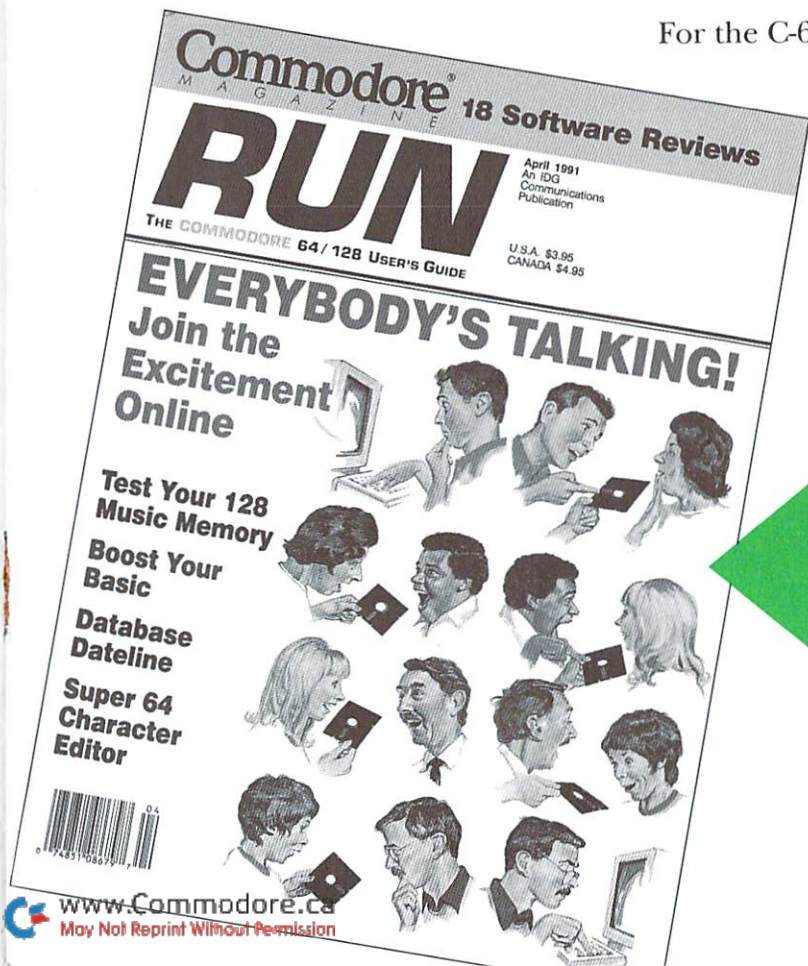


March/April 1991 Edition

# RE<sub>≡</sub>RUN

## RUN Programs on Disk

For the C-64 and 128



*Plus:  
Extra  
Bonus  
Programs!*

# Introduction

*March/April 1991 ReRUN*

IT'S DISTURBING TO SEE Commodore 64 and 128 owners get short-shrifted time and again by the software industry. Just a few years ago, software titles appeared for the C-64 first, and only afterwards would versions appear for other computer systems, such as MS-DOS machines, Macintoshes and Amigas.

Welcome to 1991, when new C-64 software titles appear sporadically, and new C-128 programs are nearly extinct. By providing C-64 and C-128 software as primary goods, rather than as an afterthought, ReRUN will do its part to eliminate disenchantment. In spite of the dwindling industry-wide software support, 1991 already looks like a banner year for ReRUN. We plan to continue our role as the number one source of Commodore 8-bit software.

In a slight deviation from the norm, this edition of ReRUN offers nothing but productivity-oriented programs. This is reflected in the first program, which is a collection of six utilities from our March/April RUN, entitled **Boosting Basic**. Written by programmer Stephen Dirschauer, these utilities give your C-64 a Search and Replace function, help screens, a pause key, a directory display, a sequential file reader and the ability to display 16 sprites simultaneously.

In response to our readers' requests for more application programs, we offer **Smart Shopper**, better known as Super Aisle Organizer. This specialized 64- and 128-mode database works by allowing you to enter the contents of each aisle from your most frequently-shopped supermarkets, then it generates an aisle-by-aisle list so that you won't miss a thing on your next shopping trip.

With a name such as **Super Character Editor**, experienced computerists probably expect our third program to be a font editor. Super Character Editor is that and more. It not only creates fonts for various graphic modes, but also saves all of the color attributes assigned to the modified fonts.

**Pitch-Memorizer** is from RUN's March/April 128 Mode column. After hearing a tone played for one second in duration, you must match the tone of the note by pressing on keys 1 through 5. Players are judged on their ability (. . . or lack thereof!) to reproduce a note close to the first note sounded.

Bonus programs found on this edition of ReRUN begin with **ADE 64**, an ASCII disk editor. Written by Greg Badros, author of GEOS Disk Editor (ReRUN November/December 1990), ADE 64 is a more compact machine language disk editor. Written in machine language for the C-64, ADE 64 allows you to edit disks in ASCII mode instead of Commodore PETSCII mode. The advantages are obvious to any programmers who have suffered through the conversions needed when editing ASCII-based disks.

Buttons, the mouse driver utility from the January/February '91 ReRUN, is used in this edition's **Pop-Up Calculator**. Utilizing the convenience of the Buttons program, Calculator gives your C-64 an interrupt-driven, mouse-controlled calculator.

Still looking for more C-128 programs? Look no further—here's **Indexer**. This specialized 40-Column mode database for the C-128 allows you to index magazine articles. Since a s-l-o-w program of this nature wouldn't be of much use, Indexer is written entirely in machine language, so it categorizes just as quickly as you can type and saves your entries to disk for rapid recall whenever needed.

The final bonus program, **128 Escape Power Keys**, answers the call for more C-128 utilities. If you program in Basic 7.0, you'll be certain to appreciate the nine new escape key functions this program adds to your C-128.

There you have it—a no-nonsense compilation of powerful utilities and applications for your Commodore. Stay tuned for more in the May/June issue of ReRUN.



Technical Manager  
RUN Magazine

# Directory

PAGE	DOCUMENTATION	DISK FILENAME	FILE TYPE
		* MENU 128	BASIC
		MENU 64	BASIC
1	BOOSTING BASIC	SEARCH-REPLCE	BASIC
		HELP SCREEN	BASIC
		SEQ SCANNER	BASIC
		DIR DISPLAY	BASIC
		FREEZE KEY	BASIC
6	SMART SHOPPER	DOUBLE DISPLAY	BASIC
		SMART SHOPPER	BASIC
		GROCERY	SEQ
		SHOPDATA	SEQ
7	SUPER CHARACTER EDITOR	BOOT OVERKILL	BASIC
		OVERKILL.GEN	BASIC
		OVERKILL.ML	ML
13	PITCH MEMORIZER	* PITCH 128	BASIC
14	£ ADE 64	ADE 64	ML
19	£ POP-UP CALCULATOR	CALCULATOR	BASIC
		BUTTONS.O	ML
		CALCULATOR.BTN	ML
		MOUSE.POINTER	ML
		CALCPOP.O	ML
		CALCULATOR.O	ML
		CALCULATOR.TEXT	ML
21	£ 128 ESCAPE KEYS	* ESCAPE KEY 128	BASIC
		* ESC KEY.ML	ML
25	£ INDEXER	* INDEXER	ML
		* INDEXER/GEN	BASIC
		* INDX.RUN	ML
		* CUSTOMIZER	ML
		* INDEXER.ML	ML

\* — C-128 mode only

£ — Bonus program





# How to Load

## LOADING FROM THE MENU

To get started, C-64 users should type LOAD "MENU 64",8 and press the Return key. When you get the Ready prompt, the menu is loaded and you should type RUN to see a list of the programs on your disk. C-128 users need only press the Shift and Run/Stop keys. When all the programs are displayed on the screen, you can run the one you select by pressing a single key.

## LOADING FROM THE KEYBOARD

If you do not wish to use the menu program, follow these instructions.

**C-64:** To load a C-64 program written in Basic, type: LOAD "DISK FILENAME",8 and then press the Return key. The drive will whirl while the screen prints LOADING and then READY, with a flashing cursor beneath. Type RUN and press the Return key. The program will then start running. To load a C-64 program written in machine language (ML), type: LOAD "DISK FILENAME",8,1

**C-128:** All C-64 programs can be run on the C-128 as long as your computer is in C-64 mode. All C-128 programs are clearly labeled on the directory page. Your C-128 *must* be in C-128 mode to run these programs. To load a C-128 mode program, press the F2 key, type the disk filename and then press the Return key. When the program has loaded, type RUN.

## MAKING COPIES OF ReRUN FILES

Many programs on your ReRUN disk have routines that require a separate disk onto which the program writes or saves subfiles. To use these programs, you must first make a copy of the original program onto another disk that has enough free space on it to hold these newly written subfiles.

It's simple to make a copy of a Basic program. Just load it into your computer as outlined above, and then save the program back onto a separate disk that has plenty of free space for extra files.

Copying an ML program is not so simple. You cannot simply load and save an ML program; you'll need to use a disk-backup utility program, such as the one on your Commodore Test Demo disk.

# Boosting Basic

By Stephen Dirschauer

BASIC 2.0 IS A GOOD programming language, easy to use and adequate for most tasks. However, when you need extra speed and power, it's great to be able to call up the heavy artillery—machine language.

Even if you don't know how to program in machine language (and don't care to learn), read on. This article provides six short, painless routines that can greatly enhance your Basic programs. They will, among other things, allow you to display 16 sprites, and add an automatic help screen and a pause/continue key to Basic.

## 1. MEMORY MASTERY

Memory Search & Replace, when activated, searches through memory to find occurrences of a given byte and replaces them with another. The syntax is simple:

`SYS 50555,start address,end of address+1, search byte,Replacement byte`

For instance, if you wanted to search from location 49152 to 50000, replacing all appearances of 0 with 255, you would enter `SYS 50555,49152,50001,0,255`. (Note that the second parameter is the last address plus one. So you use 50001 and not 50000.)

This routine has special uses, but its real strength lies in its power to do character animation. By searching through screen memory (where your C-64 keeps track of its text display) from 1024 to 2023, you can exchange groups of characters at great speed. If you've redefined the default character set, you can achieve impressive results. Just remember to search for the characters' screen codes, and not their PETSCII codes. Check your C-64 owner's manual for a list of both.

Search and Replace is a Basic loader that pokes Memory Search & Replace into memory. You can include it in your own programs as a sub-routine.

## 2. HELP AVAILABLE

Many commercial programs tout the instant help screens they make available to perplexed users. Help screens are a fantastic idea, but dif-

ficult to implement. If you want the help screen to be available all the time, it's necessary for the computer to constantly check the keyboard (never mind the hassle of putting up the screen and restoring the old display), thereby slowing down your program. However, there is an easier way, using Help Screen.

Through the interrupt request (IRQ) vector, Help Screen provides a key-activated help screen that's always available. Once the machine language part has been loaded and activated, you don't have to do a thing. What's more, installing it takes only two steps.

First, create your help screen using Print statements. Help Screen recognizes the help screen's default starting address of 12288 by using POKE 648,48. Any and all characters are available for the help screen, however color is not. When you've finished creating your help screen, restore the Print statement with POKE 648,4 and a Print command. You shouldn't see anything happen, because while you created the help screen, the old display was preserved.

The second step is to activate the machine language routine with SYS 50618. Press the F7 key at any time to view the screen and any other key to exit.

I use Help Screen during programming to keep a copy of the current disk's directory handy. To use it, switch the Print command (POKE 648,48), get a list of files on the disk and reset the Print command (POKE 648,4: PRINT). This can save time when checking for filenames.

### 3. SEQUENTIAL SCANNER

Reading text files from disk in Basic is possible, but it's also painfully slow. So slow, in fact, that you'll probably want to do it in machine language. The program SEQ SCANNER not only reads the files, but waits for a key press after every page of information.

Before you can activate SEQ SCANNER, you must open the file using channel 1. For example:

```
OPEN 1,8,2,"0:PROGRAM.DOCS,M,R"
```

The M after the filename tells Commodore DOS to ignore the filetype and open the file, whether it's program, sequential or user. Now activate the machine language reader with SYS50682,{number of lines}.

The lines parameter is the vertical size of the screen. Most of the time you'll probably use the default screen size of 25 lines, but anything from 0 to 255 is acceptable. An interesting value is 1, which makes you stop after every line when reading. (A line of text is considered a group of characters followed by a carriage return.) You can then press a repeat-

ing key (such as the space bar) to keep the display going. Pressing run/stop at any time aborts SEQ SCANNER.

#### 4. DIRECTORY DISPLAY

You can also display a disk directory in Basic, but once again, it's too sluggish. Still, if your program asks for filenames at all, a directory listing is a must. DIR DISPLAY prints a list of centered filenames from disk, discarding the seldom-used (from within a program, at least) file size and type. As with SEQ SCANNER, open a file to read the directory. You can then SYS 50764 to read the file. Finally, don't forget to close the file when you're done. For example:

```
OPEN1,8,0,"$":SYS50764:CLOSE1
```

You can use Commodore DOS wildcards for selective lists of files, as shown in Table 1. Pressing the Run/Stop key will halt the display. To pause the directory, just hit any other key. Hitting an additional key will make it continue.

#### 5. FREEZE KEY

The Display Directory routine pauses with the press of a key, but what about the rest of your program? If you need to print a great deal of information at one time, an option to pause output is essential. However, as with the help screen, it's impractical to check the keyboard at every chance. Freeze Key uses the IRQ vector to add an automatic pause key to your C-64. Again, once it's installed and activated, you don't have to do a thing. SYS 50960 will make the routine work behind the scenes. When the Control and P keys are pressed, everything will freeze and the border color will change. Press another key to continue.

#### 6. SEEING THINGS?

Sprites are undoubtedly the best feature of the VIC-II graphics chip. The only problem is the limit of eight. However, Double Display increases the number of available sprites from eight to 16. Just call up

**Table 1. Using wildcards to display files with Display Directory.**

Statement	Function
OPEN1,8,0,"\$*=S"	Lists only sequential files
OPEN1,8,0,"\$DATA.*"	Lists only files with a "DATA." prefix
OPEN1,8,0,"\$A*=P"	Lists program files beginning with A





the Double Display routine at the beginning of your program with SYS 51018,split point.

Though Double Display does give you 16 sprites, each with individual characteristics, there's a catch: The screen is split in two, with eight of the sprites in the first portion and eight in the second. It may be helpful to think of the two halves as different screens. Fortunately, you decide where to split the screen. Set the split point to anything from 0 to 255. For a split point in the middle of the screen, use 150 (the first visible screen line is number 49).

You'll have to use new sprite registers, since the old ones can only handle eight sprites. Just keep in mind the limit: Sprites can't escape their respective subscreens. In fact, they should stay at least ten pixels away from the borderline separating the two.

See Table 2 for more detailed information. All Double Display registers emulate the old VIC-II sprite registers, so the same values must be supplied for twin sprites. For example, to turn on sprite 1 and position it at 65,70, enter:

POKE 51273,1:POKE 51248,65:POKE 51249,70

To turn on sprite 10 and place it at 175,100, enter:

POKE 51313,2:POKE 51290,175:POKE 51291,100

**Table 2. New sprite registers for use with Double Display.**

Screen 1	Screen 2	VIC-II	Description
51248-51263	51288-51303	53248-53263	X/Y locations
51264-51271	51304-51311	2040-2047	Image pointers
51272	51312	53264	Last bit of X position
51273	51313	53269	Sprites enabled
51274	51314	53271	Vertical expansion
51275	51315	53275	Priority register
51276	51316	53276	Multicolor or hi-res
51277	51317	53277	Horizontal expansion
51278	51318	53285	Multicolor register 0
51279	51319	53286	Multicolor register 1
51280-51287	51320-51327	53287-53294	Sprite color registers

Here's a tip: Set variable V1 to 51248 and variable V2 to 51288. You can then refer to registers as V1+X and V2+X. Not only does this save on typing, but it's easier to remember. In the last example, you would need to type only:

```
POKE V2+25,2:POKE V2+2,175:POKE V2+3,100
```

Although the VIC-II chip can only handle eight sprites at a time, Double Display gets around this limitation by using the eight sprites twice. To understand how, consider the way a television set or computer monitor displays a picture. Sixty times a second (50 on European PAL systems), an electron beam, called a raster, redraws the screen display. The beam does this by drawing one pixel-high horizontal line after another as it scans down the screen.

Only 200 of these scan lines are visible, the first one being number 50. (You'll find that these are the same lines used to position sprites vertically.) Memory location 53266 in the C-64 keeps track of the current line being drawn. Basic is much too slow to take advantage of this, but Double Display, written in machine language, can easily use it.

When the top portion of the screen is being drawn, Double Display takes the first eight sprites' registers and copies them to the VIC-II's own memory, then waits until the first portion of the screen is finished. Next, it copies the following eight sprites' registers into VIC-II memory and waits for the electron beam to return to the first line and start again. The gullible VIC-II chip is fooled into displaying each of its sprites twice; the ninth sprite is really the first, and so on.

## PARTING WORDS

Two final points: First, three of the six routines (Help Screen, Freeze Key and Double Display) use the C-64's IRQ vector. They are compatible with each other, so you can use all three at the same time. On the other hand, you may encounter trouble using them with other programs that hog the IRQ vector. If this happens, install the other routines first, and then install any of the three routines just mentioned. These routines share the vector with the program preceding them, thus this should eliminate any conflict.

Second, if you're a budding machine language programmer, you may be interested in disassembling the code for any of the routines. Looking at short, working code is a great way to improve your skills. While doing this, keep a good reference book handy to save time; anything with a comprehensive memory map of the C-64 will do nicely.

# Smart Shopper

By Alan Bradford

HAVE YOU EVER walked into a supermarket with a shopping list, pushed your cart up and down the aisles selecting cans and boxes, and then waited half-way through the check-out line . . . only to realize you hadn't picked up the stuffed artichokes? They were right there on the list, two items from the bottom, but you missed them because stuffed artichokes are in the *first* aisle, not the last. Use Super Aisle Organizer and it will never happen again!

Super Aisle Organizer is a specialized database program that lets you browse through a list of items you usually buy on your weekly safari to the supermarket and select the ones you want on the current trip, then print out a list of those items *organized by aisle*. You won't forget anything, and you won't have to retrace steps for things you missed.

The program is written entirely in Basic and uses two sequential files for the item, aisle and store data. You can include up to 500 items and five stores (the same items for all stores) in your master list.

After loading the program using Menu 64 or 128, it's a good idea to save all of your changes to a formatted work disk for future reference. The program's Main menu offers the following three options, plus Quit:

The Shopping List option lets you select a store and items to buy, then print out a shopping list. When you choose a store, the master list of items will appear on the screen. Press the space bar to move the cursor down to the next item, press C to go to the second column on the page, and press N to go to the next page. To select an item, press the Return key, then respond to the prompt for quantity by pressing return again for one, or by typing a number and pressing return for more.

You can print out a list at any time. It will show the store name at the top, then the items under their appropriate aisles.

The Edit List option lets you add items and stores to your master list and make any needed changes. It will also make a printout of an incomplete master list that you can take to the store with you so that you can fill in aisle numbers.

To add an item to the master list, type its name and aisle number

(15 characters maximum) for each store. If you don't know the aisle at a certain store, enter 0. If an item isn't available at one of the stores, enter 99 for the aisle. Number 98 is for the aisle along the back wall of a store. When you're done, press £ to return to the Edit menu.

To alter item and store names or aisle numbers, choose the appropriate Edit menu option and follow the prompts.

When you're ready to print out your master list, select the Edit menu's print option. The items will appear in alphabetical order, along with aisle numbers for all stores.

After you're done making additions or changes, return to the Main menu and the new data will be saved to disk automatically.

The Main menu's New Disk Setup option creates two data files, Grocery and Shopdata, on a freshly formatted disk. It's a good idea to save a copy of the program on this disk, too. To make backup copies of the data files, open the Edit Items menu, swap to the data disk, and exit the menu: the data will be written to the new disk. Finally, save a copy of the program to the data disk as well.

Now, thanks to your trusty Commodore computer, you'll never go without stuffed artichokes again!

---

RUN it right: C-64

# Super Character Editor

*By Rocco Saya*

WITH OVER 30 FUNCTIONS, Overkill is a character editor with a professional look and feel that experienced programmers will find useful. Overkill simplifies the process of redefining characters that can be used in any of the C-64's graphic modes. Because there are over 30 functions in the program, but no menus, learning to use it proficiently will take time. However, I expect that you'll soon come to enjoy it as much as I do.

To use Overkill, load it using Menu 64. Be sure to have a formatted work disk handy to save your edited character sets on.



## GETTING STARTED

When the split screen appears, you'll notice that the top window is for actual editing. Here, all important registers are displayed, along with an eight-by-eight pixel area where you can redefine characters.

The bottom window displays an entire character set. You'll also notice right away that the bottom window is full of junk. Don't worry—it's just static memory. I kept the junk out of the top window by using a raster routine that displays separate character graphics pages at the top and the bottom. Any work you do on your page at the bottom won't affect the display at the top. In this way, you'll never be working without being able to see what you're doing.

Overkill will let you edit characters in either multicolor or hi-res format. In addition, it provides four separate character graphics pages for you to work on.

To minimize confusion, the first time you use Overkill press F5 and answer Y to the prompt. The bottom window will contain the normal Commodore character graphics. Now that things look a little better, notice that your cursor is in the bottom window around the letter a. Use the cursor keys to move anywhere in the bottom window to pick a character. There are two other cursors in the bottom that I'll explain later.

To exit Overkill, press the Run/Stop and Restore keys simultaneously.

## EDITING COMMANDS

For Overkill to function properly, some commands are available only in the top or bottom window. If there's no other indication, a command is available in both windows.

**Choose a character to edit.** Place your cursor on the desired character and press F1. The bottom cursor will disappear and another cursor will appear inside the eight-by-eight pixel editing area in the top window. Now you can use the cursor keys to move this cursor around. To get back to the bottom window, press F1 again.

**Hi-res editing.** Press the space bar to toggle a single pixel on or off. Top window only.

There are four **multicolor editing commands**. These are available in the top window only. They are:

**Plot bit pattern 01.** Press 1 and Multi 1 is displayed at the top-right. This is also the Background Color 1 register (53282, or \$D022) used in multicolor mode.

**Plot bit pattern 10.** Press 2 and Multi 2 appears at the top-right. This

is also the Background Color 2 register (53283, or \$D023) used in multicolor mode.

**Plot bit pattern 11.** Press 3 to display color RAM at the top-right. Multicolor characters use the actual screen color of a character for this bit pattern.

**Plot bit pattern 00.** Press 0 and the background color is displayed at the top-right. It just erases a pixel using the actual Background Color 0 from register 53281, or \$D021.

Note that in Multicolor mode you lose half the horizontal resolution available on the C-64, because the C-64 needs extra information to be able to display multicolor characters. Therefore, in Multicolor mode you'll plot two pixels at a time, horizontally.

**Switching modes.** To toggle between Hi-Res or Multicolor mode, press M. This affects the bottom window only. When incorporating a character graphics page into your own program, you must use register 53270, which is a control register. To make your character graphics page multicolor, use POKE 53270, PEEK(53270) OR 16. To make it hi-res again, use POKE 53270, PEEK(53270) AND 239.

These commands are for **changing the four color registers:**

**Change multicolor mode 1.** Press F2.

**Change multicolor mode 2.** Press F4.

**Change color RAM.** Press F6.

**Change background.** Press F8.

Pressing these keys allows you to cycle through the available colors. Keep pressing the appropriate key until you get the color you want.

Be aware that when you're in Multicolor mode, color RAM must stay at eight or below to be displayed properly. If it's above eight and you switch to Multicolor, the color RAM will automatically change to eight less than its current value.

Also note that the value of color RAM in the bottom window is actually eight more than indicated. For a character to be displayed in multicolor, its screen color must be higher than seven. If it's not, the character will be displayed in hi-res. In this way, the C-64 can display multicolor and hi-res characters simultaneously in Multicolor mode.

Just remember that when you want to display a multicolor character, you must make sure its screen color is between eight and 15. Also, even though its screen color is above eight, its color RAM nibbles will be shown as a value of eight less than the screen color.

**Erase a character.** Press clear/home to erase the current character.

**Rotate 90 degrees.** Press > to rotate a character 90 degrees clockwise. Note that this causes strange results to occur in Multicolor mode.

**Mirror horizontally.** Press shift/R to produce a mirror image of a character in a horizontal perspective.

**Mirror vertically.** Press F to produce a mirror image of a character in a vertical perspective.

**Reverse video.** Press R to create a negative image of the character. This results in some blurring in Multicolor mode.

**Copy character into buffer.** Pressing I puts the current character into the character buffer.

**Replace character with buffer.** Pressing O replaces the current character with the contents of the character buffer, copying a character.

**Replace character with ROM image.** Pressing F3 replaces the current character with the default ROM image.

**Show data.** Press D to display the Data statements that define the character. Bottom window only.

**Scroll character.** Press the Back Arrow (←) then the Cursor keys to move the current character bit by bit in any direction. When in Multicolor mode, scroll left or right twice to achieve proper results. When you scroll a character, all pixels will wrap around. To toggle in and out of this scroll mode, press ←.

**Change page.** Press P. Overkill lets you work on up to four separate pages. Here is a list of them, their memory locations and the appropriate pokes for the VIC chip memory control register, which resides at memory location 53272.

Addresses	Pokes	Addresses	Pokes
8192-10239	POKE 53272,24	12288-14335	POKE 53272,28
10240-12287	POKE 53272,26	14436-16383	POKE 53272,30

For your own program to display your character graphics page, you poke a value into location 53272, for the page involved. To get the default character graphics back, POKE 53272,21. Remember that these pages are independent of one another, and normally you can use only one at a time. I say "normally," because with Overkill it is possible to have more than one character graphics page on the screen at once. However, it's beyond the scope of this article to explain the raster interrupts required. I'd recommend the book *Mapping the Commodore 64 & 64C* by Sheldon Leemon, a *Compute!* books publication. The following four commands are for **manipulating an entire character graphics page**. For your program's safety, a prompt will ask if you're sure you want to issue the command. They are:

**Copy entire page into page buffer.** Press shift/I to put the entire current page into the page buffer.



***Replace entire page with page buffer.*** Pressing shift/O replaces the entire current page with the contents of the page buffer.

***Replace entire page with character ROM.*** Pressing F5 replaces the entire character page with the default characters.

***Erase entire page.*** Pressing F7 erases the entire character page.

## DISK OPERATION COMMANDS

The rest of Overkill's commands deal with disk operations. If an error occurs while you're accessing your drive, the program reports it.

Before describing the disk commands, I should mention that a page save occurs within a specified range of memory, defined by beginning and ending markers. The default for start-of-range is zero and the default for end-of-range is 255. With the default settings, the Save command will store the entire current character graphics page. However, the first two disk commands will let you alter the range, so you can save any portion of the current character graphics page. After any disk activity is complete, the start and end ranges of a page default back to their original values.

You can view the range in either half of the screen. In the top half, it's represented by the two numbers labeled "Range." In the bottom half, use those two mysterious cursors. Just move the first cursor to the position desired for the start and the second cursor to that desired for the end of range, then press the appropriate key. The **disk commands** are:

***Set start of range.*** Press the + key. Bottom window only. Marks where you want to start saving your characters.

***Set end of range.*** Press the - key. Bottom window only. This marks the stop position for a save.

***Save data.*** Press S. The screen will clear, turn color and request a filename. Make sure a formatted work disk is in the drive, enter a filename and press return. If you try to save your character graphics with a filename that exists, the program will ask if you want to replace that file. If you answer Y, it scratches the old file and saves the new one.

***Load data.*** Press L. The screen will clear, turn light blue and request a filename. Enter the filename and press return. Note: Overkill makes no provision as to the load address—therefore, the file will load into another application at the default memory from which it was saved.

When you are entering the filename for a load or save, you can press run/stop to abort the operation and return to the Main menu.

After you use the Load or Save command, several pointers will default to their original values. They are: color RAM, both start and end of range, mode, and current page. If you loaded data into Overkill using the Load





function, just change pages to find your character graphics.

### FINAL TIPS & REMINDERS

Overkill resides in memory from location 16384 to 20889 and uses locations 28672 to 30720 for variables. Consequently, you could use other machine language utilities at the same time, since these areas of memory are rarely occupied. Be aware, however, that a long Basic program could cut into Overkill, so be careful when running multiple programming utilities.

As a programming trick, you could use a small Basic program with Overkill to test your characters in action. To reactivate Overkill from Basic, just enter SYS 16384.

Remember that you must use control register 53272 to actually use your character set. Also, if you plan to use multicolor characters, you must use register 53270. When using multicolor characters, be sure to use registers 53282 and 53283, poking these with the colors you want. And finally, to load your character graphics data into memory, remember to enter LOAD "NAME",8,1.

---

RUN it right: C-128

## Pitch Memorizer

*By Mark Jordan*

A RECENT VISIT with my family to the Chicago Museum of Science and Industry planted a new programming idea in my head. In the Sound exhibit we had a lot of fun playing with a machine that measured our ability to memorize and then match an audio pitch.

The process was simple. Wearing earphones, we listened to a tone for a few seconds. Then it stopped and we heard a new tone at a different pitch. Our object was to turn a knob that raised and lowered the second pitch until it matched the first. When we had the second tone just the way we remembered the first, we pressed a button to see how close we came. It was fun—especially when we were correct!

I remember playing with this same machine over a decade ago, but

a decade ago I knew nothing about computers, oscillators or the Sound statement, and little about pitch frequencies. After this visit, it struck me how simple it would be to program this sort of pitch-memory device for the C-128.

## THE PROGRAM

The following program, Pitch-Memorizer, is the result. Just load it from disk using Menu 128. Pitch-Memorizer is almost as easy to use as the machine at the museum, although you will control the pitch by pressing keys rather than by twisting a knob.

The number 4 and 5 keys both raise the pitch. The 4 key raises the pitch in large steps while the 5 key moves it in small increments for fine-tuning. Number keys 1 and 2 are similar; the 1 key lowers the pitch in large steps, the 2 key in small increments. This lets you move quickly to the approximate pitch zone with the 1 and 4 keys, then fine-tune with the 5 and 2 keys. When you think your tone matches the first, press return to see how well you remembered.

You will enjoy testing your pitch memory with this program, but that's not the only reason I presented it this month. I also wanted to illustrate the value of keeping your programs small. Pitch-Memorizer, a 32-liner, was neither time-consuming nor difficult. It took less than two hours to write. With the C-128 Sound statement, programming the sounds was a snap, even if a tad crude. CHAR, GET-KEY and DO/LOOP are other C-128 features that saved time and frustration.

I'm a strong believer in short programs. By keeping the program simple, I enjoyed writing it as much as using it. Pitch-Memorizer is short enough that I didn't even need to pre-plan it or add comments for future reference. I just began at the top and wrote it. Done. *AlFin*. What's more, the main structure of the program is complete, so it will be easy to modify. I envision a refined, animated future version with more precise controls, more options and more tests.

# ADE 64

*By Greg Badros*

WITH A GOOD SECTOR EDITOR you can inspect, edit and change every byte on a disk. Unfortunately, standard Commodore disk editors aren't very useful for editing ASCII disks. Because most Commodore disk editors support only Commodore PETSCII, changing the text messages in the true ASCII becomes nearly impossible. ADE 64 solves this and other problems, to make editing GEOS and other ASCII disks a breeze.

ADE 64 is compatible with all printers that accept PETSCII characters and all Commodore disk drives, including the 1581. Pressing U toggles between devices 8 and 9, so you can choose the drive you want.

To use ADE 64, load and run it just like a Basic program using Menu 64. Then place a work disk in the drive, press Return, and enter the track and sector you want to work on.

## BACKGROUND

Commodore disks are divided into numerous tracks, which are, in turn, divided into sectors. The number of tracks per disk and sectors per track varies with the type of drive. Each sector (also called a block) contains 256 bytes of information stored there by the computer. With ADE 64, you can inspect and change each byte of any sector.

The first two bytes of each sector serve a special purpose: They point to the next track and sector in a sequence of blocks that belong together. This makes it possible to have files as long as needed.

For instance, track 18, sector 0 (or 40,0 on a 1581 disk drive) is the directory header. It contains the BAM (block allocation map), the disk name and some other information. The first two bytes of that sector contain track and sector values for the next directory block. That block, in turn, contains the information for the first eight files on the disk, along with, in the first two bytes, another link to the following sector of the directory. The end of a file, or chain of sectors, is signified by a track pointer of 0 in the first byte of the sector.

Sectors in tracks other than 18 usually contain the actual data in a file. You can edit these blocks, also. In fact, with ADE 64 you can edit any-

thing on a disk, including directory entries, Basic programming and even text within a GEOS application. The possibilities are endless.

## THE SCREEN DISPLAY

The editing screen is divided into two main sections and several sub-sections. The top half of the screen contains the block-contents window; the bottom half contains other information you'll need to intelligently edit the current sector. When you press the help (£) key, the bottom of the screen will switch to a list of editing commands. All values are given in both decimal and hexadecimal.

## MOVING THE CURSOR

Once a block has been read in and displayed, a flashing cursor appears in the upper-left corner of the block-contents window. To edit the bytes you want within the sector, move the cursor to the proper position. The four cursor keys behave normally, except that the cursor will wrap right/left and top/bottom.

Several other keys also move the cursor. The space bar acts like the cursor-right key, except it always moves to the next position in the block. For instance, at the right of the block-contents window, cursor-right moves the cursor to the far left of the same line, but the space bar moves it down one line as well. Delete is the opposite of space. It moves the cursor to the left and up one line, if necessary. Both Space and Delete wrap between positions 0 and 255. The Home key works as usual, returning the cursor to the upper-left corner of the block-contents window. The last special cursor control is Shift/Return. This combination returns the cursor to the left end of the current line. Once the cursor moves, all information is updated to reflect the new position.

## THE COMMANDS

ADE 64 includes many commands to simplify the disk editing process. All of the commands that require numerical input accept both decimal and hexadecimal values. Just press return after typing in the number, and be sure to precede hexadecimal numbers with \$. To abort any command, press Return without entering a number.

**F1/F2** Increments/decrements the track number. Note that these commands don't read in the sector (see F5/F6).

**F3/F4** Increments/decrements the sector number. These commands don't read in the sector either (see F7/F8).

**F5/F6** This is the same as F1/F2, but they do read the block after changing the track.



**F7/F8** The same as F3/F4, but they do read the block after changing the sector.

**R** Reads into memory and displays the information from the block at the track and sector currently listed.

**W** Writes the block to the track and sector currently listed. You must use this command to save any changes you've made in the block-contents window before editing a new block.

**U** Toggles the disk drive device number currently in use.

**C** Changes disks. ADE 64 prompts you to insert the new disk, then enter a track and sector to be read and edited (see the S command).

**P** Inputs new text at the current cursor position in the sector. All key presses are stored as Commodore PETSCII values. Pressing return exits this mode.

**T** The same as the P command, except all key presses are stored as true ASCII for use with GEOS programs. Return exits this mode also.

**Q** Quits editing and returns to Basic.

**J** Reads the block pointed to by the track-and-sector link under the cursor and jumps to the next block. For instance, in a GEOS directory entry the two bytes after the filename hold the track and sector of the file header. By positioning your cursor at the first of these bytes and pressing J, ADE 64 will automatically read in the file header.

+ Reads the block pointed to by the next-sector link information at the beginning of the current block.

**S** Changes the sector after you enter a new track and sector. The block is automatically read and displayed. Use the function keys if you don't want the block to replace the current sector in memory.

**O** Sends the currently displayed block to the printer. Press return to start printing the block and any other key to abort. The printout will show a hexadecimal position within the block, followed by 16 consecutive hexadecimal values from the sector and the true ASCII characters they represent.

**N** Reads the next block in the list of sectors accessed. If you haven't used the P command, or are at the end of the list, N works like the + command. See the section below on the list of accessed blocks.

**L** Reads the block that was read just before the current block. Here too, see the section below about the list of accessed blocks.

**D** Sends a command to the disk drive. ADE 64 will send the command, then read and display the error channel. Note that you must send a u0>m1 command to switch a 1571 to double-sided mode and a u0>m0 command to switch back to single-sided mode.

**B** Changes a single byte value. You'll be prompted to enter a value

to store at the current position. After you change the byte, the cursor will advance to the next position.

**A** Changes a two-byte address value. This command works just like **B**, except it changes two bytes and moves the cursor two positions to the right.

**H** Hunts for a string of values. This is ADE 64's most advanced command, and probably its most useful. See the Hunt mode section below.

## THE LIST OF ACCESSED BLOCKS

Two of the above commands, **N** and **L**, control an ordered listing of accessed blocks. Every time a sector is read in, ADE 64 remembers which sector it is and which track it's in. Then, after you've read in another block, you can reread the first by just pressing **L** for "Last block accessed." You can then continue pressing **L** to trace back through the blocks that you previously accessed, or press **N** for "Next block."

It's possible to step back and forth through as many as 45 blocks using the **L** and **N** commands. However, if you step back too far, ADE 64 will report a disk error and you'll have to enter a new block number. Changing disks or toggling the device number clears the list.

## HUNT MODE

ADE 64 has many useful features, but the Hunt command is definitely its most powerful. When you press the **H** key, the bottom message box asks what you want to do. To abort Hunt mode, press Return and the question will disappear. Otherwise, you have two options: hunt for a string, or set the Hunt mode.

To hunt for a string, you must choose the type of string. You may enter a string of ASCII characters (press **A**), a string of PETSCII characters (press **P**), a sequence of bytes (press **#**), or whatever you hunted for last (press **N**, for next occurrence). Once you make a selection, ADE 64 will hunt starting from the current cursor position. If it can't find the desired string in the current block, it will automatically continue looking in the next block. If the hunt is successful, the cursor will appear at the start of the area you sought and a "Found!" message will be displayed. If it's unsuccessful, a "Not Found!" message will appear.

Setting the Hunt mode is also easy. After pressing **H**, press **F** (for file) if you want ADE 64 to follow the track-and-sector links in the first two bytes of each block, or press **D** (for disk) if you'd rather go sequentially through each sector on the disk. The File mode ends when a track-

link of 0 is found; the Disk mode ends at the end of the disk.

### AN EXAMPLE

Let's edit a text message in geoPaint to demonstrate several functions of ADE 64. After running the program, place in your disk drive a 1541-formatted work disk with a copy of geoPaint on it. Then enter 18 and 0 when prompted for the track and sector. You'll see the block type "Dir. Header" appear, along with the screen codes for each of the 256 bytes in track 18, sector 0 of the disk in device 8.

Since we want to change information stored in geoPaint, we must first find its directory entry, which will tell where on the disk the data for geoPaint is stored. Before doing the search, however, we must set the editor to File Hunt mode, so press H and then F. Now press H to hunt, then A for an ASCII string search. When prompted for the string, enter GEOPAINT (notice it's all capital letters).

Since we set the search to File mode, the track-and-sector links will be followed through the entire disk directory. In fact, each successive sector that ADE 64 loads will be displayed on the screen. Before you know it, your cursor will be sitting on the G of GEOPAINT, the ASCII-string display area will show the complete filename, and the message box will tell you the string has been "Found!."

Next we want to go to the place on the disk where the program and text for geoPaint is stored. To do this, move the cursor left twice. Now it's on the two-byte link to the VLIR index block for geoPaint. Press J to jump to that block.

This new block contains a list of two-byte block pointers. Each pair of bytes, starting at position 2, is a track-and-sector pointer to a record in the file. Usually, the first of these records will have any text you might want to change, so move the cursor right to position 2 and press J again, to go to the first block in that record.

Now we're ready to decide which text to change. It's much easier to shorten a message than lengthen it, so, for the sake of this example, we'll change "update" in the File menu to "save." First we have to find where "update" appears in the record. Remember that ADE 64 is still set to File mode, so press H followed by A to hunt for an ASCII string again. This time enter "update" (all lowercase letters) and wait a minute or so as the editor searches for the string. When it's been found, your cursor will appear on the "u" in "update."

Next, press T to enter ASCII text at the current cursor position, type in "save", and press return. Since "save" is two bytes shorter than "update", you'll need to show GEOS that the menu text ends after the "e"



by using the B command with a value of 0. This null signifies the end of the string. After doing all that, press W to write the modified sector back to disk. Now, whenever you use that copy of geoPaint, "save" will appear in the File menu instead of "update." Try it and see!

Remember, you don't need to limit yourself to changing menu choices. Try editing the deskTop messages that appear in dialog boxes, or even the high-score list in your favorite arcade game!

## CLOSING NOTES

ADE 64 is a powerful program, but with power come hazards. Writing a sector with just one wrong byte could erase an entire file (at least until the wrong byte is fixed). You must be very careful when editing a disk. In fact, you should make a backup copy of the disk you want to edit before changing anything.

When editing disks, it's important to know which bytes in which sectors are particularly important. I suggest that you have a copy of *Inside Commodore DOS* handy for reference while you're working. You should also have *The Official GEOS Programmer's Reference Guide* nearby when editing GEOS disks.

---

RUN It Right: C-64; mouse

# The Pop-Up Calculator

By Kevin Smotherman

EVERYONE HAS A HOBBY. Some people climb mountains, others collect things, but me? I just lose stuff—my calculator for example. It always disappears just when I need it most—like while I'm running a program and need to do some calculations. To end my woes, I finally wrote Calculator, a mouse-driven machine language program that puts the power of the C-64's built-in Basic math functions just a SYS away.

Calculator operates in the environment created by Buttons, which appeared in the January/February 1991 issues of *RUN* and *ReRUN*. Buttons lets you program a mouse-driven, object-oriented input system that's very easy to use.

Calculator is completely self-contained, usable in any environment and works in "pop-up" mode. When you start it with SYS 53000, it saves



the current screen, color RAM and status of the Buttons environment, then pops up on the screen. When you're done with your arithmetic, point and click your mouse on the off button, and instantly your video and button environments will be restored.

## PROGRAM OPERATION

To use Calculator, just select the "Calculator" option from Menu 64. The program automatically loads and initializes the Buttons driver, and saves and restores the buttons environment when you exit. That means that the description window parameters at 690–693 and the table vector at 833, 834. You'll note that the last selected Button byte (at 679) and the current table number to activate (at 680) are not maintained, so restore them after a Calculator call.

The power of Calculator comes from Basic itself. Calculator builds a tokenized buffer from your key presses. Then, when you press the = key, it calls the Basic expression evaluator to do the computations. Just use the exact Basic syntax that you would if you were coding the expression and all will go well.

Calculator's display window is limited to 24 visible characters, but you can input up to 128 characters for evaluation. The filled-in circles on either side of the display window let you scroll in either direction, although if you scroll to the left and then attempt to add to your expression, Calculator will automatically move back to the far right-hand end so you can see what you are doing!

## SPECIAL FUNCTIONS

The CE (clear entry) key "undoes" previous key presses—as many as you like, right back to the beginning of the expression. Be careful with the AC (all clear) key, it resets the Calculator (except for the memory).

Calculator contains a single memory, which can hold up to 128 characters, just like the main Calculator display buffer. Whenever you press MS (memory store), the entire contents of the display buffer is saved to memory. Then, subsequent presses of the MR (memory restore) key append this memory to the end of the current display buffer, providing it will all fit. This means that in order for a memory-restore operation to work, the current size of the display buffer plus the size of the data in memory must be less than 128 characters. You can recall the stored data as many times as you like. Clear the memory with, you guessed it, the MC (memory clear) key.

The EE (for scientific, or engineering, notation) key provides a convenient way to enter very large (or very small) numbers into Calcula-

tor. Basic sometimes returns answers in this format, but EE lets you enter them that way, too. The correct syntax is [number1]E[number2], and the operation performed is [number1] \* 10<sup>[number2]</sup>. If [number2] is negative, you create a small number (for example, 1.23E-3=0.00123).

## PROGRAM NOTES

Most of Calculator, including the table definitions for Buttons, resides in the RAM under the Basic ROM. However, three of its routines reside in RAM at 53000, just before Basic ROM. They are the initializer routine, which switches the ROM out and the RAM in; the routine that actually calls the Basic expression evaluator; and the routine that intercepts Basic errors in case you enter your expression wrong. These routines are in the CALCPOP program file, while the rest of the code is in the program file CALCULATOR. The Buttons table definitions are in CALCULATOR.BTN, and the text for the buttons and descriptions are in CALCULATOR.TXT.

---

RUN it Right: C-128

# 128 Escape Power Keys

*By Bill Alaniva*

128 ESCAPE POWER KEYS gives you nine powerful new escape keys that add printing power, screen-swapping and double-wide directories to your programming arsenal. Table 1 summarizes the sequences and their functions.

Why use escape key sequences to implement new commands? Because the escape keys are easy to use. In Immediate mode, just press and release the escape key, then press the desired sequence key, 1-9. In Program mode, add a PRINT CHR\$(27)+"1" or PRINT CHR\$(27)+CHR\$(49) statement to your program to use escape sequence 1. (CHR\$(27) and CHR\$(49) are the character codes for Escape and 1, respectively.)

What's more, using escape key sequences leaves the function keys available for other purposes. However, if there are function keys you aren't using, it's easy to add escape sequences to them. Escape se-

quences are easier to use in Immediate mode if you only have to press one key. To redefine a function key, use Basic 7.0's Key command. Here's an example: KEY 1, CHR\$(27)+CHR\$(49).

## ENABLING THE PROGRAM

After activating 128 Escape Power Keys using Menu 128, the program's Basic loader places its machine language in the default location 4864. Before running the loader, decide if that location is compatible with any other machine language programs you'll be running. If you want to move the machine language, change variable AD in line 10 to any number that is less than 10,000 and divisible by 256. It must be less than 10,000 for the directory commands to work properly. Locations between 4864 and 7168 are recommended, because they don't require moving the start of Basic program space. The machine language must be in bank 15 to work properly, and the BANK 15 command in the loader will take care of that. If you aren't good with numbers or don't have a calculator, just assign variable AD to a bogus number (such as 100) and the loader will give you a list of starting addresses to choose from.

Once installed, the machine language code will automatically be saved. Then you can just BLOAD the machine language into the same spot in memory when you want to use the new escape keys. For example, if you chose location 4864 you would use BLOAD"ESC KEY.ML" and SYS4864 to enable the new escape-key sequences.

Following the save, the program will ask if you want an escape-key cheat sheet. Enter Y and you'll be prompted to verify line 480 of the program to make sure it will set up the printer for condensed printing.

**Table 1. The nine escape sequences.**

Escape Sequence	Function Screens	Modes
Escape-1 Window printer	40/80-column	Immediate/Program
Escape-2 List to printer	40/80-column	Immediate/Program
Escape-3 80-column screen	1 80-column	Immediate/Program
Escape-4 80-column screen	2 80-column	Immediate/Program
Escape-5 80-column screen save	80-column	Immediate/Program
Escape-6 40-column screen save	40-column	Immediate/Program
Escape-7 40-column screen retrieve	40-column	Immediate/Program
Escape-8 Directory, drive 8	40/80-column	Immediate/*
Escape-9 Directory, drive 9	40/80-column	Immediate/*

*\* Usable in Program mode with the appropriate SYS command.*



Check your printer or interface manual for the appropriate sequences. If line 480 will work, make sure the printer is ready and type C to continue. You can have the cheat sheet printed out in normal-size print by typing N, to bypass line 480. If you need to modify line 480, type M and follow the prompts.

Once the machine language has been installed, you won't need the Basic loader anymore, so type in NEW and you'll be ready to start programming.

The escape sequences are not disabled by the run-stop/restore key-press. However, if you're using 80-column screen 2 and press Run-Stop/Restore, you'll return to 80-column screen 1. Just press Escape-4 to get back. To disable 128 Escape Power Keys, you must reset the computer. To restore it after resetting, type in SYS4864 or your new starting address.

## ESCAPE KEY FUNCTIONS

**The Window Printer:** The Escape-1 sequence prints the currently defined text screen window. Normally, the entire screen is the window, but you can use Basic 7.0's Window command to redefine the window to whatever portion of the screen you want to print. The window printer works with either the 40- or the 80-column screen. The active screen (that is, the one with the cursor) will be printed.

The Basic loader portion of the program shows how to use the window printer by printing out an escape-key cheat sheet that lists all nine new escape sequences. Cut it out and keep it near your keyboard for quick referencing. You might also want to design a cheat sheet for your function-keys or anything else you have a hard time remembering. The window printer makes it easy.

If you have trouble printing, make sure the printer is online and that its cables are properly connected. If you're still getting poor printouts, you can change the secondary address that the program uses when it opens a channel to the printer. Remove the REM from line 240, and add the new secondary address to the Poke statement. Consult your interface or printer manual for a list of legal secondary addresses. Without any changes, the program uses a secondary address of zero.

If you're used to embedding commands in text with your word processor (for such things as colors, bold, underlining and italics), you'll be happy to know that 128 Escape Keys will let you embed printer commands in text screens. If you put the commands on the screen in the background color, they won't even show. Be sure to consult your printer manual if you've never used embedded commands before.



**Basic Lister:** The Escape-2 sequence lists the Basic program in memory to your printer. It's a handy feature that gets your program on paper without extra keying.

**80-Column Screens 1 and 2:** Escape sequences 3 and 4 let you select between two different 80-column screens. Screen 1 is the normal 80-column screen. Screen 2 is a completely separate screen that uses the free memory in the VDC's exclusive RAM. When you first select screen 2, you'll be confronted with flashing white squares. These are normal and won't appear again until after the computer has been turned off and back on again. Just do a screen clear and you'll be ready to use the new screen. Each screen has its own color memory, so don't be afraid to program in different colors.

**80-Column Screen Saver:** The Escape-5 sequence is a great addition to the two 80-column screens. It copies whichever screen is being used to the other. This is useful for saving a screen before pulling down a menu or help screen, so that you can restore the screen after you are finished with the menu or help screen. You can do just about anything when combining escape sequences 3, 4 and 5!

**40-Column Screen Save & Retrieve:** The Escape-6 and -7 sequences provide a screen save and restore feature in 40-Column mode. Both the text and color memory are saved, so don't hesitate to change colors between saving and restoring. Forty-column text and color is saved to the same free memory as 80-column screen 2. This shouldn't be a problem, because few programs use both a 40-column screen and two 80-column screens.

**Double-wide Directories:** The Escape-8 and -9 sequences make excellent use of the 80-column screen. With a SYS command, they call up a double-wide directory, showing twice as many files, from disk drive 8 or 9, respectively. Use this with the Escape-1 window printer sequence to get a directory printout lickety-split. Use the directory sequences in 40-Column mode to view a normal-width directory.

These are the only escape key sequences that won't work in Program mode. They simply load the keyboard buffer (the memory where the computer records key presses) with the appropriate SYS command to get the directory. You can still get double-wide directories in your programs by using the appropriate SYS command (normally SYS 5120,8 or 9). The 128 Escape Power Keys program can be relocated, which changes the SYS command. To get the new SYS command, either execute a double-wide directory in Immediate mode and note the SYS command that appears above the directory, or add 256 to your new starting address.

# Instant Indexer

*By Mahmood Hasan Merchant*

INDEXER IS A PROGRAM that I wrote in machine language to help me find specific articles from my heap of computer magazine back issues. With the accompanying Basic program, Customizer, you also can adapt it to a variety of other database applications.

Before customizing Indexer, you should understand the basic structure of a database program. For an introduction to databases, see "File It," by Ellen Rule in the 1991 March/April *RUN*.

I designed Indexer with seven fields: article name, author, issue, page, computer, category and section. You may have noticed that I didn't include a field for the name of the magazine. That's because I have many back issues, but they are of very few magazines, so I prefer to devote an entire file to each. If you'd like to have a magazine field, by all means use Customizer to create it.

The Category field can be very useful. When planning your first Indexer file, make a list of the letters you want to assign to each type of article, then stick to them throughout that file and others. Some of the letters that I use are U for utility, G for game, A for general article, E for educational, F for feature, R for review, and so on.

## THE COMMANDS

To load and activate Indexer, simply select it using Menu 128. The screen will show the number of records free and used, plus a menu of commands. Choose a command by pressing the first letter (L for load, etc.). All commands can be aborted by pressing the Escape key.

A sample file named "RUN" is included on this disk for demonstration purposes.

**Load:** Loads a previously saved database file. Enter a maximum ten-character filename when prompted. Pattern matching is also allowed. Since all data is saved in program files, the load is very fast on the 1571 drive.

If another file is already in memory, a warning appears stating that

it will be erased if you load another. To save the file already in memory, abort the load command by pressing the Escape key, then save as instructed below.

**Find:** Searches the database in three different ways, selected from a submenu.

Press A from the submenu to find all of the records with specific data in a given field. Then select the field number (1-7) and the data that you are looking for. For example, if you wanted to find every article with the word "disk" in its title, you would choose field one and type "Disk." Indexer would then search all of the records, pulling up those with "disk" in the title. The gray right and left arrow keys move you through the list of records.

Press B from the submenu if you want to search using two pieces of data. For example, if you wanted to find all utilities for the C-128, select fields 5 and 6, which call for computer type and category, respectively. Selecting the And option will bring up all of the files that have C-128 in field 5 and Utilities in field 6, whereas the Or option brings up records that have either C-128 in field 5 or Utilities in field 6. In another example, if you wanted to find articles for the C-64 or the C-128, you would choose field five twice, and enter both computer names.

**Add:** With this command you can either add data to a file in memory or create a new file. Enter the data for the fields, pressing return each time.

To add to an existing file, load it into memory as described above, then select add. Create a new file by selecting Add without loading an existing file into memory.

**Correct:** Enter the record number that needs to be corrected. Make the changes to each field, pressing return after each. When the changes have all been made, press escape to return to the Main menu.

**Erase:** Enter the record number to be erased. Remaining records are adjusted automatically to fill the gap.

**Insert:** If you omit an article while entering data, use this option to insert it in the proper place. Just enter the record number when prompted. All records from that number to the end will move one record-space right, then you'll be asked to type the data into the fields.

**View:** Choose this command to view the records in a file. Enter a number at the prompt and that record will appear. Then use the gray left and right arrow keys to move back and forth in the list.

**Save:** Pressing S displays the message "SAVING filename." Press return to go ahead with the save or escape to abort.

Please note that if you save a new file under a name that has already been used, Indexer will automatically scratch the old file before saving the new, so be sure to use a unique name for each new file. Also note that the program automatically appends the prefix INDX. to all filenames before saving, to differentiate them from other files on the disk.

**Directory:** Lists the disk status, and the Indexer data files on disk.

## CUSTOMIZING

Customizer can be used to modify Indexer for other needs. It prompts you to enter the filename Indexer.ML, loads the program into memory, then offers the following options:

**Directory:** All files or only Indexer data files—those with the prefix INDX.

**Number of Fields:** Minimum of two, maximum of eight.

**Names of Fields:** Maximum of eight characters each.

**Length of Fields:** Maximum of 30 characters each.

After you're satisfied with the changes, enter a filename for saving the modified Indexer.

Before you start entering real data, though, spend some time planning the format of your records and testing it. If, after typing a hundred records, you felt you needed another field, maybe for the section of the magazine where an article appears, you'd have to retype all that data into the newly formatted file. ■



# **RE** **== RUN**

EDITOR-IN-CHIEF

**DENNIS BRISSON**

TECHNICAL MANAGER

**TIM WALSH**

MANAGING EDITOR

**VINOY LAUGHNER**

PROOFREADER

**JANICE GREAVES**

COPY EDITOR

**PEG LE PAGE**

ART DIRECTOR

**HOWARD G. HAPP**

DESIGN AND LAYOUT

**ANN DILLON**

TYPESETTING

**DEBRA DAVIES**

FULFILLMENT CONSULTANT

**DEBBIE BOURGALT**



## 8 Programs on this Disk

### *From the March/April RUN:*

- Boosting Basic
- Smart Shopper
- Super Character Editor
- Pitch Memorizer

### *Plus: Extra Bonus Programs!*

- ADE 64
- Pop-Up Calculator
- 128 Power Escape Keys
- Indexer

If any manufacturing defect becomes apparent, the defective disk will be replaced free of charge if returned by prepaid mail within 30 days of purchase. Send it, with a letter specifying the defect, to:

**ReRUN • 80 Elm Street • Peterborough, NH 03458**

Replacements will not be made if the disk has been altered, repaired or misused through negligence, or if it shows signs of excessive wear or is damaged by equipment.

ReRUN contains programs taken directly from listings prepared to accompany articles in *RUN* magazine, as well as bonus programs. These programs will not run under all system configurations. As your guide, use the RUN It Right information included with each article.

The entire contents are copyrighted 1991 by IDG Communications/Peterborough. Unauthorized duplication is a violation of applicable laws.

© Copyright 1991 IDG Communications/Peterborough



[www.Commodore.ca](http://www.Commodore.ca)

May Not Reprint Without Permission