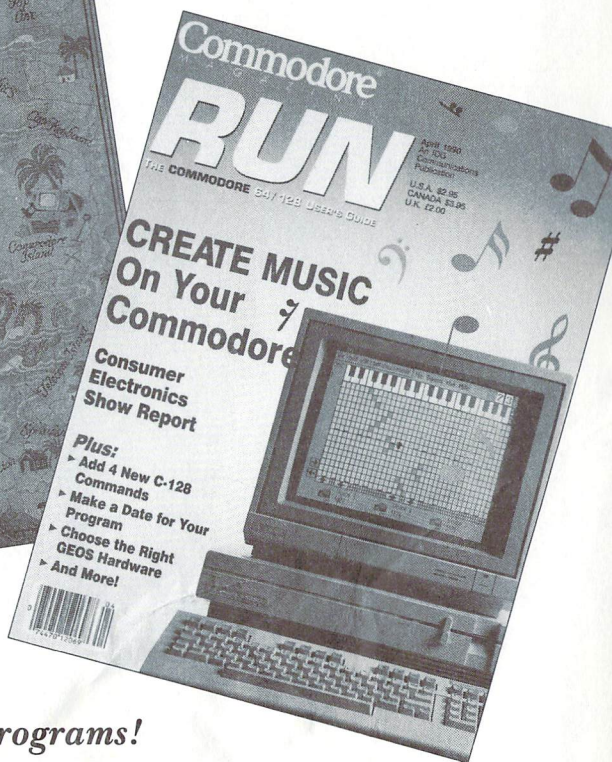
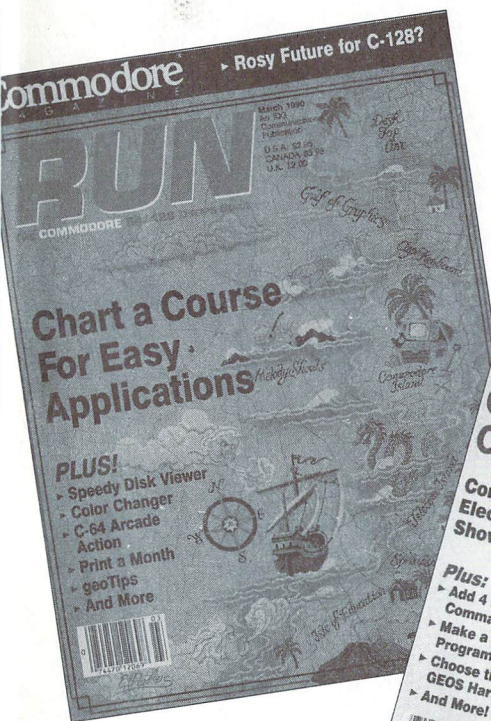


March–April 1990 Edition

# RE RUN

## RUN Programs on Disk

For the C-64 and C-128



**Plus: Extra Bonus Programs!**

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

May Not Reprint Without Permission

# Introduction

*March–April '90 ReRUN*

I RELY ON TELECOMMUNICATION NETWORKS such as QuantumLink and CompuServe for *RUN* reader feedback and for maintaining close ties to fellow Commodore computerists. Many readers on these networks voice concern over Commodore's lack of support for their 8-bit computers. I do my part setting their minds at ease with reassurances that *RUN* and ReRUN play major roles in the continuing development of software and support for any shortcomings on Commodore's part.

This is well-represented with Colorout, the first program in the array of March listings. A marvelous C-64 adaptation of the classic game, Breakout, Colorout uses a joystick in port 2 for moving your paddle at the bottom of the screen to bounce a ball up and knock tiles down, gaining points in the process.

Since we've been chided recently by some readers for printing too many games, you'll be glad to find that the remaining March programs consist of productivity software. With Speedy Viewer, a useful machine language program that allows you to display the contents of program and sequential text files on the C-64's screen, you no longer need to boot a word processor or other lengthy program when you want to view files in a hurry.

Print-a-Month! does just that—creates a hard-copy calendar page, on which you can keep track of a month's worth of activities. March's fourth program, Color Me Quick, is a handy screen utility for the C-64 that permits text, border and screen color changes at the press of a function key. In addition, the program is easily modified so you can press the Commodore key in conjunction with a function key for color changes, thereby freeing the function keys for other uses.

April's programs begin with AFCO, otherwise known as 128 Basic Enhancer. Written by Jim Borden, a former associate editor on *RUN*, AFCO is a programming utility that incorporates useful Add, Change, Find and Old commands into Basic 7.0. C-128 Basic computerists, whether experienced or novice, will find it a valuable aid for programming.

Doing the Alphabet Shuffle is a C-64 variation on the traditional



sliding blocks game. Alien Strike, on the other hand, is a rather untraditional battle, where you are a space-age warrior battling an attacking horde of aliens.

Basic Dater is a handy utility that allows both C-64 and C-128 programmers to date all variations of their Basic programs when saving them to disk. After running either version of this utility, just include a REM statement as the second line in any Basic program you're saving to disk, and Basic Dater will automatically insert the date and time into that line.


A favorite program among most of the *RUN* staff this month is Connex—a C-128 variation on the internationally acclaimed game, Tetris. You have a time limit to position various shaped collections of dots so that they efficiently fill the maximum area on the screen.

Finally, Mark Jordan's 128 Mode for April contains two programs for manipulating C-128 directories. The first places the C-128's disk directory into an array, while the second shows how to use the Directory command to select files from within a program.

As long-time ReRUN users have come to expect, we also include bonus programs. Representing the C-64 side is Grand Prix Challenge, a high-quality, joystick-controlled game for one or two players, where you can race on a variety of tracks.

Finally, there's a C-128 bonus program called Snake Bite. In this bizarre scenario, you must pogo-stick about the screen, collecting scamp-ering mice while avoiding deadly snakes. It's actually a lot of fun.

That wraps things up for this edition of ReRUN. I'll see you again in a couple of months.



*Technical Manager*  
*RUN Magazine*



# Directory

PAGE	DOCUMENTATION	DISK FILENAME	FILE TYPE
		* MENU 128 _____	BASIC
		MENU 64 _____	BASIC
1	COLOROUT	COLOROUT _____	ML
		COLOROUT.HEX _____	BASIC
2	SPEEDY VIEWER	SPEEDVIEWER _____	ML
		SPEEDVIEW.HEX _____	BASIC
3	PRINT-A-MONTH!	CALENDAR _____	BASIC
5	COLOR ME QUICK	COLOR QUICK _____	BASIC
6	128 BASIC ENHANCER	* AFCO _____	ML
		* AFCO.HEX _____	BASIC
10	DOING THE ALPHABET SHUFFLE	ALPHABET _____	BASIC
11	ALIEN STRIKE	ALIEN STRIKE _____	BASIC
		ALIEN SPRITES _____	ML
		ALIEN _____	ML
		SPRITES.HEX _____	BASIC
		ALIEN.HEX _____	BASIC
11	BASIC DATER	* DATER 128 _____	BASIC
		* DATER 128.HEX _____	BASIC
		* DATER128.0 _____	ML
		DATER 64 _____	BASIC
13	CONNEX	* CONNEX _____	BASIC
14	128 MODE	* 128 MODE1 _____	BASIC
		* 128 MODE2 _____	BASIC
17	£ GRAND PRIX CHALLENGE	GRAND PRIX _____	BASIC
		ML CODE _____	ML
18	£ SNAKE BITE	* SNAKE BITE _____	BASIC

\* — C-128 mode only

£ — Bonus program

Before you run a program, read with care the documentation that pertains to it.



# How To Load

## LOADING FROM 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 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 DISKS

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.

---

RUN it right: **C-64; joystick optional**

# Colorout

*By Thomas Czarnecki*

EVEN IF YOUR NAME isn't Joshua, here's a chance to get rid of some frustration by knocking down some walls. Colorout is a fast-paced one-person game for the C-64 that challenges you to knock as many bricks as possible out of a brightly colored, six-course wall.

The missiles you use are ten high-speed balls, kept in play with a movable paddle that's controlled with a joystick in port 2 or the right and left arrows on the keyboard. The wall stretches across the screen, the balls come at it from below, and the paddle moves back and forth along the bottom of the screen.

To serve a ball, press the joystick firebutton or the keyboard space bar. To double the speed of the paddle, hold the firebutton or space bar down. Try putting a little english on the ball or bouncing it off the end of the paddle to send it where you want. If you miss a ball, you lose it off the bottom of the screen. Appropriate sound effects add to the fun of the game.

At first, action is slow, but it speeds up with each new screen of bricks. Also, a ball usually takes out only the first brick it hits on the way up, but if you break through the wall and bounce balls off the "ceiling," they take out every brick they pass coming down.

Each brick you knock out adds five times the number of its course (layer) to your score. In other words, bricks in the bottom course are worth five points, those in the second course 10 points, and so on. Your score so far in the game, your highest score for the session and the number of the current ball are shown at the top of the screen at all times.

A game is over when you've used up all ten balls. To play again, press the restore key.



# Speedy Viewer

*By Howard I. Goldman*

LIKE MOST COMMODORE USERS, you probably have several disks filled with text files that you need to view from time to time. Of course, it's easy to access sequential files from Basic with that quick-and-dirty one-liner:

```
10 OPEN 5,8,5,"0:filename":FOR J=0 TO 1:GET#5,A$:PRINT  
A$;J=ST:NEXT:CLOSE 5
```

But Basic is slow and doesn't word-wrap at the end of lines.

SpeedViewer is a small machine language utility that overcomes these limitations and allows you to examine both sequential and program text files quickly and with the convenience of word-wrap. It's menu-driven and can be loaded and saved like a Basic program.

SpeedViewer's menu provides five options, the first two of which let you choose program or sequential file format, respectively. All word processors save text files in one of these formats. After pressing 1 or 2, enter the name of the file you want, and it will start scrolling on the screen. If you decide not to view a file, press the return key at the filename prompt, and the menu will reappear.

SpeedViewer scrolls files too fast to read, but you can press the space bar to pause it, and again to continue. You can also abort the display and return to the menu by pressing the run-stop key.

Note that some word processors achieve their word-wrapping by padding out lines with extra spaces. If these spaces have been saved to disk with the file, they won't confuse SpeedViewer, but they may result in a strange-looking display.

Option 3 displays a disk directory. As in the viewer modes, pressing the space bar toggles the pause feature, while pressing run-stop brings back the menu.

Option 4 lets you delete a file from the disk. Just enter the filename at the prompt, or, to abort the operation, press return without entering a filename.

Use option 5 to exit SpeedViewer and return to Basic.



# Print a Month!

*By Jared Reynolds*

IF YOUR FAMILY or organization plans activities well in advance, you can save time and frustration in scheduling by using Print-a-Month, a C-64 program that prints out monthly calendars filled in with your own reminders. I wrote the program to help my soccer coach keep track of two teams he was coaching at one time, and now my Scout troop is using it to schedule outings and other events.

Print-a-Month provides space for six entries for every date in the month. By saving your calendars to disk, you can use them as planning guides for the following year's activities.

While the program is complete as is, you may need to customize line 1360 to work with your printer. Variables CPI17\$, FFEED\$ and RESET\$ in that line specify Condensed mode (17 characters per inch), form feed and printer reset (initialize), respectively. Place the appropriate numbers from your printer manual into these variables, using CHR\$ codes.

In addition, you may need to precede some commands with the escape code CHR\$(27) and place your printer interface in Transparent mode. When you've finished customizing Print-a-Month, save it to a formatted work disk.

## CREATING AND EDITING

Print-a-Month works through the menu shown in Table 1. When you select Option 1 to start a new calendar, the program displays a message telling you that if you continue, all of the notes in memory will be erased. If you wish to proceed anyway, enter Y at the prompt; otherwise, enter N, which will return you to the menu.

Assuming that you proceed, the program asks you to enter a month and year separated by a comma, such as 1,1990. The calendar display that appears is headed by the month and year, so you can make sure you entered the numbers correctly. If everything's okay, enter Y to return to the menu.

Option 2 lets you add and edit up to six notes for individual days.





Enter the day you want, then enter up to 18 characters into the first note line for the date, omitting colons, commas and quotes. After you're done, press the return key to move to the next line. Pressing return at the sixth line brings you back to the calendar to choose another date. Pressing return at the calendar without entering a new date brings you back to the menu.

To view your notes without editing them, select Option 3 and press the cursor-right and -left keys to scroll from date to date.

To print out your calendar, select Option 4. The program then checks to make sure you want to continue, and, if you press Y, it proceeds to send the calendar to your printer.

Option 5 lets you load a previously saved calendar so you can update or print it.

Save your calendar to disk with Option 6. The notes currently in memory will be saved in a format that Print-a-Month recognizes, so you can load the calendar by merely entering the month and year. You should be aware that saving a calendar will replace one that may have been previously saved for the same month; so if you want to save a second calendar for any month, you must use a second disk.

Print-a-Month is written completely in Basic, so you can see how it works. Lines 1460–1480 calculate the day of the week on which the selected month begins.

Lines 840–860 and 1210–1230 set up the notes in memory in a format the program can use for printing and saving. The Save and Load options both check the drive for any errors that may occur during data transfer.

**Table 1. The Print-a-Month menu.**

Start a New Calendar  
Add/Edit Notes on Calendar  
View Notes on Calendar  
Print Calendar  
Load Calendar  
Save Calendar  
Quit



# Color Me Quick

*By Steven Gregg*

A FRIEND OF MINE, who programs strictly in Basic, once asked me to write a routine that would give him instantaneous control over the screen, border and text colors of his monitor display. The program had to use only three keys, be totally compatible with Basic and not slow down a running program. As if that weren't enough, it needed to be a short routine. Seems like a pretty tall order? Not really—not for machine language, anyway. That's exactly the type of job machine language does best.

Sixty times per second the C-64 interrupts a running program to check for keypresses, update the system clock, and do other "housekeeping" chores. Screen Color Interrupt (SCI, for short) works by diverting the interrupt routine slightly. If, during the interrupt, the computer detects that F3, F5 or F7 has been pressed, it changes the background, border or text color, respectively, to the next color on the C-64 palette. Press the key again, and the color changes again with nary a hint of hesitation in the program that's running.

SCI is actually a Basic loader that pokes the machine language SCI into memory beginning at 49152 (\$C000). The program then remains active until the computer is turned off. It will deactivate if you press run-stop/restore, but can be reactivated with SYS 49152. The machine language is a scant 131 bytes long, leaving ample room for other programs.

SCI can run as a stand-alone program, but it's really meant to be incorporated into other programs. To use SCI with your own program, include a Gosub to SCI's first line in your code. Then, to return execution to your program, replace the END in SCI's third line with RETURN. The Gosub call to SCI should be located at the beginning of your program, where it will be run only once.

If you need the F3, F5 and F7 keys for other functions in your program, you can alter SCI by pressing another key in conjunction with a function key. Just replace the last 0 in line 60 with the value

you want from Table 1 in SCI's REM statements. For example, if you replaced the 0 with 2, the screen would change color when you pressed F3, F5 or F7 along with the Commodore logo key.

The values in the second REM statement table assign the function keys to SCI use. A value is given for F1, in case you'd like to use F1, F3 and F5 instead of F3, F5 and F7. By all means, leave the REM statements in the program.

You can slip SCI into your own programs for the screen colors you want—when you want them!

---

RUN it right: C-128

# 128 Basic Enhancer

*By Jim Borden*

WRITING COMPUTER PROGRAMS is enough of a challenge without all the typing that's necessary. The AFCO utility cuts down on that typing—and lets you rescue accidentally “Newed” programs—by adding four commands to the C-128's Basic 7.0 editor. With AFCO, you can Add (append) a program or subroutine to the program in memory, Find or Change text in that program or Old (un-New) a program.

Now, here's a description and several examples of each AFCO command.

## **ADD**

The Add command appends a Basic subroutine or program (from drive 8) to the end of the program in memory and adjusts the pointers as required. Using Add requires some planning on your part to ensure proper use of variables in the subroutine you're adding. To that end, I suggest that you keep a record of your subroutines, noting any variables used, the starting and ending line numbers, what data must be passed from the main program to the subroutine and what data will be passed back after the subroutine is finished. The starting line number must be written into any main program to which you add the subroutine.



The syntax for the Add command is simply: ADD "filename". Line numbers aren't important to the Add routine, but be sure to add the *lowest* numbered subroutines first, because new lines are added to the *end* of the program in memory. You can renumber later to clean up any gaps in the line number sequence.

Here's how to use the Add command. With AFKO in memory and this ReRUN disk in the disk drive, load "AFKODEMO" and list it, noting that it contains eight lines. Then type ADD "SUB60000" on an empty line, press return and list the program again to see that the first subroutine has been added to the demo. Finally, add the second subroutine by typing ADD "SUB60100" and pressing return. Note that you added the lowest numbered subroutine first.

You could also have executed the Adds by listing the disk directory and just typing ADD over the number of blocks for the file desired; AFKO ignores anything after the second quote.

Now run the demo to see how the added subroutines work with the original. Don't renumber the demo yet, since later examples will refer to the original line numbers.

## FIND

The Find command lets you locate all occurrences of a string, variable or keyword in a program listing. To use the command, the text you're seeking must be delimited by quotes, periods or colons. Use quotes when you're looking for text within quotes, and periods or colons when you want to find variables or keywords. Here's an example of the latter:

FIND :X4:

You might need to use both quotes and another delimiter (one at a time) to find *all* occurrences. (Printed text is usually in quotes, while remarks usually are not.)

With the demo program and subroutines still in memory, now type

FIND "TO"

and press return. This should list line 30 with the text TO in a different color. Note that the keyword TO (outside quotes) was not found. Typing

FIND :TO:

again should show line 60110.



If more than one match is found in a line, the line will be listed more than once, each time in a different color.

## CHANGE

To alter text in your program listing, use the Change command. The syntax for Change is similar to that for Find, but a second string follows the "Find" string and a third delimiter is required. If quotes are used as the first and third delimiters, a colon should be used in the middle to prevent the second string from being tokenized.

Here are three examples of the Change syntax:

```
CHANGE .PRINT.PRINT#4,.
```

```
CHANGE "TON:TIN"
```

```
CHANGE :ER:EX:
```

The second example won't let the strings be tokenized, because the colon in the middle is accepted as a special delimiter. The third example shows how you might change a variable throughout a program. You might use this, for example, in a case where a program written for the C-64 is being updated to run in 128 mode.

You'll be prompted Y/N/E for Yes/No (change/don't change) or Exit at each match found. If you want all matches changed (are you really sure?), just hold down the Y key.

To avoid finding strings that are parts of other strings, use spaces in the text. For example, if you want to find IS, use " IS" to avoid finding "THIS".

To see a demonstration of Change, enter

```
CHANGE .Y2.Q.
```

When lines 50, 60, 60090 and 60110 are listed, one at a time, press Y after each prompt to change the variable; then, after you're done, list the program to see that the lines actually did change.

Change will turn a short string into a longer one or a long string into a shorter one, and it will adjust all pointers within the program. Note that both strings must be at least one character long.

Always use common sense and caution with the Change command; otherwise you may change more than you intended!

## DISPLAY NOTES

When you're using either Find or Change, each line containing the requested string will be printed to the active output device, usually the screen. To help show the match on the line and avoid

the editing problems caused by using reverse video (as with the 40-column Help command), I've made the string appear in a different color. If the matches are invisible on your screen, change the background or text color and try again.

After a line is listed, the remainder of the screen line is cleared to avoid confusion and aid in editing. The text can be highlighted and edited on the same line with no strange results.

If you're using an 80-column screen, you'll find that matches are always printed in uppercase. This is because I always work in upper/lowercase, so the uppercase makes matches even easier to spot, with no interference with editing.

To output to your printer, open the printer as you normally do and issue a CMD4 to send all output to it—for example,

```
OPEN 4,4,7:CMD4
```

Then press return and type your Find command as usual, and all lines containing matches will be printed out. To keep the program compatible with any printer, use the square brackets ( [ ] ) to enclose the match.

Sending output to the printer is useful mainly when more matches will be found than will fit on the screen.

## OLD

If you accidentally "New" a program, the Old command will restore it for you. To see how Old works, type NEW, press return and list the demo program. To retrieve the program, type OLD on an empty line, press return and list the program again to confirm that it's been restored. The Old command must be used *before* you enter any lines after the New command!

If you "New" a program when AFCO isn't active, just boot AFCO.ML and type OLD to restore your program.

The AFCO commands are real time—and program—savers. I'm sure that once you try them, you'll always want them available to help write and edit your Basic 7.0 programs.

---

RUN it right: **C-64; joystick**

# Doing the Alphabet Shuffle

*By Charles Phoenix*

ALPHABET SHUFFLE IS A strategy game in which you alphabetize a grid of randomly placed letters by swapping them, one at a time, with a blank space. It's based on the little sliding-number puzzles you probably know and offers three skill levels, with eight, 15 and 24 letters. You move the letters with a joystick plugged into port 2. The program is written entirely in Basic.

At the opening screen, you select your choice of skill level with the joystick and press the firebutton to call up the brief on-screen instructions. The program sets up your puzzle, and when it appears, press the firebutton to start play.

You can move only those letters that are adjacent to the blank space. Decide which letter you want to move, then press the joystick in the direction the letter should go, and it will swap places with the blank space. As you play, your elapsed time and the number of moves you've made are continuously shown at the top of the screen.

The game is over when all the letters are in alphabetical order and the blank space is last. After the closing score-and-time display, you can press the firebutton to play again at the same or another level. You can also press the firebutton at any time during a game to quit and bring up another puzzle. Remember that all moves are related, so it's important to think ahead.





---

RUN it right: **C-64; joystick**

# Alien Strike

*By Behzad Jamshidi*

WATCH OUT! WHAT WAS THAT? Find out when you play this one-person action game, designed for the C-64 with a joystick plugged into port 2.

Here's the scenario that unfolds when you press F1 to play. An alien space ship wanders back and forth across the top of the screen, launching squadrons of fighters that try to destroy your only line of defense, a cannon at the bottom of the screen. The fighters come in numerous shapes and sizes, and they travel at various speeds on unpredictable paths. Are you fast enough to shoot them down, along with the mother ship, without getting zapped yourself?

We'll see. Using the joystick, move the cannon back and forth to avoid the attackers and to aim; then press the firebutton to shoot. Hold the firebutton down for rapid fire.

If you destroy the mother ship, a new one unfortunately appears and sends out an additional squadron of fighters. There's a third mother ship, too, and four squadrons altogether.

You get 20 points for each fighter you destroy. Mother ships are worth 120.

---

RUN it right: **C-64 or C-128**

# Basic Dater

*By Ken Huebner*

IF YOU'RE A C-64 or 128 programmer who has trouble keeping track of your Basic file listings, here's a handy utility that automatically signs the filename, date and time to every program you save.



With Basic Dater, you'll no longer have to suffer the aggravation of getting your latest updated files confused with old backup versions.

## **SETTING DATE AND TIME**

When you're ready to use Basic Dater, load and run either the C-64 or C-128 version. With the C-64, expect a short delay while the ML code is poked into RAM. Once your computer is ready, a prompt will appear, asking for the date. Answer by typing, in compressed form, the month, day and year, separated by dashes (e.g., 2-31-1989). The program then asks you, "What's the time?" Type in the time in hours and minutes, plus whether it's AM or PM, without any intervening spaces or special characters. For example, if it's 5:35 PM, type 535PM.

Once the utility has accepted your entered time, the message "Dispatcher On" appears. [Dispatcher was the original program name, generated here by the ML code.—Eds.] If you wish to turn Basic Dater off, enter SYS 52224 for the C-64 or SYS 6144 for the C-128. In case you forget the turn-off instructions, press the run-stop/restore key combination simultaneously.

## **IMPORTANT CONSIDERATIONS**

Always load and run Basic Dater before you begin a programming session. Note that the second program line will now be unavailable for your use, since Basic Dater uses this line to record program information. Just place a REM or colon at line 2 to properly format your program. Whenever you're ready to save your Basic file, simply do it as you normally would.

When you load the file again, you'll see that the second line now contains the filename in parentheses, followed by the date and time that the save occurred. Now, each time you save a file, a little record of the save is made in the Basic listing. There's also no need to worry about sharing a signed program with a friend, since a program signed by Basic Dater is fully compatible with Commodore's Basic, whether or not the Dater program is in memory.

Finally, if you're doing some program debugging, don't be hesitant about pressing the run-stop/restore keys as an escape, for this will not disable either version of Basic Dater.



---

RUN it right: **C-128 (in 40-Column mode); one or two joysticks**

# Connex

*By Leonard Morris*

WANT TO MAKE SOME good connections? Then get to know Connex, a C-128 game that's reminiscent of the currently popular Tetris. The object of Connex is to place randomly generated patterns of five dots on the gameboard so they connect to each other and fill as many board spaces as possible.

Connex can accommodate one or two players, and two can share the same gameboard or have separate boards. Players 1 and 2 use joysticks in ports 1 and 2, respectively.

When the first pattern appears at the top of the screen, move it onto the gameboard so it touches the single dot already there; then, before the timer runs out, press the firebutton to place the pattern. You earn points according to the number of dots in the pattern that touch the single dot. If the pattern doesn't touch the dot, overlaps the dot or hangs partly off the gameboard, or if time runs out, you're charged with an error and lose the pattern and the turn.

When the second pattern appears, connect it to the first, and so forth, always filling as many spaces on the board as possible. Unlike Tetris, you can move a new Connex pattern across the patterns already placed.

If you make four errors, the game ends and you must start over, but if you fill 75 percent of the board, you advance to the next level of play. You can also advance to the next level—and receive an extra 1000 points—by completely filling a row with dots. That row then vanishes and the rows above it move down to take up the space.

Connex has 16 levels, each made more difficult than the last by an increasing number of predator circles that meander around the gameboard. If one of these circles touches a pattern you've moved onto the board but haven't yet placed, you lose the pattern and are charged with an error.

When two are playing, the color of the random pattern indicates whose turn it is. If the pattern is yellow, it's the first player's turn; if it's red, the second player moves. Although there's a time limit



for each turn, I didn't put a timer on the screen, both to heighten suspense and because the action is too fast to watch it, anyway.

As you play, three numbers are always displayed on the screen: your current score, your error status and the percentage of the board you've covered so far.

---

RUN it right: C-128

# 128 Mode

*By Mark Jordan*

AMONG THE MOST POPULAR C-128 commands is Directory, which is so useful that it's even built into one of the function keys (F3). However, I fear many 128ers overlook some of the "directorial" power available to them.

The problem stems partly from the *C-128 System Guide's* cryptic description of the command:

DIRECTORY (Ddrive number)(,<ON>Udevice)(,wild card)

The first two options—Ddrive number and ON—are useless for all practical purposes. Forget both of them.

The Udevice option lets you use Directory on a second (or third or fourth) drive, just by tacking U9, or the like, onto the end of the command, like so:

DIRECTORY U9

Don't put a comma before the U9 unless you're using a search string, as described below.

## SELECTIVE DIRECTORIES

The Directory command provides two ways to limit directory listings to what you want: with search strings and by file type. A search string is a specific string of characters used to identify the filename(s) you want listed. The string could be identical with one filename, say BOB. To list BOB alone, you'd type:

DIRECTORY "BOB"

If you're accessing a second drive, tack on ,U9 (note the preceding comma) after BOB.

Search strings really come into their own when used with the wild cards: the asterisk and the question mark. For instance, most Commodore users know that placing a mnemonic or symbolic prefix on filenames makes them easy to group using the asterisk wild card. You might place WRD. before the names of all word processing files. Then the command DIRECTORY "WRD.\*" would list only word processing files.

This is old hat. But what if you want the mnemonic to be at the end of the filename, like CP/M and MS-DOS extensions? Placing it at the end makes the filenames easier to read. The Directory command can't handle that with the asterisk (except when you're using the 1581). However, with the humble question mark wild card, it can. The key is in always using 16-character filenames, with the final characters constituting the suffix, or extension.

Let's say you're saving a letter to Sue. You could call it SUE\_\_\_\_\_.LTR, where the underline characters represent spaces that pad the filename proper to 12 characters (the added period and 3-letter suffix making 16). With files thus named, you could list them by extension as follows:

```
DIRECTORY "????????????.LTR"
```

You could also save yourself the bother of typing 12 question marks by assigning them to a variable, such as:

```
Q$="?????????????"
```

Directory lets you use variable names for search strings in two ways: inside parentheses, as in:

```
DIRECTORY (Q$)
```

or added to another search string, as in:

```
DIRECTORY (Q$) + ".LTR"
```

The second way you can get selective directory listings is by file type. To do so, you must add the expression =S (or P, R or U) after the search string. Thus, to list just sequential files, you'd type:

```
DIRECTORY "*" =S"
```

Change the S to P, R or U and you'd list program, relative or user files, respectively.



Happily, these two methods of listing selective files can be combined. For example, DIRECTORY "PI\*=S" will list only those sequential files whose names start with the letters PI.

## PRINTING DIRECTORIES

Unfortunately, you can't use the Directory command to print out a hard copy of a directory. You must resort to the old C-64 method of loading the directory into memory (and thus erasing whatever is already there). At least it's simple:

```
LOAD "$",8
```

Then, to print it out, you must open a channel to the printer with:

```
OPEN 4,4,7:CMD 4:LIST:CLOSE 4
```

Wild-card search strings work here, as well. To load just those files with extension .LTR, type:

```
LOAD "$:?????????.LTR",8
```

And file-type matching works here, as well:

```
LOAD "$:*=S",8
```

## RUNNING DIRECTORIES

The tricks I've described so far are all for the Immediate, or Direct, mode of operation. How about when a program is running?

Well, the Directory command does work within a program, but it's limited: It just scrolls the files by. To make it possible to select files from the listing, you must employ a few more tricks.

Directory 1 shows how to get all the files into an array. The key is to use a secondary address of 0 when opening the directory (see line 20). When run, this little program reads a directory and converts all the filenames into Basic subscripted variables—in this case FI\$( ). FI\$(0) holds the name of the disk, and FI\$(1) through FI\$(F) hold all the filenames. This is very useful whenever you write a program that needs to read directories for the user to cycle through.

Directory 2 provides another way. It uses the Directory command itself for selecting files from within a program. With the judicious use of the Window and Input commands, plus a couple of sneaky Pokes, you come out of this routine with FI\$ holding the name of the file that was selected.

Both listings contain remarks that, with a little study, should enable you to use the listings as subroutines in your own programs.

Getting directories on the C-128 is a delight, especially when you know some power moves. List on, O Directory.

---

RUN it right: **C-64; one or two joysticks**

# Grand Prix Challenge

*By Scott Elder*

**START YOUR ENGINES**, ladies and gentlemen. It's time for Grand Prix Challenge, a one- or two-player auto-racing game that's both exciting and easy to play. It offers a choice of nine tracks that vary in length and difficulty, and features a "3-D" split-screen display showing "radar views" of the track. Realistic sound effects add to the fun.

Player 1 controls the blue car, using the bottom screen and a joystick in port 1; player 2 controls the red car on the top screen, with a joystick in port 2. Alone, you're player 2 racing against the automatically controlled red car.

The controls are easy and the same for each car. Move the joystick left and right to steer from side to side, press it forward to accelerate and pull it back to slow down. The firebutton is a turbo that provides faster acceleration than the stick-controlled throttle and lets you reach a higher top speed. The turbo works only when you're not steering left or right.

To play, load and run GP CHALLENGE, and maps of the nine tracks will appear. If you're player 2, who gets to choose the track, use your joystick to move the black square onto the desired track and press the firebutton.

Then the scene shifts to the racetrack, where the two cars are poised, impatiently waiting for the end of the countdown. Both screens are identical at this point, but as the race begins and one car pulls ahead, the screens diverge. For instance, if the red car pulls ahead, the blue car disappears from the top screen and the red car grows smaller in the bottom screen until it disappears in the distance. Then there's only one car in each screen until the blue car catches up or the red car laps the blue.





When you come to a curve, centrifugal force tends to throw your car against the guardrail. There won't be a crash, but the rail will slow you down, so try to keep away from it. Of course, when you're pulling your car toward the middle of the track the turbo won't work, so you can never make great time on a curve.

There are no crashes between cars, either. They may look as though they're touching, but one is always a little ahead of the other.

The first driver to complete two laps wins. And, yes, there can be a tie. If you want to race again, press the firebutton and. . .start your engines!

---

RUN it right: **C-128; joystick**

# Snake Bite

*By Ken Huebner*

YOUR BIOLOGY TEACHER, Mr. Crabtree, has just been called away, and he leaves you in charge of the class. Uh-oh! Some joker in the back of the room has released all the snakes and lab mice from their cages! Your only chance to make good is to hop on a pogostick and gather the mice—while avoiding the deadly snakes—before Mr. Crabtree returns. Can you save the day in this ten-level game?

To start the game, load "SNAKEBITE" from Menu 128. There will be a short delay while the machine language code is being poked into RAM. Note that Snake Bite requires a joystick plugged into port 1, not port 2.

Once the title screen appears, press the joystick button to play. The mood is set with a short rendition of Three Blind Mice, and then a baseball-capped kid bouncing on a pogostick appears, surrounded by up to six scampering white mice and a mess of slithering green snakes. Using your joystick, move the kid toward one of the mice. If you bounce close enough, you may be able to grab the mouse and win 1000 to 10,000 points. But beware! Touch a snake and you'll probably fall dead on the classroom floor.

If even one uncaptured mouse remains when Mr. Crabtree returns, you must try again. You have three "lives" in which to grab

all the mice at one level before the game ends. Try trapping the mice along a wall or in a corner of the room; they're much easier to catch there. Once you gather all the mice, you advance to the next level of difficulty.

During play, your current score and the lives and time you have left are displayed across the top of the screen. The action is accompanied by amusing sound effects.

I designed Snake Bite using a unique bitmap line-drawing algorithm. A snake consists of a series of line segments. When the head is plotted forward, the tail is erased, producing a meandering serpentine effect. The pogostick kid and the mice are all sprites.

Whatever you do, the next time you go to biology class, be sure to have your pogostick handy! ■

# **RE == RUN**

EDITOR-IN-CHIEF  
**DENNIS BRISSON**

TECHNICAL MANAGER  
**TIM WALSH**

MANAGING EDITOR  
**SWAIN PRATT**

SENIOR EDITOR  
**BETH S. JALA**

ASSOCIATE EDITOR  
**HAROLD R. BJORNSEN**

COPY EDITOR  
**PEG LePAGE**

ART DIRECTOR  
**HOWARD G. HAPP**

DESIGN AND LAYOUT  
**ANN DILLON**

TYPESETTING  
**DEBRA DAVIES**  
**KEN SUTCLIFFE**

FULFILLMENT CONSULTANT  
**DEBBIE BOURGAULT**



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

May Not Reprint Without Permission

# 12 Programs Included on this Disk

## *From the March RUN:*

- ▶ Colorout
- ▶ Speedy Viewer
- ▶ Print a Month!
- ▶ Color Me Quick

## *From the April RUN:*

- ▶ 128 Basic Enhancer
- ▶ Doing the Alphabet Shuffle
- ▶ Alien Strike
- ▶ Basic Dater
- ▶ Connex
- ▶ 128 Mode

## *Plus: Extra Bonus Programs!*

- ▶ Grand Prix Challenge (C-64)
- ▶ Snake Bite (C-128)

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:

***RUN Special Products • 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.

The programs in ReRUN are taken directly from listings prepared to accompany articles in *RUN* magazine. They will not run under all system configurations. Use the RUN It Right information included with each article as your guide.

The entire contents are copyrighted 1990 by IDG Communications/Peterborough. Unauthorized duplication is a violation of applicable laws.

©Copyright 1990 IDG Communications/Peterborough

