-OS                    'Set loop
6450  DA$(X)=DA$(X+OS)                    'Move data up by off-set amount
6460  DA(X)=DA(X+OS):DT(X)=DT(X+OS)       'Move data attrib & type
6470 NEXT X                               'Loop until done
6480 RETURN                               'Return to caller
6490 '
6500 '- - - - -CLEAR LIST - SBR
6501 '
6510 FOR X=EN+1 TO EF                     'Set loop
6520   DA$(X)=DD$(0):DT(X)=0:DA(X)=4      'Load unused lines w/memo data
6530 NEXT X                               'Loop until done
6540 RETURN                               'Return to caller
6550 '
```

```
6560 '- - - - -CALC FREE LABELS - SBR
6561 '
6570 LA(IK)=LA(IK)-1                  'Decrement label count by 1
6580 IF LA(IK)<0 THEN LA(IK)=0        'Make sure count doesn't go wrong
6590 RETURN                           'Return to caller
6600 '
6610 '- - - - -SPECIAL CASE PROCESSING - SBR
6611 '
6620 IF EN>2 THEN 6650                'EXIT if no spec case exists
6630 LA(0)=1                          'Set Memo labels to 1
6640 DA$(1)=DL$(0):DA(1)=5:DT(1)=0    'Set line #1 to Memo label
6650 RETURN                           'Return to caller
6660 '
6670 REM ----------------------
6680 REM INSERT LABEL/DATA LINE -- ^I
6690 '
6700 IF EN+1>EF THEN MG=7:GOSUB 13230:GOTO 3770 'Watch is full-prnt msg & EXIT
6710 MU=2:GOSUB 3340                  'Print Insert Menu (I-Menu)
6720 SA=31:MG=11:GOSUB 13430          'Print prompt
6730 IF EN<EF THEN GOSUB 3540 ELSE GOTO 7240 'Get KB input from user or EXIT
6740 IF IK$=CHR$(27) THEN  3770       'EXIT to edit/create @ clr menu
6750 '
6760 ' - - - - -INPUT PROCESSING
6761 '
6770 IK=VAL(IK$)-2:IF IK<1 THEN IK=0  'Set data type requested into IK
6780 IF IK$="1" THEN RQ=5:GOTO 6820   'Set requested insert to memo LABEL
6790 IF LA(IK)>0 THEN RQ=4:GOTO 6831  'If SA,WA,WT lbls exist, set to data
6792 RQ=5                             'If falls thru, insert a label
6800 '
6810 '- - - - -MAINLINE INSERT PROCESSING
6811 '
6820 GOSUB 6900                       'Chk for directory space-any left?
6830 IF Q1=0 THEN MG=8:GOSUB 13230:GOTO 6720 'No dir space-prnt msg & go menu
6831 IF DT(LN)=IK THEN LN=LN+1:GOTO 7080 'DT Match, go insert line
6832 IF LA(IK)=0 AND IK>0 THEN LN=1:GOTO 7080 'Ins SA,WA,WT lbls @ top of file
6833 IF LA(IK)=0 AND IK=0 THEN LN=EN+1:GOTO 7080 'Ins ME lbl @ bot of file
6834 IF LA(0)>0 AND IK=0 AND RQ=4 THEN GOSUB 6990:GOTO 7080 'Ins ME data undr
6835                                                          'a memo label
6850 IF LA(IK)>0 THEN GOSUB 6990:GOTO 7080 'Label exists-go find & insert
6860 IF IK=0 AND RQ=5 THEN LN=EN+1:GOTO 7080 'Put ME lbl @ end of file
6861 LN=1                             'Set LN=1 if not set by now!
6870 GOTO 7080                        'Jump to insert code
6880 '
6890 '- - - - -CALC DIR SPACE - SBR
6891 '
6900 Q1=0                             'Set "Question 1" =0  [No]
6910 FOR X=0 TO 3                     'Set loop to add four label types
6920  Q1=Q1+LA(X)                     'Add up the labels used
6930 NEXT X                           'Loop until done
6950 IF Q1>11 THEN Q1=0 ELSE Q1=1     'Set Answer: YES=1, NO=0
6960 RETURN                           'Return to caller
6970 '
```

```
6980 '- - - - -FIND LABEL - SBR
6981 '
6990 X=LN+1:LA=0                              'Set beginning search param
7000 IF DA(X)=5 AND DT(X)=IK THEN LA=X:GOTO 7050 'Search-EXIT if found
7010 X=X+1                                    'Increment counter
7020 IF X<=EN THEN  7000                      'Loop to end of list
7030 X=1                                      'Not found-start @ beg of list
7040 IF DA(X)=5 AND DT(X)=IK THEN LA=X:GOTO 7050 'Search-EXIT if found
7041 X=X+1                                    'Incerment counter
7042 IF X<LN+1 THEN  7040                     'Loop until active line# reached
7050 LN=LA+1                                  'Load cur line# w/found label address
7055 RETURN                                   'Return to caller
7060 '
7070 '- - - - -INSERT LABEL/DATA - SBR
7071 '
7080 MG=6:GOSUB 13430                         'Print insert message
7090 FOR X=EN+1 TO LN+1 STEP -1               'Set range of lines to move
7100  LSET DA$(X)=DA$(X-1)                    'Move the data
7110  DA(X)=DA(X-1)                           'Move data attrib down 1 line
7120  DT(X)=DT(X-1)                           'Move data type down 1 line
7130 NEXT X                                   'Loop until done
7140 IF RQ=5 THEN DA$(LN)=DL$(IK)             'Default label into new line
7150 IF RQ=4 THEN DA$(LN)=DD$(IK)             'Default data into new line
7160 DA(LN)=RQ:DT(LN)=IK                      'Set data attrib & type
7180 EN=EN+1                                  'Increment end of list +1
7190 IF DA(LN)=5 THEN LA(IK)=LA(IK)+1         'Increment label count
7200 GOSUB 5710                               'Print text area
7210 MU=0:F3=1                                'Set M-Menu, set data altered flag F3
7211 PO=PZ                                    'Set cursor pos to beg of line
7220 GOTO 6700                                'Re-enter insert routine
7229 '
7230 ' - - - - WATCH FULL  - - -
7231 '
7240 MG=7:GOSUB 13230:GOTO 7210               'Watch full-exit to menu input
7250 '
7300 REM ----------------------
7301 REM FIND A CITY IN WORLD TIME LIST -- ^F
7302 '
7305 IF DA(LN)=5 OR DT(LN)<3 THEN 7360        'EXIT if wrong DA or DT
7310 MG=28:GOSUB 13430                        'Prnt input message
7315 IP$="":LO=47:GOSUB 13930:GOSUB 14505'Prnt msg & wait for input
7320 IF IP$="" THEN 7360                      'EXIT if null input
7325 IP$=LEFT$(IP$,12)                        'Make sure search$ is <13 chars
7330 WT=1                                     'Set World Time counter to 1
7335 MG=31:GOSUB 13430                        'Print "searching" message
7340 GET 3,WT:WT=WT+1:IF WT>(LOF(3)/20)+1 THEN 7355 'Get WT rec from disk
7345 IF INSTR(WT$,IP$) THEN GOSUB 13630:GOTO 7526 'Srch $; clr msg; print $
7350 GOTO 7340                                'Loop if search $ not found
7355 MG=30:GOSUB 13230                        'Print "not found" msg
7360 GOSUB 13630:GOTO 3780                    'Clear msg area; EXIT to KB scan
7361 '
```

```
7400 REM ----------------------
7401 REM SEARCH FOR A STRING IN DA$( LIST -- ^S
7402 '
7405 IF LN=EN THEN 7460                     'EXIT if at end of file
7408 GOSUB 5090                             'Pick-off cur data & insert into file
7410 MG=29:GOSUB 13430                      'Print input message
7415 IP$="":LO=47:GOSUB 13930:GOSUB 14505'Null IP$, SOUND inpt, wait for input
7420 IF IP$="" THEN 7455                    'EXIT if null input
7425 IP$=LEFT$(IP$,24)                      'Make sure search$ <25 characters
7430 MG=31:GOSUB 13430                      'Print "searching" msg
7435 X=LN+1                                 'Get starting line number
7440 IF INSTR(DA$(X),IP$) THEN LN=X:GOTO 7455 'Target found! EXIT
7445 X=X+1:IF X>EN THEN 7450 ELSE 7440      'Loop to end of file
7450 MG=30:GOSUB 13230                      'Print "not found" msg
7455 GOSUB 13630                            'Clr message
7456 IF X=LN THEN GOSUB 5710                'Print string if found
7460 GOTO 3780                              'EXIT to KB scan
7480 REM ----------------------
7481 REM JUMP TO BEG OF DA$( LIST -- ^B
7482 '
7485 GOSUB 5090:PO=PZ                       'Active line into list, cursor to beg
7486 LN=1:GOSUB 5710:GOTO 3780              'Line# to 1, prt txt, EXIT to KB scan
7489 '
7490 REM ----------------------
7491 REM JUMP TO END OF DA$( LIST -- ^N
7492 '
7495 GOSUB 5090:PO=PZ                       'Active line into list, cursor to beg
7496 LN=EN:GOSUB 5710:GOTO 3780             'Lin# to EN, prt txt, EXIT to KB scan
7499 '
7500 REM ----------------------
7501 REM ROLL WORLD TIME FROM LIST - F9/F10
7502 '
7503 '- - - - - ROLL BACK - - - -
7505 IF DA(LN)=5 OR DT(LN)<3 THEN 3780      'EXIT if wrong DA or DT
7506 WT=WT-1:IF WT<1 THEN WT=LOF(3)/20      'Decrement world time record counter
7507 GET 3,WT:GOTO 7526                     'GET next World time record & jmp
7508 '- - - - - ROLL FWD- - - -
7510 IF DA(LN)=5 OR DT(LN)<3 THEN 3780      'EXIT if wrong DA or DT
7511 WT=WT+1:IF WT>LOF(3)/20 THEN WT=1      'Increment word time record counter
7512 GET 3,WT                               'GET nxt World Time rec & fall thru
7515 '- - - - - - - - - - - - - -
7526 LOCATE 18,29:PRINT WT$;                'Position & print WT record
7551 F2=1                                   'Set flag: data altered!
7560 GOTO 3780                              'EXIT to KB scan
7561 '
7570 REM ----------------------
7571 REM TAB RIGHT/LEFT 12 CHARACTERS -- ^T
7572 '
7575 IF DA(LN)=5 THEN  7580                 'Jump if current line is a label
7576 IF DT(LN)>0 THEN  3780                 'EXIT if wrong data type [1,2,3]
7580 IF PO<PZ+24 THEN PO=PZ+24:GOTO 7590 'Set position to watch's line# 2
7585 IF PO=>PZ+24 THEN PO=PZ                'Set position to watch's line#1
7590 GOTO 3780                              'EXIT to KB scan
7591 '
```

```
7600 REM ***********************************************************************
7601 REM WRITE TO WATCH
7602 REM ***********************************************************************
7603 '
7605 Q1=LN                                    'Preserve current line number
7610 MG=21:GOSUB 13430                        'Print "Writing..." messge
7611 LO=54                                    'Set print pos'n for progress report
7615 OPEN "COM1:2400,N,8,2,RS,CS0,DS0,CD0" AS #1 'Set com to RC1000 parameters
7619 '
7620 '-----------BUILD WATCH DIRECTORY
7625 '
7630 DR$=CHR$(0)+"L"                          'Set directory header
7635 FOR X=1 TO EN                            'Set loop to run thru active list
7640  IF DA(X)=5 THEN GOSUB 8705              'Found a label-go process it
7645 NEXT X                                   'Loop until done
7648 'Y=((X-1)*25)+16384:GOSUB 8715            'Calc addr of end of active file
7650 DR$=DR$+"@"+CHR$(0)                      'Calc EOF address
7655 DR$=LEFT$(DR$+STRING$(22,0),26)          'Pad unused directory
7660 '- - - - - - - - - - - - -
7665 PRINT #1, DR$;                           'Write directory to watch
7670 '
7675 '-----------MAINLINE WRITE TO WATCH
7680 '
7685 X=1
7690 LOCATE 9,LO:PRINT X;                     'Show load progress
7692 ON DT(X)+1 GOTO 7740,7805,8005,8205      'Jmp to process data type
7695 X=X+1:IF X<=EN THEN  7690                'Inc X counter til EOList
7696 '- - - - - - - - - - - - -
7697 FOR X=EN+1 TO 81                         'Set loop to fill remainder of watch
7698  LOCATE 9,LO:PRINT X;                    'Print progress report
7699  PRINT#1, "@"+STRING$(24,32);            'Fill watch with dummy data
7700 NEXT X                                   'Loop until done
7705 '- - - - - - - - - - - - -
7710 CLOSE 1                                  'Close communications file
7715 LN=Q1:LO=0                               'Restore cur line # & reset prnt posn
7720 GOTO 2010                                'EXIT to main menu
7725 '
7730 '-----------WRITE MEMO DATA AND LABELS --DT=0/DA=5
7735 '
7740 IF DA(X)=5 THEN PR$="L" ELSE PR$="d"'Set data attribute to label or data
7745 PR$=PR$+DA$(X)                           'Bld "write" string for label or memo
7750 PRINT #1, PR$;                           'Write to watch
7755 GOTO 7695                                'EXIT to main line write
7760 '
7800 '-----------WRITE SCHEDULE ALARM --DT=1
7801 '
7805 IF DA(X)=5 THEN  7740                    'If label use memo's write
7810 AP=0                                     'Make sure AP is zero
7815 GOSUB 8505                               'Set up for sort & find label
```

```
7820 '- - - - - - - - - - - - - -
7825 FOR J=0 TO LL                       'Set loop to len of sort list
7826 LOCATE 9,LO+3:PRINT"-";             'Show activity
7827 AP=0                                'Set AP to zero
7835  MO$=MID$(DA$(BL+J),13,2)           'Pick off month
7840  DY$=MID$(DA$(BL+J),16,2)           'Pick off day
7845  HR$=MID$(DA$(BL+J),20,2)           'Pick off hours
7847  AP$=MID$(DA$(BL+J),19,1):IF AP$="P" AND VAL(HR$) <12 THEN AP=12 'Pick
                                          AM/PM & set AP
7848  If AP$="A" AND VAL(HR$)=12 THEN HR$="00" 'Set midnight to zero
7850  MI$=MID$(DA$(BL+J),23,2)           'Pick off minutes
7855  HR$=STR$(VAL(HR$)+AP)              'Put hours into 24 hour time
7856  HR$="0"+RIGHT$(HR$,LEN(HR$)-1)     'Pad left w/ASCII 0 if HR <9
7857  HR$=RIGHT$(HR$,2)                  'Cut down to size
7860 '
7865  SD(J)=VAL(MO$+DY$+HR$+MI$)         'Load array elment w/sort data
7870  SN(J)=VAL(LN$(BL+J))               'Load array element w/line #s
7871 LOCATE 9,LO+3:PRINT" ";             'Show activity
7875 NEXT J                              'Loop until done
7880 '- - - - - - - - - - - - - -
7885 GOSUB 8605                          'Go sort SD( & SN(
7890 '- - - - - - - - - - - - - -
7895 FOR J=0 TO LL-1                     'Set lp to write SALARM to wtch
7900  PR$="d"+DA$(SN(J))                 'Build "write" string
7905  PRINT #1,PR$;                      'Send it to the watch
7910  NEXT J                             'Loop until done
7915 X=EL                                'Reset X new value
7920 GOTO 7695                           'EXIT to mainline write routine
7925 '
8000 '----------WRITE WEEKLY ALARM --DT=2
8001 '
8005 IF DA(X)=5 THEN  7740               'If label use memo's write
8010 GOSUB 8505                          'Set up for sort & find label
8015 '- - - - - - - - - - - - - -
8020 FOR J=0 TO LL-1                     'Set loop to list length
8022  AP=0                               'Set AP to zero
8023 LOCATE 9,LO+3:PRINT"-";             'Show activity
8025  AP$=MID$(DA$(BL+J),19,1)           'Pk-off A-PM (PM=+12 hrs)
8030  DY$=MID$(DA$(BL+J),13,1)           'Pick-off day
8035  HR$=MID$(DA$(BL+J),20,2)           'Pick-off hour
8040  MI$=MID$(DA$(BL+J),23,2)           'Pick-off minute
8042  IF AP$="P" AND VAL(HR$)<12 THEN AP=12 'Conv to 24 hour time for sort
8045  HR$=STR$(VAL(HR$)+AP)              'Hours into 24 hour time & pad left
8046  HR$="0"+RIGHT$(HR$,LEN(HR$)-1)     'Pad left w/ASCII 0 if HR <9
8047  HR$=RIGHT$(HR$,2)                  'Cut down to size
8050 '
8055  SD(J)=VAL(DY$+HR$+MI$)             'Load array elment w/sort data
8060  SN(J)=VAL(LN$(BL+J))               'Load array element w/line #s
8061 LOCATE 9,LO+3:PRINT" ";             'Show activity
8065 NEXT J                              'Loop until done
8070 '- - - - - - - - - - - - - -
8075 GOSUB 8605                          'Go sort SD( & SN(
8080 '- - - - - - - - - - - - - -
8085 FOR J=0 TO LL-1                     'Set loop to write to watch
8090  PR$="d"+DA$(SN(J))                 'Build "write" string
8095  PRINT #1,PR$;                      'Send it to the watch
8100 NEXT J                              'Loop until done
8105 X=EL                                'Reset X new value
8110 GOTO 7695                           'EXIT to mainline write routine
8115 '
```

```
8200 '-----------WRITE WORLD TIME --DT=3
8201 '
8205 IF DA(X)=5 THEN  7740                   'If label use memo's write
8210 '- - - - - - - - - - - - -
8215 HR$=MID$(DA$(X),14,2)                   'Pick-off hour difference
8220 MI$=MID$(DA$(X),17,2)                   'Pick-off minute difference
8225 HR=VAL(HR$)                             'Conv hour string to value
8228 AP$="0"                                 'Set default cond for < 12 hours
8229 HR=HR-TZ:IF HR<0 THEN HR=HR+24          '<NEW calc TZ difference
8230 IF HR>11 THEN AP$="1":HR=HR-12          'Put HR$ into RC format
8231 HR$=STR$(HR)                            'Put HR$ into RC format
8232 HR$="0"+RIGHT$(HR$,LEN(HR$)-1)          'Pad left w/ASCII 0 if HR <9
8233 HR$=RIGHT$(HR$,2)                       'Cut down to size
8235 '- - - - - - - - - - - - -
8240 PR$="d"+LEFT$(DA$(X),12)               'Build 1st half of "write" str
8245 PR$=PR$+AP$+HR$+MI$+"         "         'Build 2nd half of "write" str
8250 PRINT #1,PR$;                           'Send it to the watch
8255 GOTO 7695                               'EXIT to mainline write routine
8260 '
8500 '-----------UTILITY-Determins the list length
8501 '
8505 BL=X:LN=X                               'Beg of List & current Line Number=X
8510 GOSUB 13760                             'Go find the end of this list
8515 IF LA>0 THEN EL=LA-1                    'LA is nxt label addr
8520 IF LA=0 THEN EL=EN                      'No label found-set End srtlst=to EN
8525 LL=(EL-BL)+1                            'List Length = End List - Beg List
8530 RETURN                                  'Return to caller
8535 '
8601 '-----------SHELL SORT
8602 '
8605 G%=LL:N%=LL                             'Load G% and N% with the list length
8610 '- - - - - - - - - - - - -
8615 WHILE G%>1:G%=G%/2                      'Set WHILE condition
8620  FOR F%=1 TO 1                          'Set loop
8625   FOR C%=0 TO N%-G%-1                   'Set trip counter
8630    WHILE SD(C%)> SD(G%+C%)              'Set WHILE cond to compare elements
8631     LOCATE 9,LO+3:PRINT "*"             'Print progress report
8635     SWAP SD(C%),SD(G%+C%)               'Swap data in sort array
8640     SWAP SN(C%),SN(G%+C%)               'Swap line numbers in sort array
8645     F%=0                                'Set outer loop counter to 0
8646     LOCATE 9,LO+3:PRINT " "             'Print progress report
8650    WEND                                 'End inner WHILE-WEND loop
8655   NEXT C%                               'End inner FOR loop
8660  NEXT F%                                'End outter FOR loop
8665 WEND                                    'End outter WHILE-WEND loop-srt done!
8666 LOCATE 9,LO+3:PRINT " "                 'Print progress report
8670 RETURN                                  'Return to caller
8675 '
```

```
8700 '----------UTILITY -- WATCH DIRECTORY
8701 '
8705 Y=(X-1)*25                         'Calculate address from line number
8710 Y=Y+(4096*DT(X))                   'Set data type bit in High nybble
8715 DR$=DR$+RIGHT$(MKI$(Y),1)+LEFT$(MKI$(Y),1) 'Put address into dir string
8720 RETURN                             'Return to caller
8725 '
8750 REM ----------------------
8751 REM CUT -- LOAD C&P BUFFER
8752 '
8755 CT$="":CT=DT(LN)                   'Clr cut buffer & get DT
8758 FOR X=0 TO 24                      'Set loop to read screen
8760 CT$=CT$+CHR$(SCREEN(18,29+X))      'Load C&P buff from screen
8762 NEXT X                             'Loop until done
8764 COLOR 0,7:LOCATE 7,54:PRINT"*";    'Show buffer loaded
8766 COLOR 7,0:GOTO 3780                'Reset video & EXIT
8780 REM ----------------------
8781 REM PASTE -- PRINT CUT BUFFER
8782 '
8785 IF DT(LN)=CT OR DT(LN)=0 OR DA(LN)=5 THEN GOTO 8788 'Chk target attribs
8786 LOCATE 18,29:PRINT LEFT$(CT$,12);:GOTO 8790 'Print C&P buffer
8788 LOCATE 18,29:PRINT CT$;            'Print C&P buffer
8790 F2=1:GOTO 3780                     'Set flage: data altered
8999 '
```

```
9000 REM ******************************************************************
9010 REM FILE I/O  -- DISK/CASSETTE
9020 REM ******************************************************************
9030 '
9040 REM ----------------------
9050 REM READ DISK FILE
9060 '
9070 IF FI$="" THEN MG=27:GOSUB 13230:GOTO 9220 'Print err msg if no file name
9071 IF F3>0 THEN MG=10:GOSUB 13430 ELSE 9075 '<NEW chk for altered file
9072 IK$=INKEY$:IF IK$="" THEN 9072       'Scan KB for user's answer to prompt
9073 ON INSTR(YN$,IK$) GOTO 9075,9075,9220,9220,9220 '<NEW jump on Y/N/Esc
9074 GOTO 9072                            'Invalid input - rtn to KB scan
9075 ON ERROR GOTO 9550                   'Set error trap; EXIT thru "WRITE"
9080 OPEN"I",1,FI$                        'Open data file
9090 MG=16:GOSUB 13430                    'Print message
9100 INPUT #1, DA$(0)                     'Read file record 0 (active lines)
9110 EN=VAL(DA$(0))                       'Set End of List
9140 LA(0)=0:LA(1)=0:LA(2)=0:LA(3)=0      'Set label counts to 0
9150 FOR X = 1 TO EF                      'Set Read loop
9160  INPUT#1, DA$(X):                    'Read record into file array
9170  GOSUB 9930                          'Unpack data file
9180 NEXT X                               'Loop until done
9185 ON ERROR GOTO 9225                   'Set error trap if no TZ in file
9186 INPUT #1, PR$:TZ=VAL(PR$)            'Read TZ & convert to value
9190 ON ERROR GOTO 0                      'Reset error trap
9200 CLOSE 1                              'Close input file
9210 F3=0:LN=1:GOSUB 5710                 'Set file altered flg=0, line#=1
9211                                      'and display data in text area
9220 RETURN                               'Return to caller
9225 RESUME NEXT                          'Continue if no TZ in file
9230 '
9300 REM ----------------------
9310 REM WRITE DISK FILE -- SAVE
9320 '
9330 IF FI$="" THEN MG=27:GOSUB 13230:GOTO 9520 'No file name, prt msg & EXIT
9335 IF F2=1 THEN GOSUB 5090              'If data altered, insert into list
9340 ON ERROR GOTO 9590                   'SEt error trap
9350 OPEN"I",1,FI$:CLOSE 1                'Test file-OPEN for INPUT & CLOSE
9360 MG=26:GOSUB 13430                    'If file OPENed, it exists, prt msg
9370 IK$=INKEY$:IF IK$="" THEN 9370       'Wait for user response
9380 ON INSTR(YN$,IK$) GOTO 9410,9410,9490,9490,9490 'Jump on input
9390 GOTO 9370                            'Rtn to KB scan on invalid input
9400 ' - - - - - - - - - - - - -
9410 MG=15:GOSUB 13430                    'Print "saving..." message
9420 ON ERROR GOTO 9550                   'Set error trap
9430 OPEN"O",1,FI$                        'Open file for Output
9440 PRINT #1, STR$(EN)                   'Write the active # of records
9450 FOR X=1 TO EF                        'Set loop to write active records
9460  GOSUB 10030                         'Pact the data into PR$
9470  PRINT #1, PR$                       'Write PR$ to disk
9480 NEXT X                               'Loop until done
9485 PRINT #1, STR$(TZ)                   'Write cur time zone to end of file
9486 F3=0                                 'Set file not saved flag = 0
9490 CLOSE 1                              'CLOSE the file {EXIT point 1}
9500 ON ERROR GOTO 0                      'Reset error trap
9510 CLOSE 1                              'CLOSE the file {EXIT point 2}
9520 GOSUB 13630                          'Clear mesage
9530 RETURN                               'Return to Caller
```

```
9540 '- - - ERROR TRAP- - - - -
9550 RESUME 9560                         'Set next line to execute
9560 MG=17:GOSUB 13230                   'Print messge
9570 GOTO 9490                           'Re-enter WRITE routine
9580 '- - - ERROR TRAP- - - - -
9590 RESUME 9410                         'Re-enter WRITE routine
9600 '
9900 REM ----------------------
9910 REM UNPACK DATA FILE - SBR
9920 '
9930 DA(X)=4                             'Set default DT to "data"
9940 IF LEFT$(DA$(X),1)="L" THEN DA(X)=5 'If data attribute into variable
9950 DT(X)=VAL(RIGHT$(DA$(X),2))         'Read data type into variable
9960 IF DA(X)=5 THEN LA(DT(X))=LA(DT(X))+1 'If line is label, count it
9970 LN$(X)=RIGHT$(STR$(X),2)            'Load the line number into variable
9980 DA$(X)=MID$(DA$(X),3,24)            'Trim off excess fat-load data line
9990 RETURN                              'Return to caller
9999 '
10000 REM ----------------------
10010 REM PACK DATA FILE - SBR
10020 '
10030 IF DA(X)=5 THEN PR$="L " ELSE PR$="d " 'Set data attrib for WRITE
10070 PR$=PR$+DA$(X)+RIGHT$(STR$(DT(X)),2)   'Build WRITE string
10080 RETURN                                 'Return to caller
10090 '
10100 REM ----------------------
10110 REM CREATE "EMPTY" FILE
10120 '
10130 DA(1)=5:DT(1)=0:LN$(1)=" 1":DA$(1)=DL$(0) 'Create line #1
10140 FOR X=2 TO EF                      'Set loop
10150  DA(X)=4:DT(X)=0                   'Intialize data attribs & type
10160  LN$(X)=RIGHT$(STR$(X),2)          'Create line numbers
10170  DA$(X)=DD$(0)                     'Load default data strings
10180 NEXT X                             'Loop until done
10190 EN=1:LN=1:LA(0)=1:FI$=""           'Initialize len, lbls & active files
10200 RETURN                             'Return to caller
10210 '
```

```
10300 REM *******************************************************************
10301 REM UTILITY ROUTINES
10302 REM *******************************************************************
10303 '
10305 REM ------------------------
10306 REM DATE & TIME INPUT
10320 '
10330 ON ERROR GOTO 10490                   'Set error trap
10340 MG=18:GOSUB 13430:MG=20               'Print "enter time" message
10350 IP$="":LO=53:GOSUB 13930:GOSUB 14505 'Print prompt & wait for input
10360 IF IP$="" THEN  10400                 'If IP$ is null, EXIT
10370 TIME$=IP$                             'Load system time
10380 '- - - - - - - - - - - - - -
10390 ON ERROR GOTO 10490                   'Set error trap
10400 MG=19:GOSUB 13430:MG=20               'Print "enter date" message
10410 IP$="":LO=53:GOSUB 13930:GOSUB 14505 'Print prompt & wait for input
10420 IF IP$="" THEN 10460                  'If IP$ is null, EXIT
10440 DATE$=IP$                             'Load system date
10450 '- - - - - - - - - - - - - -
10460 ON ERROR GOTO 0                       'Reset error trap
10465 GOSUB 11330                           'Update status display
10470 GOTO 2950                             'EXIT to S-Menu input
10480 '- - - -ERROR TRAP - - - -
10490 RESUME 10500                          'Set RESUME to next line
10500 GOSUB 13230                           'Print error message
10510 IF MG=20 THEN 10330 ELSE 10390        'Return to offending input routine
10520 '
11030 '
11040 REM -----------------------
11050 REM CLEAR MENU SCREEN AREA
11060 '
11070 FOR X = 1 TO 9                        'Set loop size
11080  LOCATE X+1,2                         'Set print position
11090  PRINT SPC(78)                        'Print spaces
11100 NEXT X                                'Loop until done
11110 RETURN                                'Return to caller
11120 '
11300 REM ------------------------
11310 REM PRINT WATCH MEMORY STATUS
11320 '
11330 LOCATE 12,1                           'Set print position
11331 IF TZ=0 THEN TZ=4                     'Set time zone if not set
11340 PRINT MG$(3);80-EN                    'Print status, number of lines free
11350 LOCATE 13,1                           'Set print position
11360 PRINT MG$(4);12-(LA(0)+LA(1)+LA(2)+LA(3)) 'Print labels free
11361 LOCATE 14,1:PRINT MG$(2);TZ           'Position & print time zone
11390 LOCATE 22,1:PRINT MG$(5);             'Position & print msg
11400 LOCATE 23,1:PRINT STRING$(20,32);     'Position & erase line #23
11410 LOCATE 23,1:IF FI$="" THEN PRINT"--none--" ELSE PRINT FI$; 'Prnt filspec
11415 LOCATE 25,19:PRINT"TIME: ";TIME$;:LOCATE 25,49:PRINT"DATE: ";DATE$;
11420 RETURN                                'Return to caller
11440 '
```

```
11500 REM ---------------------
11510 REM INPUT FILE NAME
11520 '
11530 MG=25:GOSUB 13430:MG=20          'Print prompt message
11540 ON ERROR GOTO 11660              'Set error trap
11550 GOSUB 13930                      'SOUND input
11560 IP$="":LO=50:GOSUB 14505         'Null IP$ & wait for input
11570 IF IP$="" THEN 11590             'If IP$ is null EXIT
11580 IF ASC(IP$)<65 THEN GOSUB 13230:GOTO 11530 'Chk $ for legal 1st char
11590 FI$=IP$                          'Load into "filename" string var
11600 GOSUB 11330                      'Update status area
11610 GOSUB 13630                      'Clear messages
11620 ON ERROR GOTO 0                  'Reset error traps
11630 GOTO 2950                        'EXIT to S-Menu
11640 '
11650 ' - - -ERROR TRAP - - - -
11660 RESUME 11670                     'Set RESUME to next line
11670 GOSUB 13230:GOTO 11530           'Print error msg & re-ntr routine
11680 '
12000 REM ---------------------
12010 REM ROLL AM/PM
12020 '
12030 IF DT(LN)=0 OR DT(LN)=3 OR DA(LN)=5 THEN 3780 'EXIT if wrong DT or DA
12040 ON DT(LN) GOTO 12050, 12050, 12060 'Jump on data type
12050 LO=47:GOTO 12070                 'Set location & jump to next code
12060 LO=41                            'Set location & fall through
12070 AP$=CHR$(SCREEN(18,LO))          'Pick-off AM/PM data
12080 IF AP$="A" THEN AP$="P" ELSE AP$="A" 'Toggle AM/PM string
12090 IF DT(LN)<3 THEN AP$=LEFT$(AP$,1)  'Set to A/P if DT=1 or 2
12100 LOCATE 18,LO:PRINT AP$;          'Locate & print AM/PM string
12110 F2=1:GOTO 3780                   'EXIT to KB scan
12120 '
12200 REM ---------------------
12210 REM ROLL HOURS
12220 '
12230 IF DT(LN)=0 OR DA(LN)=5 THEN 3780  'EXIT if wrong DT or DA
12240 ON DT(LN) GOTO 12250, 12250, 12260 'Jump on data type
12250 LO=48:GOTO 12270                 'Set loc & jump to next code
12260 LO=42                            'Set loc & fall through
12270 LOCATE 18,LO                     'Set print location
12280 GOSUB 13120                      'Pick-off hours
12282 PR=PR+1                          'Increment hours
12285 IF DT(LN)=3 AND PR>23 THEN PR=1   'Chk: if diff >24 if DT is WT
12286 IF DT(LN)=3 THEN  12300          'Print hour if WT
12290 IF PR>12 THEN PR=1               'Inc & check hours for >12
12300 GOSUB 13160                      'Print new hour data
12310 F2=1:GOTO 3780                   'Set data altd flg & EXIT to KB scan
12320 '
```

```
12400 REM ----------------------
12410 REM ROLL MINUTES
12420 '
12430 IF DT(LN)=0 OR DA(LN)=5 THEN 3780  'Exit if wront DT or DA
12440 ON DT(LN) GOTO 12450, 12450, 12460 'Jump on data type
12450 LO=51:GOTO 12470                   'Set loc & jump to next code
12460 LO=45                              'Set location & fall thru
12470 LOCATE 18,LO                       'Set print position
12480 GOSUB 13120                        'Pick of hours
12490 PR=PR+1:IF PR>59 THEN PR=0         'Increment & check hours for >12
12500 GOSUB 13160                        'Print new hour data
12510 F2=1:GOTO 3780                     'Set data altd flag & EXIT to KB scan
12520 '
12600 REM ----------------------
12610 REM ROLL MONTHS
12620 '
12630 IF DT(LN)=1 AND DA(LN)=4 THEN 12640 ELSE 3780 'EXIT if wrong DT or DA
12640 LO=41:LOCATE 18,LO                 'Set print position
12650 '
12660 GOSUB 13120                        'Pick-off data
12670 PR=PR+1:IF PR>12 THEN PR=1         'Check months for >12
12680 GOSUB 13160                        'Print month data
12690 IF DT(LN)=1 THEN LOCATE 18,44:PRINT"01"; 'Reset days if month changed
12700 F2=1:GOTO 3780                     'Set flg: data altd & EXIT to KB scan
12710 '
12800 REM ----------------------
12810 REM ROLL DAYS
12820 '
12830 IF DA(LN)=5 THEN 3780              'EXIT if wrong DA
12840 IF DT(LN)=1 OR DT(LN)=2 THEN 12850 ELSE 3780 'EXIT if wrong DT
12850 ON DT(LN) GOTO 12860, 12970        'Jump on data type
12860 LO=41:GOSUB 13120                  'Pick-off month data
12870 MO=PR:YR=VAL(RIGHT$(DATE$,2))      'Get month data & current year value
12880 DY=VAL(MID$("312831303130313130313031",MO*2-1,2)) 'Set max days in month
12890 IF MO< VAL(DATE$) THEN YR=YR+1     'Chk: is alarm for next year?
12900 MO=VAL(PR$):IF MO<>2 THEN 12920    'Jump if month not February
12910 IF (YR/4)-INT(YR/4)=0 THEN DY=29   'Set max days if leap year
12920 LO=44:GOSUB 13120                  'Now, pick-off day data
12930 PR=PR+1:IF PR>DY THEN PR=1         'Increment days & check if >DY
12940 LOCATE 18,LO:GOSUB 13160           'Set print position & print days
12950 F2=1:GOTO 3780                     'Set flg: data altd & EXIT to KB scan
12960 '- - - - - - - - - - - - - -
12970 LO=41:GOSUB 13120                  'Pick-off day data
12980 PR=PR+1:IF PR>6 THEN PR=0          'Increment days & check if >6
12990 LOCATE 18,41                       'Set print position
13000 PRINT DY$(PR);                     'Print the day data
13010 F2=1:GOTO 3780                     'Set flg: dta altd & EXIT to KB scan
13020 '
13100 '- - - - -PICK-OFF TARGET STRING
13110 '
13120 PR$=CHR$(SCREEN(18,LO))+CHR$(SCREEN(18,LO+1)) 'Val from screen into PR$
13130 PR=VAL(PR$):RETURN                 'Convert $ to val & return to caller
13140 '
```

```
13150 '- - - - -PRINT VALUE IN TARGET AREA
13151 '
13160 IF PR>9 THEN PRINT RIGHT$(STR$(PR),2); 'Print values greater than 9
13170 IF PR<10 THEN PRINT "0"+RIGHT$(STR$(PR),1); 'Prnt values less than 10
13180 RETURN                                'Return to caller
13190 '
13200 REM ----------------------
13210 REM PRINT ERROR MESSAGE - SBR
13220 '
13230 LOCATE 9,2:PRINT SPC(78);             'Clear message line
13240 LOCATE 9,2:COLOR 15                   'Pos'n & set video attributes
13280 PRINT SPC(39-(LEN(MG$(MG))/2));       'Center message
13290 PRINT MG$(MG):SOUND 100,20            'Print message and make BEEP
13300 FOR X=1 TO 1000:NEXT X                'Delay
13310  COLOR 7:GOSUB 13630                  'Restor display & clr message
13311 ON ERROR GOTO 0                       'For good measure: reset error trap
13320 RETURN                                'Return to caller
13330 '
13400 REM ----------------------
13410 REM PRINT MESSAGE
13420 '
13430 IF SA=0 THEN SA=15                    'Set video attrib variable
13440 GOSUB 13630                           'Clear previous message, if any
13450 LOCATE 9,2:COLOR SA                   'Locate & set video attrib
13460 PRINT SPC(39-(LEN(MG$(MG))/2));       'Center message
13470 PRINT MG$(MG)                         'Print message
13480 COLOR 7:SA=0                          'Restore video attributes & variable
13490 RETURN                                'Return to caller
13540 '
13600 REM ----------------------
13610 REM CLEAR MESSAGE AREA
13620 '
13630 LOCATE 9,2                            'Set print position
13640 PRINT SPC(78)                         'Clear message area
13670 RETURN                                'Return to caller
13680 '
13700 REM ----------------------
13710 REM FIND LABEL (2 PARTS-LN TO END & 1 TO LN)
13720 ' LA=line number of found label: if LA=0, no label was found
13730 '
13740 '- - - - -FIND LABEL - CUR LINE TO END - SSBR
13750 '
13760 X=LN+1:LA=0                           'Set beginning search parameters
13770 IF DA(X)=5 THEN LA=X:GOTO 13800       'Search-EXIT if target found
13780 X=X+1                                 'Increment counter
13790 IF X<EN+1 THEN  13770                 'Loop to end of file
13800 RETURN                                'Return to caller
13810 '
13820 '- - - - -FIND LABEL - BEG TO CUR LINE - SSBR
13830 '
13840 X=1:LA=0                              'Not found; start at beg of file
13850 IF DA(X)=5 THEN LA=X:GOTO 13880       'Search-EXIT if target found
13860 X=X+1                                 'Increment counter
13870 IF X<LN+1 THEN  13850                 'Loop until
13880 RETURN                                'Return to caller
13890 '
```

```
13900 REM ---------------------
13910 REM "INPUT" SOUND
13920 '
13930 FOR X=1 TO 3:SOUND X*400,4:NEXT    'Make "INPUT" sound
13940 RETURN                             'Return to caller
13950 '
14200 REM ---------------------
14201 REM PRINT HELP SCREEN
14203 '
14210 FOR Z=0 TO 3                       'Set outer print loop
14215 LOCATE Z+2,2                       'Set print position
14220 FOR X=0 TO 9 STEP 2                'Set inner print position
14225 COLOR 15,0:PRINT HL$(X+(10*Z));:COLOR 0,7:PRINT HL$(X+((10*Z)+1));
14226                                    '   Print help screen
14230 NEXT X                             'Loop until done w/inner loop
14235 NEXT Z                             'Loop until done w/outer loop
14240 '----------
14245 COLOR 7,0:LOCATE 6,2:PRINT SPC(78);:LOCATE 7,4 'Erase line 6
14250 COLOR 0,7:PRINT HL$(40)+HL$(41)   'Print F1-F10 help
14255 '----------
14260 COLOR 15,0                         'Set color
14265 LOCATE 7, 3:PRINT" 1";             'Position & print funct. key numbers
14270 LOCATE 7,10:PRINT" 2";             '    "
14275 LOCATE 7,17:PRINT"  3";            '    "
14280 LOCATE 7,25:PRINT" 4";             '    "
14285 LOCATE 7,32:PRINT"  5";            '    "
14290 LOCATE 7,40:PRINT" 6";             '    "
14295 LOCATE 7,47:PRINT"  7";            '    "
14300 LOCATE 7,55:PRINT" 8";             '    "
14305 LOCATE 7,62:PRINT"  9";            '    "
14310 LOCATE 7,70:PRINT" 0";             '    "
14315 IF LEN(CT$)>0 THEN COLOR 0,7:LOCATE 7,54:PRINT"*"; 'Show buffer loaded
14318 COLOR 7,0                          'Reset video to normal
14320 LOCATE 8,2:PRINT SPC(78);          'Erase line #8
14325 LOCATE 9,2:PRINT SPC(78);          'Erase line #9
14335 RETURN                             'Return to caller
14340 '
```

```
14500 REM ---------------------------
14501 REM GENERAL INPUT ROUTINE
14502 '
14505 X=0                              'Set posn counter to 1
14510 '
14515 '- - - - - -KB SCAN- - - - -
14520 '
14525 LOCATE 9,LO:PRINT IP$+"_ ";      'Set posn & print $+cursor
14530 IK$=INKEY$:IF IK$="" THEN 14530  'Scan keyboard
14535 '
14540 '- - - - - -VALIDATE INPUT -
14545 '
14550 ON INSTR(V4$,IK$) GOTO 14615, 14640, 14660 'Jump on valid command
14555 '                        BSp  , CR , Esc
14560 IF INSTR(V2$,IK$)>0 THEN 14585   'Check for valid input
14565 IF ASC(IK$)>90 THEN IK$=CHR$(ASC(IK$)-32):GOTO 14560 '<NEW conv to l/c
14568 SOUND 37,1:GOTO 14530            'Invalid input-go to KB scan
14570 '
14575 '- - - - - -VALID CHAR - - -
14580 '
14585 X=X+1:IF X>24 THEN X=24:SOUND 37,1:GOTO 14530 'Inc cntr & chk for length
14590 IP$=IP$+IK$                      'Add valid input to string
14595 GOTO 14525                       'Re-enter keyboard scan
14600 '
14605 '- - - - - -BACK SPACE - - -
14610 '
14615 X=X-1:IF X<0 THEN X=0:IP$="":GOTO 14525 'Dec counter & chk for length
14620 IP$=LEFT$(IP$,X):GOTO 14525      '-1 char & re-enter KB scan
14625 '
14630 '- - - - - -CARRIAGE RTN - -
14635 '
14640 RETURN                           'Return to caller
14645 '
14650 '- - - - - -ESC- - - - - - -
14655 '
14660 IP$="":GOTO 14640                'Null input & EXIT thru routine
14665 '
20000 GOTO 3870 '-------DEAD END INPUTS---------END PROGRAM
```