PAGE

DOS 3.3 O 0

Create Hi-Res graphics and save them on disk as screens or program lines. This Applesoft program features a Help screen and options to let you draw, erase, and change colors.

ProDOS O 0

by David Krathwohl

Programmer's Aid for Graphics Entry (PAGE) is a programming tool designed to make Applesoft's HPLOT graphics function a viable alternative to the intricacies of shape tables. It accomplishes this by providing an easy, cursor-oriented method of creating graphic displays, and by allowing you to save these displays as either text files of Applesoft program lines or as binary files containing Hi-Res screen images.

MOVING AND DRAWING

When you first run PAGE, you will be asked to choose a color. These are given in the range 0 to 7 and correspond to the normal Hi-Res colors described in the Applesoft manual. Once that choice is made, you are presented with a blank graphics screen with a small, blinking dot in the center. That dot is your cursor, and its up-down-left-right motions are controlled by the up, down, left and right arrows. The A key also moves the cursor up, and the Z key moves it down.

Graphic designs are created by designating a starting point at the cursor position, moving the cursor to another position and then connecting the two points. You start a line by pressing the S key at the appropriate point. You can then move the cursor using the four arrow keys and the A and Z keys. When the cursor is in the correct position, simply press the space bar, and a line will be plotted connecting the two points. If you wish to continue plotting from that point, move the cursor to a new position and press the space bar again. If you wish to start a new line separate from the last, move to a new position and press the S key again. PAGE will not respond to a press of the space bar unless you have first started the drawing with the S key.

MORE FUNCTIONS

While these six keys can be used to create almost any design, the addition of a few more keys and functions greatly speeds the process. First, the initial "step" taken by the cursor at each press of a key is one screen dot. Since the Apple's screen is 280 dots wide by 192 dots high, it would take some time to cross the screen. To change the cursor "step" size, press any of the digits from 1 to 9. This value will be the number of dots in each of the following steps. It can be reset at any time during your drawing.

To utilize the color capabilities of the Hi-Res screen, press the C key. The four bottom lines on the screen will display the eight color choices and their corresponding numbers. The number of the present color is displayed in inverse video, and the screen prompts you to make a new choice. If you decide you like the present color after all, just press the <RETURN> key and return to your drawing. Colors may be changed as often as you like, but each time a new color is chosen, you must start a new line. If you forget to do this, you will be reminded of the necessity by a bell and an error message. When changing color, keep in mind the fact that colors 1 and 5 will only appear on odd numbered X-coordinates, while colors 2 and 6 will only appear on even numbered X-coordinates.

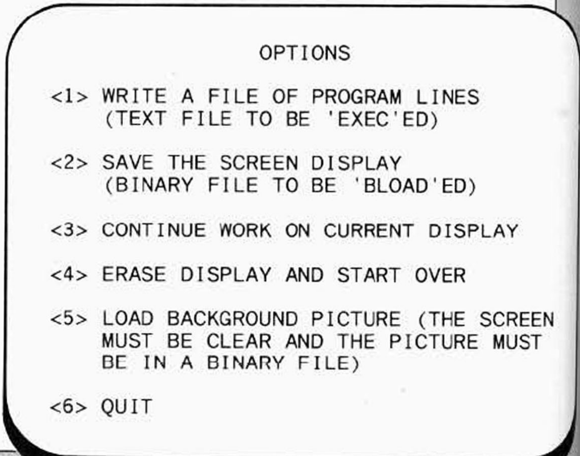
Since it is inevitable that you will draw a line that you decide you really don't want, an eraser is provided. Pressing the E key erases either the last line drawn or the last starting point. If you press E several times consecutively, and in the process back up into another color, you will be told that you have done so, and informed of the current color.

It is likely that some of your lines will cross each other in an intricate drawing. Since erasing a line that crosses another will also erase a small part of the original line, PAGE provides a way of restoring your drawing to its original state. Just press the R (redraw) key, and the holes will vanish.

For those programmers who like to know where they are, the P key helps you find your place. Press this key to display the X- and Y-coordinates of the present cursor position at the bottom of the screen.

When you press the H key, you will be shown two screen pages that describe the keys and their functions. Though text is displayed,

FIGURE 1: Main Menu



```
                    OPTIONS

<1> WRITE A FILE OF PROGRAM LINES
    (TEXT FILE TO BE 'EXEC'ED)

<2> SAVE THE SCREEN DISPLAY
    (BINARY FILE TO BE 'BLOAD'ED)

<3> CONTINUE WORK ON CURRENT DISPLAY

<4> ERASE DISPLAY AND START OVER

<5> LOAD BACKGROUND PICTURE (THE SCREEN
    MUST BE CLEAR AND THE PICTURE MUST
    BE IN A BINARY FILE)

<6> QUIT
```

your drawing is not affected. When you have finished reading the Help screens, you will return to your previous position and color on the Hi-Res screen.

SAVING YOUR GRAPHICS

When you have finished creating your drawing, pressing <ESC> will take you out of the drawing mode and present you with the options shown in **Figure 1**. Option 1 makes PAGE a programmer's tool rather than merely an artist's tool. This option creates a text file containing a sequence of Applesoft program lines. You will be asked to specify both the text file name and the starting line number for the sequence. Since PAGE requires that you choose a color before you do any drawing, the first program line will contain an HCOLOR= statement. The following lines may contain other HCOLOR assignments or HPLOT statements. The latter will contain the end points of each line segment you have specified by pressing the S key and space bar in the drawing mode.

For those unfamiliar with the use of text files, a short explanation is in order. One of the more powerful Apple DOS commands is the EXEC command. This causes a text file to be read and the statements within it to be executed sequentially. When these statements are immediate mode commands, each action is completed before the next is begun. When the statements are preceded by numbers, the Apple interprets them as line numbers for a BASIC program. In this case, they are added to memory either as a program in themselves, or as a part of the program currently in memory. (They will replace any currently existing line with the same line number.) Once EXECed, the program in memory may be saved as an Applesoft file for future use. Remember, you cannot RUN the text file created by PAGE, but you can RUN and/or SAVE the BASIC program that results from EXECing the text file.

Before exploring some of the particular advantages of saving your program as Applesoft program lines, let's first look at the other options in the list.

Option 2 allows the more customary BSAVEing of the entire Hi-Res screen display. The screen may be re-displayed by BLOADing the file while displaying Hi-Res graphics.

Option 3 allows you to return to your drawing without changing it. This option can be used either after saving your drawing using option 1 or 2, or as a way of recovering from an accidental press of the escape key.

Option 4 is used to clear the screen and begin again. If you choose this option, you will be asked to confirm your choice before any action is taken.

The fifth option can be used at the beginning of a session (or in the middle if you use option 4 to clear the screen first) to load a background picture from a file on disk. This is particularly helpful when you are writing a program that successively adds images onto an existing drawing. Each stage can be saved as both a binary and a text file. The binary file can be used to provide orientation as a background picture, while each successive text file can be used to supply the graphics subroutines in the finished program.

The last option is simply the graceful way to quit. It requests confirmation before clearing the screen and returning you to Applesoft.

ADVANTAGES

The real advantages of using PAGE text files are the increased speed of graphic presentation and the ability to create dynamic graphics in Applesoft.

While it is a common practice to interrupt a program with a BLOAD command to display a picture, any disk access time is time wasted from the user's point of view. It also requires that the program disk remain in the drive. While there are times when this is the preferred solution, it is more often used because it is the only solution. A sequence of Applesoft plotting commands, on the other hand, executes very quickly, and does not require that the disk be in the drive.

Prior to PAGE, however, writing these commands was very tedious work. It required working with a 280 by 192 grid and translating X- and Y-coordinates into the appropriate HPLOT statements. PAGE makes all of this unnecessary, and thus makes Applesoft HPLOT graphics a very attractive alternative.

Another advantage to HPLOT graphics is that pictures may be built incrementally by successively calling subroutines. While you won't be able to achieve the animation effects possible with shape tables and the XDRAW command, you can certainly achieve a much more dynamic display than is possible by BLOADing picture files from the disk.

ENTERING THE PROGRAM

To key in PAGE, enter the Applesoft program shown in **Listing 1** and save it on disk with the command:

```
SAVE PAGE
```

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

HOW THE PROGRAM WORKS

PAGE (**Listing 1**) consists of two major program sections: the drawing section and the filing section. Each of these sections is then further broken down into a controlling routine and a series of subroutines that it calls.

When PAGE is first run, a short initialization routine (**line 1170**) is called. This POKEs in the shape table for the cursor dot, dimensions the array that holds the X- and Y-coordinates of your drawing, sets the initial position of the cursor, and prints the title. It then calls the subroutine for choosing a color (at **line 520**) before clearing

TABLE 1: PAGE Variables

Variable	Description
CF	Cursor flag 1 (on/off state of cursor)
CU	Cursor counter (controls the blink rate)
CL	Cursor flag 2 (to erase cursor from starting position)
DA(*n*)	Data array
DI	Data array index
FA	Number of dots per cursor move
FL	Line number increment for text file
H	Pause loop index
HU	Color
I	Loop index for text file
K$,K	Miscellaneous keypresses
LC	Counter for number of midpoints on a program line
LE	Line number of error
LN$,LN	First line number for text file
MV	Value of keyboard buffer PEEK
ML	Machine language data for shape table
N$	File name
NE	Error number
SF	Flag for start of line
X,Y	Present cursor coordinates
XT,YT	Temporary cursor position
XL,YL	Last plotted cursor position
YB	Flag for bottom four lines of screen bell

the graphics screen, setting the scale and starting the main program section. Notice that before initialization takes place, LOMEM is set at 17000 to protect the Hi-Res display from intrusion by variables, and an ONERR GOTO is set to handle any errors that might occur. (For a description of the variables used in PAGE, see **Table 1**.)

PART ONE: DRAWING

The drawing section of the program is made up of the cursor loop beginning at **line 100** and ending at **line 320**. Within this loop, the keyboard is PEEKed and the cursor is drawn and erased so that it appears to blink. The variables CU and CF are used to keep track of the cursor and make it blink at a comfortable rate. The variable CL is used only the first time through the loop to erase the cursor at its initial position.

When a key is pressed, the value of the keyboard PEEK, held in the variable MV, will be greater than 127, and the program branches to the conditional statements at **line 160**. Here, MV is tested against the ASCII values of the function keys. If one of the movement keys has been pressed, the value of either X or Y is incremented or decremented (depending on the move) by the value stored in FA. This variable holds the value representing the number of dots in each move, and is changed by pressing any of the digit keys (except zero).

Colors may be changed as often as you like.

Notice that if the value of the new X- or Y-coordinate exceeds the screen limitations of the Apple, a beep sounds and the value of the coordinate is restored to the previous move.

Since the four bottom lines of the screen can be used to display part of the text screen, it is important to know when you have entered this portion of the screen. The tests at **lines 220 and 230** check for this condition and set a flag which causes the bell to sound whenever that imaginary line is crossed.

When the S key is pressed, the subroutine at **line 350** is called. Here the array, DA(*n*), is used to hold a flag value of −1 and the values of the X- and Y-coordinates. The variable DI is used to increment the array. The point is plotted and its coordinate values stored in XL and YL for future reference. Finally, a flag variable, SF, is set to indicate that a starting point has been chosen.

When the space bar is pressed, indicating that the present cursor position is a midpoint, the new X- and Y-coordinates are again stored in the array and a line is plotted from XL,YL to the new position. A check of the start flag (SF) at the beginning of the routine prevents drawing a midpoint to a line that has not yet been started.

Erasing a line calls the subroutine at **line 400**, which re-draws the last line in color 0 (black) and decrements the DI counter. A special section at **line 440** checks for a color change and resets the color to the previous one if a change is found. This is done by counting backwards in the array to find the most recent value of −2, which is the color change flag.

Color changing is done at **line 510**. Here the lower four lines of the text screen are used to display the choices of color. **Lines 530-590** test the present color value (stored in variable HU) and print it in inverse video at the proper position. If no change in color is made, the routine is exited without adding values to the array, DA(*n*). If a new color is chosen, the color flag value, −2, is stored in the array and following it, the new color value is stored.

Re-drawing the screen takes place at **line 640**. To accomplish this, the DA(*n*) array is read from beginning to end and its values translated into the appropriate HCOLOR and HPLOT statements. Whenever a value of −1 is encountered, the next two values are interpreted as the X- and Y-coordinates of a starting point, and the point is plotted using HPLOT. If the next value is neither a −1 nor a −2, then the next two points are interpreted as the X- and Y-coordinates of a midpoint, and an HPLOT TO command is issued. A value of −2 indicates a color change, and the value of the array element following the −2 is assigned in an HCOLOR statement.

Both the P(osition) and H(elp) keys call simple text printing routines. The first uses only the lower four lines of the screen, and the latter uses the full text display. The Help routine starts at **line 720** and the Position routine starts at **line 700**.

PART TWO: THE FILING SYSTEM

The second part of PAGE consists of the menu of filing options and their associated subroutines. The first option, creating a text file of program lines, calls a subroutine at **line 1040**. This subroutine uses the same technique to read the DA(*n*) array as is used by the R (re-draw) routine. The only difference is the use of additional variables to keep track of the line numbers and the number of statements on each line. Each starting point and each color assignment is put on a separate program line. As many as ten midpoints are put on a single program line in a single **HPLOT TO X,Y TO X1,Y1 TO X2,Y2. . .** statement.

The variable LN holds the current line number, while FL is used to increment it. LC is used to count the number of midpoints contained in a single program line.

The other five options are fairly straightforward subroutines. Saving the screen as a binary file (option 2) first requests a file name and then executes a BSAVE command. Option 3 switches back to the graphics mode and re-starts the cursor loop. Option 4 relies on the HGR statement to clear the screen and the CLEAR statement to zero all variables before starting over. Option 5 checks the DI counter before BLOADing a picture file to be sure it doesn't destroy your artwork. Finally, option 6 clears the screen and ENDs the program.

PROGRAM MODIFICATIONS

There are undoubtedly many ways of modifying PAGE. You might want to change some of the function keys to make them easier for you to remember, or perhaps even add your own functions. (An automatic method of drawing curves would be an excellent addition.) Paddle or joystick control of the cursor is another possibility.

However, if you do consider making additions, note that the program presently occupies nearly all of the program space below the Hi-Res screen. Any lengthy additions will require some compression and/or deletion, or perhaps the relocation of the program above the Hi-Res screen.

LISTING 1: PAGE

```
10  REM  ************************
20  REM  *        PAGE          *
30  REM  *  BY  D.A. KRATHWOHL  *
40  REM  *  COPYRIGHT (C) 1985  *
50  REM  *  BY MICROSPARC, INC  *
60  REM  *  CONCORD, MA. 01742  *
70  REM  ************************
8Ø  LOMEM: 17ØØØ: ONERR  GOTO 122Ø
9Ø  GOSUB 117Ø
1ØØ MV =  PEEK ( - 16384): IF DI > 1995 THEN
     127Ø
11Ø  IF MV > 127 AND CF > Ø THEN  XDRAW 1 AT
    X,Y:CF = Ø: GOTO 16Ø
12Ø  IF MV > 127 THEN 16Ø
13Ø  IF CU = Ø THEN  XDRAW 1 AT X,Y:CF = CF +
    1: IF CF >  = 2 THEN CF = Ø
14Ø CU = CU + 1: IF CU > 1Ø THEN CU = Ø
15Ø  GOTO 1ØØ
16Ø  POKE  - 16368,Ø: XDRAW 1 AT XT,YT: IF CL
     = Ø THEN  XDRAW 1 AT 14Ø,96:CL = 1
17Ø  IF MV > 176 AND MV < 186 THEN FA =  VAL
    ( CHR$ (MV - 128))
18Ø  IF MV = 136 THEN X = X - FA: IF X < Ø THEN
     PRINT  CHR$ (7):X = X + FA
19Ø  IF MV = 149 THEN X = X + FA: IF X > 279 THEN
     PRINT  CHR$ (7):X = X - FA
2ØØ  IF MV = 193 OR MV = 139 THEN Y = Y - FA:
     IF Y < Ø THEN  PRINT  CHR$ (7):Y = Y +
    FA
21Ø  IF MV = 218 OR MV = 138 THEN Y = Y + FA:
     IF Y > 191 THEN  PRINT  CHR$ (7):Y = Y -
    FA
22Ø  IF Y > 16Ø AND YB = Ø THEN YB = 1: PRINT
     CHR$ (7)
23Ø  IF Y < 161 AND YB = 1 THEN YB = Ø: PRINT
     CHR$ (7)
24Ø  IF MV = 211 THEN  GOSUB 35Ø: REM S
25Ø  IF MV = 197 THEN  GOSUB 4ØØ: REM E
26Ø  IF MV = 21Ø THEN  GOSUB 64Ø: REM R
27Ø  IF MV = 2ØØ THEN  GOSUB 72Ø: REM H
28Ø  IF MV = 195 THEN  GOSUB 51Ø: REM C
29Ø  IF MV = 16Ø THEN  GOSUB 37Ø: REM SPC
3ØØ  IF MV = 2Ø8 THEN  GOSUB 7ØØ: REM  P
31Ø  IF MV = 155 THEN 81Ø: REM ESC
32Ø  XDRAW 1 AT X,Y:XT = X:YT = Y: GOTO 1ØØ
33Ø  REM  END OF CURSOR LOOP
34Ø  REM  START PT
35Ø DI = DI + 1:DA(DI) =  - 1:DI = DI + 1:DA(
    DI) = X:DI = DI + 1:DA(DI) = Y: HPLOT X,
    Y:XL = X:YL = Y:SF = 1: RETURN
36Ø  REM  * CONNECT MID POINT *
37Ø  IF SF = Ø THEN  PRINT  CHR$ (7): GOSUB 1
    28Ø: RETURN
38Ø DI = DI + 1:DA(DI) = X:DI = DI + 1:DA(DI)
     = Y: HPLOT XL,YL TO X,Y:XL = X:YL = Y: RETURN

39Ø  REM  ERASE PT
4ØØ  IF SF = Ø THEN  PRINT  CHR$ (7): GOSUB 1
    28Ø: RETURN
41Ø  IF DI <  = 5 THEN X = DA(DI - 1):Y = DA(
    DI):SF = Ø:DI = 2: HCOLOR= Ø: HPLOT X,Y:
     HCOLOR= HU: RETURN
42Ø  IF DA(DI - 2) >  - 1 THEN 48Ø
43Ø  HCOLOR= Ø: HPLOT DA(DI - 1),DA(DI): HCOLOR=
    HU:X = DA(DI - 4):Y = DA(DI - 3):XL = X:
    YL = Y:DI = DI - 3
44Ø  IF DA(DI - 1) =  - 2 THEN  PRINT  CHR$ (
    7): HOME : POKE  - 163Ø1,Ø: VTAB 21: HTAB
    9: PRINT " YOU HAVE ERASED A COLOR ": HTAB
    9: PRINT " CHANGE. COLOR IS NOW  ";: GOSUB
    46Ø: GOSUB 13ØØ: POKE  - 163Ø2,Ø:X = DA(
    DI - 3):Y = DA(DI - 2):XL = X:YL = Y:DI =
    DI - 2: RETURN
45Ø  RETURN
46Ø  FOR I = DI - 2 TO 1 STEP  - 1: IF DA(I) =
     - 2 THEN HU = DA(I + 1): PRINT HU: HCOLOR=
    HU: RETURN
47Ø  NEXT I: RETURN
48Ø  HCOLOR= Ø: HPLOT DA(DI - 3),DA(DI - 2) TO
    DA(DI - 1),DA(DI): HCOLOR= HU:X = DA(DI -
    3):Y = DA(DI - 2):XL = X:YL = Y: IF DA(D
    I - 4) =  - 1 THEN  HPLOT DA(DI - 3),DA(
    DI - 2)
49Ø DI = DI - 2: RETURN
5ØØ  REM  COLOR
51Ø  PRINT : VTAB 21: CALL  - 958: POKE  - 16
    3Ø1,Ø
52Ø  PRINT "COLOR OPTIONS:": POKE 33,9: POKE
    32,18: VTAB 21: PRINT "Ø BLACK": PRINT "
    1 GREEN": PRINT "2 VIOLET": PRINT "3 WHI
    TE";: POKE 32,31: VTAB 2Ø: PRINT : PRINT
    "4 BLACK2": PRINT "5 ORANGE": PRINT "6 B
    LUE": PRINT "7 WHITE2";: POKE 33,4Ø: POKE
    32,Ø
53Ø  IF HU = Ø OR HU = 4 THEN  VTAB 21
54Ø  IF HU = 1 OR HU = 5 THEN  VTAB 22
55Ø  IF HU = 2 OR HU = 6 THEN  VTAB 23
56Ø  IF HU = 3 OR HU = 7 THEN  VTAB 24
57Ø  IF HU >  - 1 AND HU < 4 THEN  HTAB 19: GOTO
    59Ø
58Ø  HTAB 32
59Ø  INVERSE : PRINT HU;: NORMAL
6ØØ  HTAB 1: VTAB 24: PRINT "CHOICE:";: GET K
    $: IF K$ <  >  CHR$ (13) AND ( ASC (K$) <
    48 OR  ASC (K$) > 55) THEN 6ØØ
61Ø  IF K$ =  CHR$ (13) OR HU =  VAL (K$) THEN
     POKE  - 163Ø2,Ø: RETURN
62Ø DI = DI + 1:DA(DI) =  - 2:SF = Ø:HU =  VAL
    (K$): HCOLOR= HU:DI = DI + 1:DA(DI) = HU
    : POKE  - 163Ø2,Ø: RETURN
63Ø  REM  RE-DRAW
64Ø  FOR I = 1 TO DI
65Ø  IF DA(I) =  - 1 THEN  HPLOT DA(I + 1),DA
    (I + 2):I = I + 2: NEXT I
66Ø  IF DA(I) =  - 2 THEN  HCOLOR= DA(I + 1):
    I = I + 1: NEXT I
67Ø  IF I < DI THEN  HPLOT  TO DA(I),DA(I + 1
    ):I = I + 1: NEXT I
68Ø  RETURN
69Ø  REM  COORDS
7ØØ  HOME : POKE  - 163Ø1,Ø: VTAB 22: HTAB 12
    : PRINT "X:";X;: HTAB 27: PRINT "Y:";Y: GOSUB
    13ØØ: POKE  - 163Ø2,Ø: RETURN
71Ø  REM  HELP
72Ø  POKE  - 163Ø1,Ø: POKE  - 16298,Ø: TEXT :
     HOME : VTAB 2: INVERSE : PRINT " KEY"; SPC(
    14);"FUNCTION"; SPC( 14): NORMAL
73Ø  POKE 32,2: PRINT : PRINT : PRINT "->": PRINT
    : PRINT "<-": PRINT : PRINT "A": PRINT :
     PRINT "Z": PRINT : PRINT "1...9": PRINT
    : PRINT "<ESC>"
74Ø  POKE 32,1Ø: VTAB 4: PRINT : PRINT "MOVE
    CURSOR RIGHT": PRINT : PRINT "MOVE CURSO
    R LEFT": PRINT : PRINT "MOVE CURSOR UP":
     PRINT : PRINT "MOVE CURSOR DOWN": PRINT
    : PRINT "SET NUMBER OF DOTS PER MOVE": PRINT
    : PRINT "MENU OPTIONS": POKE 32,Ø
75Ø  GOSUB 13ØØ: POKE 34,2: HOME
76Ø  POKE 32,2: VTAB 4: PRINT : PRINT "S": PRINT
    : PRINT "<SPC>": PRINT : PRINT "C": PRINT
    : PRINT "E": PRINT : PRINT "R": PRINT : PRINT
    "P": PRINT : PRINT "H"
77Ø  POKE 32,1Ø: VTAB 4: PRINT : PRINT "START
     A LINE": PRINT : PRINT "CONNECT TWO POI
    NTS": PRINT : PRINT "CHANGE COLORS": PRINT
    : PRINT "ERASE A LINE OR POINT": PRINT :
     PRINT "RE-DRAW (FILLS ERASURE GAPS)": PRINT
    : PRINT "READ PRESENT COORDINATES": PRINT

78Ø  PRINT "REQUEST HELP": POKE 32,Ø: POKE 34
    ,Ø
79Ø  GOSUB 13ØØ: PRINT : HOME : GR : POKE  -
    163Ø2,Ø: POKE  - 16297,Ø: RETURN
8ØØ  REM  CONT MENU
81Ø  TEXT : HOME : HTAB 16: INVERSE : PRINT "
     OPTIONS ": NORMAL : VTAB 5: PRINT " <1>
     WRITE A FILE OF PROGRAM LINES": PRINT "
         (TEXT FILE TO BE 'EXEC'ED)": PRINT
    : PRINT " <2> SAVE THE SCREEN DISPLAY": PRINT
    "      (BINARY FILE TO BE 'BLOAD'ED)": PRINT

82Ø  PRINT " <3> CONTINUE WORK ON CURRENT DIS
    PLAY": PRINT : PRINT " <4> ERASE DISPLAY
     AND START OVER": PRINT : PRINT " <5> LO
    AD BACKGROUND PICTURE (THE SCREEN      MU
    ST BE CLEAR AND THE PICTURE MUST      BE
     IN A BINARY FILE)": PRINT : PRINT " <6>
     QUIT"
83Ø  VTAB 22: HTAB 2Ø: GET K$:K =  VAL (K$): IF
    K < 1 OR K > 6 THEN 81Ø
84Ø  ON K GOTO 95Ø,95Ø,86Ø,88Ø,91Ø,1Ø1Ø
```

```
850  REM  CONT WORK
860  HOME : GR : POKE  - 16297,Ø: POKE  - 163
     Ø2,Ø: GOTO 32Ø
870  REM  ERASE/RE-START
880  HOME : VTAB 12: HTAB 15: INPUT "ERASE? (
     Y/N)";K$: IF  LEFT$ (K$,1) = "Y" THEN  CLEAR
     : GOTO 9Ø
890  GOTO 81Ø
900  REM  BACKGROUND
910  IF DI > 5 THEN  HOME : PRINT  CHR$ (7): VTAB
     12: PRINT "ERASE THE SCREEN BEFORE LOADI
     NG A BACK- GROUND PICTURE.": POKE  - 163
     68,Ø: GOSUB 13ØØ: GOTO 81Ø
920  HOME : VTAB 12: INPUT "DO YOU WANT TO SE
     E A DISK CATALOG? ";K$: IF  LEFT$ (K$,1)
      = "Y" THEN  PRINT D$"CATALOG": PRINT
930  GOSUB 135Ø: HGR : POKE  - 163Ø2,Ø: PRINT
     D$"BLOAD";N$: GOTO 32Ø
940  REM  B FILE
950  TEXT : HOME : IF DI < 5 THEN  PRINT  CHR$
     (7): VTAB 12: HTAB 7: PRINT "THERE IS NO
     THING TO SAVE.": POKE  - 16368,Ø: GOSUB
     13ØØ: GOTO 81Ø
960  IF K = 1 THEN 1Ø4Ø
970  HTAB 11: INVERSE : PRINT " SAVE SCREEN D
     ISPLAY ": NORMAL
980  GOSUB 134Ø: HOME : VTAB 12: HTAB 13: FLASH
     : PRINT " SAVING FILE ": NORMAL : PRINT
     D$"BSAVE ";N$;",A$2ØØØ,L$2ØØØ"
990  GOTO 81Ø
1000  REM  QUIT
1010  HOME : VTAB 12: HTAB 15: INPUT "EXIT? (
     Y/N)";K$: IF  LEFT$ (K$,1) = "Y" THEN  HOME
     : END
1020  GOTO 81Ø
1030  REM  TXT FILE
1040  HTAB 5: INVERSE : PRINT " WRITE A FILE
     OF PROGRAM LINES ": NORMAL
1050  GOSUB 134Ø: PRINT
1060  PRINT : PRINT "FIRST LINE NUMBER:";: INPUT
     "";LN$:LN =  VAL (LN$): IF LN < 1 OR LN >
     6ØØØØ THEN  INVERSE : PRINT "YOUR NUMBER
      MUST BE BETWEEN 1 AND 6ØØØØ": NORMAL : PRINT
     : GOTO 1Ø6Ø
1070  HOME : VTAB 12: HTAB 1Ø: FLASH : PRINT
     " WRITING TEXT FILE ": NORMAL : PRINT D$
     "OPEN "N$: PRINT D$"CLOSE": PRINT D$"DEL
     ETE "N$: PRINT D$"OPEN "N$: PRINT D$"WRI
     TE "N$:FL = Ø:LC = Ø: FOR I = 1 TO DI
1080  IF DA(I) =  - 1 THEN  PRINT LN + FL;"HP
     LOT ";DA(I + 1);",";DA(I + 2):FL = FL +
     1:I = I + 2: NEXT I: GOTO 114Ø
1090  IF DA(I) =  - 2 THEN  PRINT LN + FL;"HC
     OLOR=";DA(I + 1):FL = FL + 1:I = I + 1: NEXT
     I: GOTO 114Ø
1100  PRINT LN + FL;"HPLOT";
1110  PRINT " TO ";DA(I);",";DA(I + 1);:I = I
      + 1:LC = LC + 1: IF DA(I + 1) > Ø AND L
     C < 1Ø AND I < DI THEN I = I + 1: GOTO 1
     11Ø
1120  IF LC > Ø THEN LC = Ø: PRINT  CHR$ (13)
     :FL = FL + 1
1130  NEXT I
1140  PRINT D$"CLOSE "N$: GOTO 81Ø
1150  HOME : END
1160  REM  INIT
1170  POKE 232,Ø: POKE 233,3: FOR I = Ø TO 5:
      READ ML: POKE 768 + I,ML: NEXT I: DATA
     1,Ø,4,Ø,4,Ø,45:YT = 96:XT = 14Ø:X = 14Ø:
     Y = 96: DIM DA(2ØØØ):D$ =  CHR$ (4)
1180  TEXT : HOME : VTAB 6: INVERSE : POKE 32
     ,15: PRINT : PRINT "P": PRINT "A": PRINT
     "G": PRINT "E": NORMAL : POKE 32,16: VTAB
     6: PRINT : PRINT "ROGRAMMER'S": PRINT "I
     D FOR": PRINT "RAPHICS": PRINT "NTRY": POKE
     32,Ø: PRINT
1190  HTAB 12: PRINT "BY DAVID KRATHWOHL": PRINT
     : PRINT : PRINT "** COPYRIGHT 1985 BY MI
     CROSPARC, INC. **": GOSUB 13ØØ
1200  HOME : VTAB 1Ø: PRINT "PLEASE CHOOSE YO
     UR STARTING COLOR:":HU = 3: GOSUB 51Ø: IF
     K$ =  CHR$ (13) THEN K$ = "3": GOSUB 62Ø
1210  HGR : POKE  - 163Ø2,Ø: SCALE= 1: HCOLOR=
     HU:FA = 1: RETURN
1220  REM  ERR TRAPS
1230  PRINT D$"CLOSE": TEXT : HOME : VTAB 12:
      PRINT  CHR$ (7);:NE =  PEEK (222):LE =
      PEEK (218) +  PEEK (219) * 256: IF NE >
     Ø AND NE < 22 THEN 126Ø
1240  PRINT "   **ERROR #";NE;" IN LINE #";LE
1250  PRINT : PRINT "      PRESS ANY KEY TO RE
     TURN TO MENU": PRINT : HTAB 2Ø: GET K$: PRINT
     : GOTO 81Ø
1260  PRINT "   DISK ERROR #";NE: PRINT : PRINT
     : GOTO 125Ø
1270  TEXT : HOME : VTAB 12: PRINT "YOU ARE O
     UT OF SPACE.  SAVE THIS PICTUREAND START
      AGAIN.": POKE  - 16368,Ø: GOSUB 13ØØ: GOTO
     81Ø
1280  HOME : POKE  - 163Ø1,Ø: VTAB 22: HTAB 8
     : INVERSE : PRINT " PRESS <S> TO START A
      LINE ": NORMAL : FOR H = 1 TO 2ØØØ: NEXT
     : POKE  - 163Ø2,Ø: RETURN
1290  REM  PRESS KEY
1300  VTAB 24: PRINT "        <PRESS ANY KEY T
     O CONTINUE>     ";
1310  K =  PEEK ( - 16384): IF K < 127 THEN 13
     1Ø
1320  POKE  - 16368,Ø: RETURN
1330  REM  FILE NAME
1340  VTAB 6
1350  INPUT "FILE NAME:";N$: IF  VAL (N$) <  >
     Ø OR  LEN (N$) > 15 OR N$ = "" THEN  PRINT
      CHR$ (7): GOTO 134Ø
1360  RETURN
```

END OF LISTING 1

KEY PERFECT 4.Ø
RUN ON
PAGE

CODE	LINE# - LINE#
621D	1Ø - 1ØØ
968D	11Ø - 2ØØ
69Ø2	21Ø - 3ØØ
8869	31Ø - 4ØØ
E2B4	41Ø - 5ØØ
A2BC	51Ø - 6ØØ
98F9	61Ø - 7ØØ
Ø11D17	71Ø - 8ØØ
F5CØ	81Ø - 9ØØ
C14A	91Ø - 1ØØØ
EFEF	1Ø1Ø - 11ØØ
EA5C	111Ø - 12ØØ
CEB2	121Ø - 13ØØ
374F	131Ø - 136Ø

PROGRAM CHECK IS : 1497