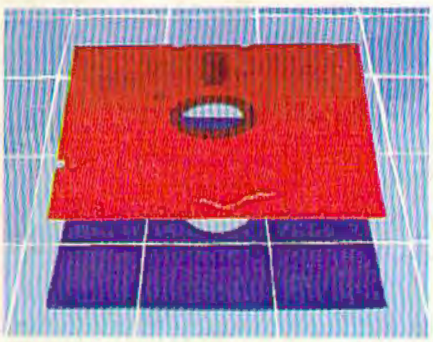
Custom Characters For VIC And 64

COMPUTE!'s GAZETTE

$2.50
November 1983
Issue 5 Vol. 1, No. 5
63380 $3.25 in Canada

For Owners And Users Of **Commodore VIC-20**™ And **64**™ Personal Computers

Getting Started With A Disk Drive



A disk drive is much more than just a fast tape drive. This new series begins where the manual leaves off. Part I: The basics of floppy disks.

Chicken Little

Action Game For VIC-20 And 64

The sky is falling! Only you, as Chicken Little, can save the world by hurling stones at the falling pieces of sky. But if you get bonked on the head....

MERGING PROGRAMS ON THE 64

With this utility you can easily add favorite routines to new programs, saving time and effort. A must for every programmer's toolbox.

MUNCHMATH

Educational Game With Animation And Sound For VIC-20 And 64

With a video game character in hot pursuit, the goal is to solve all the math problems correctly. If you miss one too many, prepare to give up your ghost. Multiple levels for young people of all ages.

Also In This Issue

Software Reviews

This Month's Best Sellers

Machine Language For Beginners

0 74470 63380 11

T&F SOFTWARE

Two Hot New Action Games!

FOR COMMODORE 64

Speed Racer

$29.95

Feel like a devil today? Better adjust your halo. Start the ignition. It's up to you to decide... Okay, hit the accelerator. You can be the ultimate speed demon or put wings on your wheels! There's a pedestrian now... will you go for speed, no matter what pedestrians are in the way (for the devilish points)— or be an angel in a challenging obstacle course? The heavenly points may get you to heaven, but the devilish points may get you to... well, better make your decision— the time clock is running!

Candy Bandit

$29.95

So your sweet tooth has gotten out of hand again? Well, this time the sheriff is after you and he's no sweetheart. There he is now! Quick! Grab all the candy in sight and dive for the doorway! Don't look now, the doors are rotating... better be good at getting out of sticky situations, 'cause if you hit the wall you're stuck with it! There's plenty of candy here and many more levels to cross, so get on the move, sugar!

Available at finer Software Stores everywhere.

OR CALL (213) 501-5845 FOR THE NAME OF YOUR LOCAL DEALER OR DISTRIBUTOR.

T&F SOFTWARE

T&F Software Company 10902 Riverside Drive / North Hollywood, California 91602. (213) **501-5845**

CAN A COMPUTER MAKE YOU CRY?

Right now, no one knows. This is partly because many would consider the very idea frivolous. But it's also because whoever successfully answers this question must first have answered several others.

Why do we cry? Why do we laugh, or love, or smile? What are the touchstones of our emotions?

Until now, the people who asked such questions tended not to be the same people who ran software companies. Instead, they were writers, filmmakers, painters, musicians. They were, in the traditional sense, artists.

We're about to change that tradition. The name of our company is Electronic Arts.

SOFTWARE WORTHY OF THE MINDS THAT USE IT.

We are a new association of electronic artists united by a common goal — to fulfill the enormous potential of the personal computer.

In the short term, this means transcending its present use as a facilitator of unimaginative tasks and a medium for blasting aliens. In the long term, however, we can expect a great deal more.

These are wondrous machines we have created, and in them can be seen a bit of their makers. It is as if we had invested them with the image of our minds. And through them, we are learning more and more about ourselves.

We learn, for instance, that we are more entertained by the involvement of our imaginations than by passive viewing and listening. We learn that we are better taught by experience than by memorization. And we learn that the traditional distinctions — the ones that are made between art and entertainment and education — don't always apply.

TOWARD A LANGUAGE OF DREAMS.

In short, we are finding that the computer can be more than just a processor of data.

It is a communications medium: an interactive tool that can bring people's thoughts and feelings closer together, perhaps closer than ever before. And while fifty years from now, its creation may seem no more important than the advent of motion pictures or television, there is a chance it will mean something more.

Something along the lines of a universal language of ideas and emotions. Something like a smile.

The first publications of Electronic Arts are now available. We suspect you'll be hearing a lot about them. Some of them are games like you've never seen before, that get more out of your computer than other games ever have. Others are harder to categorize — and we like that.

WATCH US.

We're providing a special environment for talented, independent software artists. It's a supportive environment, in which big ideas are given room to grow. And some of America's most respected software artists are beginning to take notice.

We think our current work reflects this very special commitment. And though we are few in number today and apart from the mainstream of the mass software marketplace, we are confident that both time and vision are on our side.

Join us.

We see farther. ELECTRONIC ARTS

FEATURES

GAMES

REVIEWS

EDUCATION/HOME APPLICATIONS

PROGRAMMING

DEPARTMENTS

PROGRAM LISTINGS

* = General, **V** = VIC-20, **64** = Commodore 64.

COMPUTE!'s Gazette is published twelve times each year by **COMPUTE!** Publications, Inc., Post Office Box 5406, Greensboro, NC 27403 USA. Phone (919)275-9809. Editorial offices are located at 505 Edwardia Drive, Greensboro, NC 27409. Domestic subscriptions: 12 issues, $20. Send subscription orders or change of address (P.O. Form 3579) to Circulation Dept., *COMPUTE!'s Gazette*, P.O. Box 5406, Greensboro, NC 27403. Second class application pending at Greensboro, NC 27403 and additional mailing offices. ISSN 0737-3716.

THE EDITOR'S *notes*

Good News for Commodore users... Jim Dionne, former Canadian sales manager for Commodore, has been appointed vice president of sales and marketing for Commodore. In this capacity he'll report directly to vice-chairman and founder Jack Trameil. We applaud this move, since for many years Mr. Dionne has been cognizant of, and responsive to, the needs of users and the marketplace.

Will the IBM home computer knock Commodore out of the low-end personal computer market? We think not, for two main reasons. The IBM entry is projected to arrive with a price tag around $700 for the basic unit. While the price and name will certainly sell it well, Commodore has shown it's quite ahead of the game in successive generation introduction of hardware and features. Even if sales of the 64 were impacted by an IBM entry, we suggest that not many months would pass before a mid-level, full-featured computer would arrive from Commodore with a price tag no one could touch. And we certainly don't mean to imply that the 64 is currently vulnerable; nothing on the market can yet match its capabilities at its current price.

We've been predicting increasing availability of software for the VIC and 64. Word from the field is that it's coming in droves as everyone from major existing vendors to smaller startups move to support the 1.5 million machines currently installed. Judging from a recent article in *Electronic News* (Vol. 29, No. 1459), most of that software will be sold on disk. Commodore indicated that their original projection that 70 percent of 64s would be sold with a disk drive was low. Actually, sales with disk drives are running at 90 percent.

Five issues old and approximately the sixth-largest magazine in the personal computing field. Again our thanks to you, our readers and writers, for your support and encouragement in the very successful launch of the *Gazette*. We're still looking for experienced writers and programmers to join our inhouse staff. If you're interested, drop us a résumé, Attn: Personnel Office, COMPUTE! Publications, Inc., P.O. Box 5406, Greensboro, NC 27403.

A bit of history on the origins of your computers for those of you new to the industry. Commodore introduced the first fully integrated personal computer system in the summer of 1977. It was called the PET (Personal Electronic Transactor). It weighed 30 or 40 pounds, had a bulky metal case, built-in keyboard, green phosphor monitor, and tape cassette. The keyboard had tiny, calculator-type keys. The unit was first demonstrated at the Philadelphia Computer Show, and Commodore was immediately swamped. Personal computing was off and running. Commodore creatively financed much of the launch out of customers' pockets, with demand running so far ahead of production that the company was able to accept customer orders on a direct basis with full payment up front and expectation of a delivery within at least 90 days (frequently in the early days of shipping, that date went to 120 days or more). If you became distressed over lengthy delivery, your only option was to cancel your order, get a refund, and wait even longer. Your VICs and 64s are far more powerful and sophisticated than the $800 PET, but that's where they came from. A whole new generation of computing was launched.

Robert Lock

Editor In Chief

COMPUTE! Publications, Inc.

A Subsidiary Of American Broadcasting Companies, Inc.

Publisher Gary R. Ingersoll
Editor in Chief Robert C. Lock
Publisher's Assistant Alice S. Wolfe
Senior Editor Richard Mansfield
Managing Editor Kathleen E. Martinek
Art/Design Director Georgia Bikas Davis

Editorial
Tom R. Halfhill, Gazette Editor; Orson Scott Card, Editor, COMPUTE! Books Division; Gail Walker, Production Editor; Ottis R. Cowper, Technical Editor; Charles Brannon, Program Editor; Tony Roberts, Assistant Managing Editor

Assistant Editors
Dan Carmichael (Publications), Stephen Levy, Gregg Keizer (Books), Lance Elko (Publications), Kathy Yakal, Editorial Assistant (Features)

Editorial Programmers
Patrick Parrish, Gregg Peele

Technical Assistant
Dale McBane

Copy Editing/Proofreading
Juanita Lewis (Assistant), Becky Hall, Sarah Johnston, Linda Shaw

Administrative Staff
Vicki Jennings, Laura MacFadden, Julia Fleming

Associate Editors
Jim Butterfield (Toronto), Harvey Herman (Greensboro), Fred D'Ignazio (Roanoke), David Thornburg (Los Altos), Bill Wilkinson (Cupertino)

Production
Irma Swain, Assistant Production Manager; De Potter, Mechanical Art Supervisor; Terry Cash, Debi Thomas, Typesetting

Artists
Leslie Jessup, Cindy Mitchell (Publications), Janice Fary, Debbie Bray (Books); Todd Heimarck, Promotions Assistant; Harry Blair, Illustrator

Operations/Customer Service
Carol Lock, Manager; Patty Jones, Customer Coordinator; Assistants: Chris Patty, Chris Gordon; Fran Lyons, Dealer Coordinator; Assistants: Gail Jones, Sharon Minor

Customer Service Staff
Dorothy Bogan, Supervisor; Rhonda Savage, Lisa Flaharty, Anita Roop, Sharon Sebastian, Debi Goforth; Operators: Cassandra Robinson, Mary Sprague
Jim Coward (Warehouse Manager), Larry O'Connor, Chris Cain, Dai Rees, Jack McConnell

Data Processing
Leon Stokes, Manager; Joan Compton, Assistant

Accounting
W. Jerry Day, Manager; James M. Hurst, Comptroller; Assistants: Linda Miller, Doris Hall; Staff: Ruth Granger, Anna Harris, Emilie Covil, Anne Ferguson

Advertising Sales
Andy Meehan, National Sales Manager; Patti Williams, Production Coordinator; Bonnie Valentino, Accounting Coordinator; Rosemarie Davis, Sales Assistant

Sales Representatives

Jerry Thompson	415-348-8222
Phoebe Thompson	408-354-5553
JoAnn Sullivan	619-941-2313
Ed Winchell	213-378-8361
Harry Blair	919-275-9809

Jules E. Thompson, Inc.
National and Canadian Sales Representatives
1290 Howard Avenue, Suite 303
Burlingame, CA 94010

Address all advertising materials to:
Patti Williams, *COMPUTE!'s Gazette*,
505 Edwardia Drive, Greensboro, NC 27409

Sales Offices, The Thompson Company

New England	617-720-1888
Mid-Atlantic	212-772-0933
Southeast	919-275-9809
Midwest	312-726-6047
Texas	713-731-2605
Northwest	408-354-5553
Northern CA	415-348-8222 or 408-354-5553
Southern CA	619-941-2313 or 213-378-8361
Nevada, Arizona	619-941-2313
New Mexico	213-378-8361
Colorado	303-595-9299

COMPUTE! Publications, Inc., publishes
COMPUTE! **COMPUTE! Books** ***COMPUTE!'s Gazette***

Corporate Office:
505 Edwardia Drive, Greensboro, NC 27409

Mailing Address:
Post Office Box 5406, Greensboro, NC 27403

Telephone: 919-275-9809
Office Hours: 8:30 AM to 4:30 PM Monday-Friday

Chief Executive Officer Robert C. Lock
President Gary R. Ingersoll
Vice President of Finance W. Jerry Day
Comptroller James M. Hurst
Executive Assistant Debi Nash
Assistant Carol Dickerson

Subscription Information

COMPUTE!'s Gazette Circulation Dept.
P.O. Box 5406, Greensboro, NC 27403

TOLL FREE
Subscription Order Line
800-334-0868
In NC 919-275-9809

COMPUTE!'s Gazette
Subscription Rates

(12 Issue Year): US (one year) $20. Canada, Mexico and Foreign Surface Mail $25. Foreign Air Mail $45.

The COMPUTE!'s Gazette subscriber list is made available to carefully screened organizations with a product or service which may be of interest to our readers. If you prefer not to receive such mailings, please send an exact copy of your subscription label to: COMPUTE!'s Gazette, P.O. Box 961, Farmingdale, NY 11737. Include a note indicating your preference to receive only your subscription.

WordPro 3 Plus/64™

The Best Word Processor for your Commodore 64™

When choosing a Word Processor for your Commodore 64 computer, there's no reason to settle for anything but the best—in a word. . . . WordPro™.

With over 30,000 happy WordPro clients churning out letters and documents all over the world, the WordPro Plus™ Series is unquestionably the #1 selling software package on Commodore computers! So when you choose WordPro, you know you're investing in a trial-tested program that's a real winner. And WordPro 3 Plus/64 is NOW available for your Commodore 64 computer at prices as low as $89.95.

Designed for the novice user with no computer or word processing experience whatsoever, WordPro 3 Plus/64 brings a whole new dimension to the term "user-friendly." More than just easy to use, WordPro 3 Plus/64 is a commercial level word processor loaded with powerful features including: Auto Page Numbering, Headers and Footers, Math Functions, Global Search and Replace, the Ability to Create Multiple Personalized Letters and Documents, Underlining, Boldface, Super and Subscripts and much, much, more.

Best of all, WordPro 3 Plus/64's powerful arsenal of features can be put to use almost immediately—by even the novice user. So whether you're a writer, in business, education, or just a hobbyist, you'll quickly become a WordPro Pro!

Invest in the best. . . WordPro Plus. In a class by itself.

Call us today for the name of the WordPro 3 Plus/64 dealer nearest you.

Professional Software Inc.

51 Fremont Street
Needham, MA 02194

(617) 444-5224
Telex: 951579

WordPro 3 Plus/64™ is a trademark of Professional Software Inc.
The WordPro Plus Series was designed and written by Steve Punter of Pro-Micro Software Ltd.
Some printers may not support certain WordPro 3 Plus/64 functions and/or require an interface. Please check with your dealer.
Commodore 64™ is a trademark of Commodore Electronics Ltd.
Dealer and Distributor inquiries are invited.

GAZETTE FEEDBACK

EDITORS AND READERS

Do you have a question or a p roblem? Have you discovered something that could help other VIC-20 and Commodore 64 users? Do you have a comment about something you've read in COMPUTE!'s Gazette for Commodore? "Gazette Feedback" wants to hear from you. Write to Gazette Feedback, COMPUTE!'s Gazette, P.O. Box 5406, Greensboro, NC 27403.

Disk Menu

In the August 1983 issue of *COMPUTE!'s Gazette,* a "Disk Menu" program was published. In the "How It Works" section of the article, the author recommended saving the program as the first program on the disk. My problem is that I already have programs on each disk. Is it possible to save the Disk Menu program as the first program on my disks anyway?

Kevin Bergmann

To begin with, you don't have *to save Disk Menu as the first program on your disks. The author simply recommended this procedure because it makes it easier to load the menu. Entering LOAD"*",8 automatically loads the first program on a disk, no matter what it is. If the first program happens to be Disk Menu, you can quickly flip through the on-screen directory to load and run the program you want.*

You can insure that Disk Menu will be the first program on your disks by saving it immediately after formatting a blank disk with the NEW command.

On your partially full disks, you'll have to do a little rearranging to make Disk Menu the first program. Here's how:

1. Insert the disk you want to rearrange into the disk drive.

2. Enter LOAD"$",8 and then LIST to get a disk directory.

3. Note the program at the top of the directory list. (Press RUN/STOP if necessary to keep the list from scrolling off the screen.) This is the first program on the disk, the one you'll have to move to make room for Disk Menu.

4. Now use the COPY command explained on page 16 of the VIC-1541 User's Manual. *For example,*

CLOSE 15:OPEN 15,8,15, "COPY0:newfile=0:oldfile" where newfile *is a new filename you give the program, and* oldfile *is the existing filename as listed on the directory.* Be sure to give the program a different filename. *This makes another copy of the program elsewhere on the same disk.*

5. Once you've copied the program elsewhere on the disk, you can safely delete the original with the SCRATCH command explained on page 17 of the manual. For example:

CLOSE 15:OPEN 15,8,15,"SCRATCH0:oldfile"

6. Now there are two possibilities. If Disk Menu is already on this disk, repeat steps 4 and 5. The COPY command will move Disk Menu to the front of the disk, and you can then SCRATCH the original. If Disk Menu isn't already on the disk, insert a disk that contains it. LOAD Disk Menu. Insert the first disk into the drive again. Then SAVE Disk Menu.

7. If you've done everything right, Disk Menu should now be the first program on the disk. You can find out by entering LOAD"",8.*

Customizing Controls

The July 1983 article "Snake Escape" stated that lines 200-230 could be changed to accept any key command. Despite careful study, I cannot figure out how to accomplish this. What I want is to move *right* using the L instead of the K. Can you explain?

Roger Bingham

The first thing you must do is determine the keycode number for L. Use the sample routine as explained in the Snake Escape article on page 56. We'll repeat it here:

1 PRINT PEEK(197):FOR I=1 TO 400:NEXT:GOTO 1

RUN this routine, and hold down the L key. The number running down the screen is the keycode for L. You don't mention whether you have a VIC-20 or Commodore 64, but the keycode for L is 21 on the VIC and 42 on the 64. Depending on your computer, this is the number you must substitute in line 210 of either the VIC or 64 version. Line 210 detects the keypress for a move to the right (as denoted in the REM statement). If you have a VIC, substitute 21 for the 44 in line 210. If you have a 64, substitute 42 for the 37 in line 210. Movement to the right will now be controlled by the L key.

Presenting the class of 64.

The Commodore 64™ is one of the most exciting home computers in memory.

But memory isn't the only thing that's exciting about the 64.

Because Tronix is here.

Class act.

The people who have been bringing out the best in the VIC 20™ (and Atari® too) have graduated to the Commodore 64.

Which means that now you can enjoy fast action, complex strategies, interesting characters, superior sound effects and challenging, play patterns.

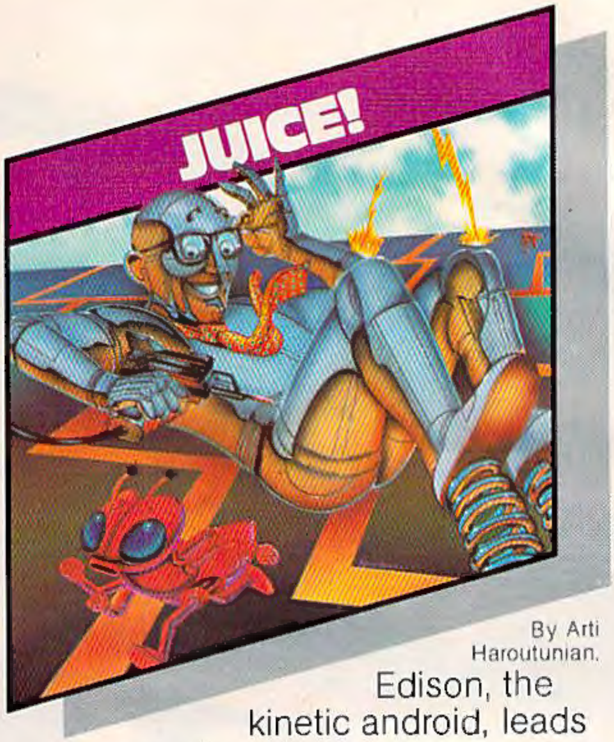
Just like VIC 20 and Atari owners. Only faster, more complex, and more challenging, too.

More memorable, in other words.

In a class by ourselves.

Of course, if you'd rather not take our word for it, you don't have to. The experts at Electronic Games have called *Kid Grid* for Atari "one of the most compulsive, utterly addictive contests in the world of computer gaming."

They haven't seen anything yet.



By Arti Haroutunian.

Edison, the kinetic android, leads a frustrating life.

All he wants to do is build his circuit boards and go with the flow. But things keep getting in the way.

Nohms—a negative influence—bug him constantly. Flash, the lightning dolt, disconnects everything in his path.

And the cunning Killerwatt is out to fry poor Edison's brains.

You'll get a charge out of this one. And a few jolts, too!

(Suggested retail $34.95)

By Arti Haroutunian.

Connecting the dots on our colorful grid should be easy, right?

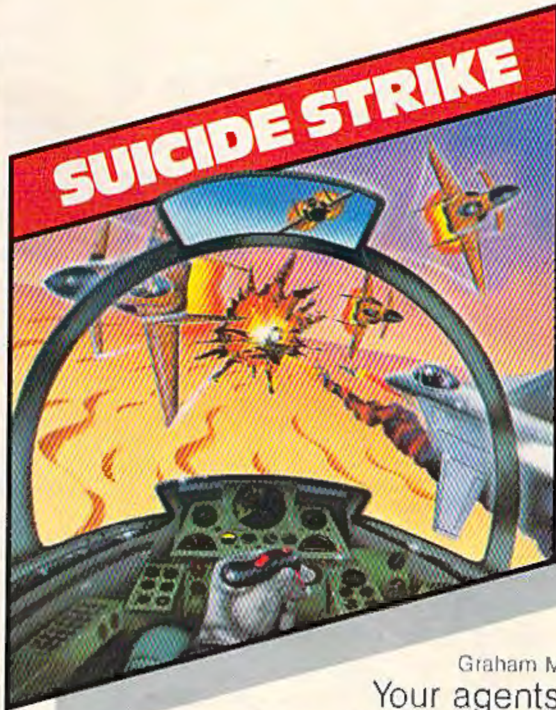
Wrong. Because the bullies are in hot pursuit!

Squashface, Thuggy, Muggy and Moose are their names. And *you* are their game. And what's more, they're faster than you are.

But you're smarter. And you control the stun button.

So keep your eyes peeled for the mysterious question mark and don't slow down at corners!

(Suggested retail: $34.95)



By Graham McKay.

Your agents risked their lives to find the enemy's secret headquarters.

Now you're risking yours to destroy it.

And they know you're coming.

As you fly over water and across hundreds of miles of unfriendly territory, the action is thick, fast and three-dimensional.

Fighter aircraft. Surface-to-air missiles. Helicopter gunships. The attacks come from every direction.

Even from behind.

(Suggested retail: $34.95)

TRONIX™

8295 South La Cienega Blvd., Inglewood, CA 90301

Incidentally, you might want to see the October 1983 "Bug-Swatter: Modifications & Corrections." A reader modified the 64 version of Snake Escape to work with a joystick instead of the keyboard.

TV Interference

We have a VIC-20 and two color TVs. The VIC is hooked up to the one in the family room. The other TV is in the living room. If both the computer and the other TV are on, the TV reception is bad. This problem is especially bad on channel 2. My sister (12 years old) and I (13) like to use the computer but sometimes can't because of the TV reception. I hope you can help.

Daina Jauntirans

The problem you are experiencing is called RF (radio frequency) interference. All computer circuitry emits RF signals which can interfere with nearby TV sets, often even the TV to which the computer is attached. Some computers emit more interference than others. The federal government, through the Federal Communications Commission (FCC), regulates how much interference is permitted for home computer devices and videogame machines. The manufacturers, in turn, must build enough shielding into their devices to meet the regulations.

Chances are you have an older-model VIC-20. Newer-model VICs (and Commodore 64s) that we've seen emit much less RF interference than earlier models, thanks largely to improved shielding.

There are a few steps you can take to minimize this interference. One solution is to locate the computer as far away from your family's second TV as possible. This might not be practical in your case, though. From the floorplan map you drew in your letter, it appears the computer and second TV already are pretty far apart. Also, the family room seems to be the most logical place for your computer.

You might try reorienting the computer, its TV, and the second TV. Sometimes changing the direction the computer faces and rearranging its wires can make a difference. Experiment with this. Also try plugging the computer into another wall outlet that is on a different branch circuit than the living room TV.

Some people wrap several layers of aluminum foil around the cable connecting the computer to the TV. This can help if the interference is coming from the cable. Also try wrapping the RF modulator — the small metal box between the computer and the TV (not the switchbox).

There's one more thing you might check, too. You mentioned that the interference is worst on channel 2. Look at the RF modulator box and see what channel it is tuned to. The VICs we've seen have RF modulators that select between channels 3 and 4, but it's possible that yours selects between 2 and 3. If so, make sure it is switched to channel 3. If your RF modulator selects between 3 and 4, try it on channel 4 (and remember in either case that the VIC's TV must be tuned to the same channel as the RF modulator). If there is no channel 4 in your area, this might reduce interference with channel 2 and yield a better computer picture. However, it also might cause new interference with channel 5 if there is a station on this channel in your area. The solution might be to switch the VIC to channel 4 when somebody in the living room is watching channel 2, and switch the VIC to channel 3 when somebody is watching channels 4 or 5.

For more information on correcting RF interference, you can write for a booklet prepared by the FCC entitled How to Identify and Resolve Radio-TV Interference Problems. *The booklet is listed as Stock Number 004-000-00345-4, available from the U.S. Government Printing Office, Washington, D.C. 20402.*

Stringy Floppy Chains

I very much enjoyed reading your review [COMPUTE!'s Gazette, July 1983] of the Entrepo Corporation (formerly Exatron) Stringy Floppy. I also liked the part on page 60 where you covered daisy chaining to the disk drive. I was almost ready to go out and invest in one. Your review, however, left me with one important question: does the Stringy Floppy have a serial port for us printer owners to daisy chain to?

James C. Nipert

Yes, the Stringy Floppy has two serial ports in the back — it can be at the end of the daisy chain or somewhere in the middle. No matter where it's located, it is addressed by its separate device number (20).

However, you might have trouble finding an Exatron Stringy Floppy now. The VIC and 64 models from Exatron were recently discontinued. Exatron was dissolved and is now known as Entrepo, Inc. Entrepo still manufactures Stringy Floppy drive assemblies, but no longer sells directly to the public. Instead, it sells the assemblies to other companies which finish the drives and then market the units under their own brand names. Sales of blank wafers are being handled the same way. Blank wafers are still available from one of these companies, A&J Microdrive (Sunnyvale, California). Jim Howell of A&J Microdrive says he may reintroduce the Stringy Floppy for the VIC and 64 sometime in the future, but there are no solid plans.

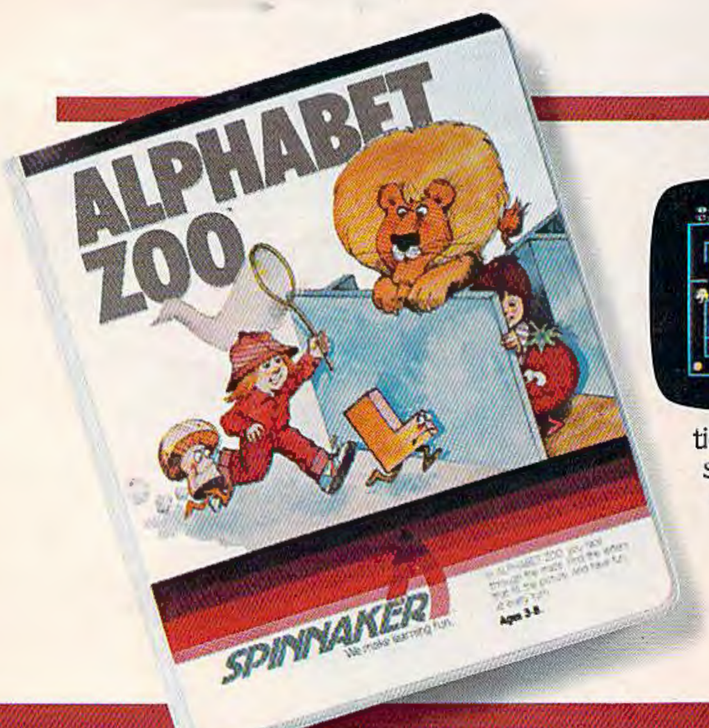
In the meantime, Stringy Floppies should be available this fall from another company, Unitronics (Oakland, California). Unitronics introduced its Waferdrive Module for the VIC and 64 at the Summer Consumer Electronics Show (see "News From The Summer CES," COMPUTE!'s Gazette, *September 1983). Unitronics plans to sell the Waferdrives for less than $100 (including 8K memory expansion on the VIC version) and make available some commercial software on wafers.*

SPINNAKER'S LINE OF EARLY LEARNING GAMES IS GROWING AS FAST AS YOUR CHILD'S MIND.

Watching your kids grow up is a lot of fun. But making sure their minds grow as fast as their bodies is even more rewarding. That's where we can help. With a growing line of Early Learning Programs that are not only lots of fun to play, but also educational.

Some of the games you see on these two pages help exercise your child's creativity. Others help improve vocabulary and spelling skills. While others improve your child's writing and reading abilities. And all of them help your child understand how to use the computer.

So if you're looking for computer programs that do more than just "babysit" for your kids, read on. You'll find that our Early Learning Programs are not only compatible with **Apple,® Atari,® IBM®** and **Commodore 64™** computers, but also with kids who like to have fun.



A trip through Alphabet Zoo.™ Ages 3 to 8.

It's a race. It's a chase. It's Alphabet Zoo, the exciting game that will have your kids zipping through the maze, after letters that fit the picture on the screen.

And at the same time, your kids will be learning the relationship of letters and sounds, and sharpening their spelling skills. So they'll be laughing and learning at every turn.

SIMPLE ANSWERS TO COMMON QUESTIONS

TOM R. HALFHILL, EDITOR

QA

Each month, COMPUTE!'s Gazette for Commodore will tackle some questions commonly asked by new VIC-20/ Commodore 64 users and by people shopping for their first home computer.

Q. ***The "Global Scan" program from the August 1983 issue of COMPUTE!'s Gazette seems to have an error on lines 500 and 510:***

500 IF H1 = 0 THEN H1 = 1E – 6
510 IF C = 0 THEN C = 1E – 6

In Commodore BASIC, 1E is not a proper variable name. I don't understand how this could work. Please help a confused programmer with a little correction for this bug.

A. This is not a bug. You are correct that 1E would not be a proper variable name; in practically all BASIC dialects, variables must start with a letter, not a number. However, 1E *is* a proper constant using what's known as *scientific notation*. This is a shorthand method of expressing very large or very small numbers without long strings of zeroes. The ability to recognize and manipulate numbers expressed in scientific notation is built into the computer.

For instance, 1E–6 means 1×10^{-6}, or 0.000001. The number 10,000,000, expressed as 1×10^{7} in scientific notation, would be entered on the VIC-20 or Commodore 64 as 1E7.

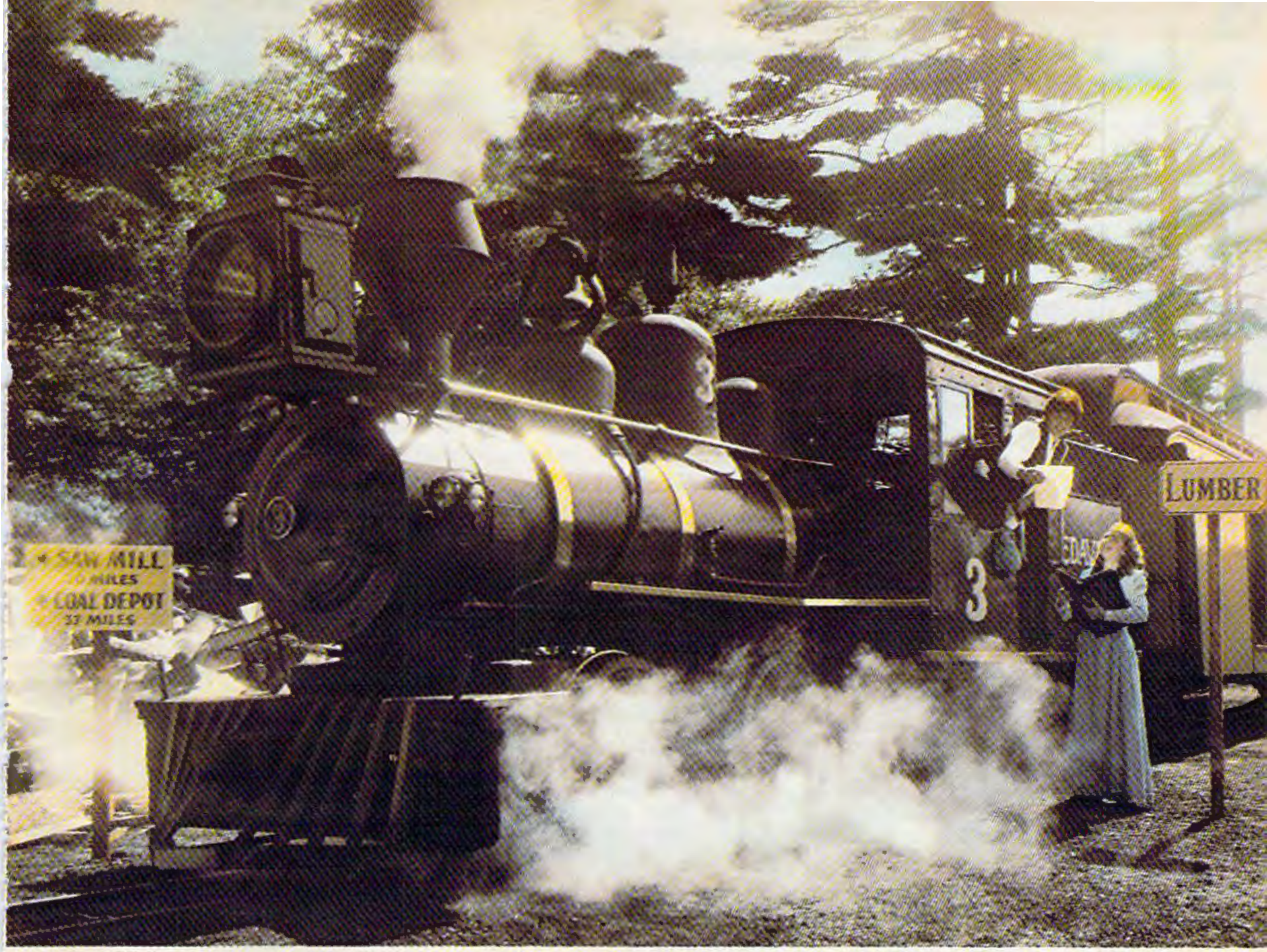
For more information on scientific notation, see the *VIC-20 Programmer's Reference Guide* (pp. 55-56) or the *Commodore 64 Programmer's Reference Guide* (pp. 4-6).

Q. ***I've been having some strange problems with my Commodore disk drive. Several files have gotten mixed up on some of my disks. Sometimes when I load a program, pieces of another program are mixed up with it. Some files won't load right at all. My manual says to "initialize" each disk and I haven't been doing that. Could that be the cause of my problem? Also, I've been using a friend's disks on my disk drive. Could this cause problems?***

A. You've zeroed in on your problem, all right. Initializing disks is *vital* to keep from garbling up your files. It's also possible that your disk drive is faulty, but true "hardware errors" are rare and should be suspected last.

Your problem, almost certainly, is the fault of a messed-up *Block Allocation Map* (BAM). Each disk contains a BAM. The BAM is a map of the remaining free space on the disk. The purpose of the BAM is to keep newly SAVEd files from overwriting old ones. Before SAVEing a new file, the disk drive checks the BAM to find free room on the disk. It then avoids writing data to blocks which have already been used. But if the BAM somehow gets garbled, the disk drive may SAVE a new file atop an existing one without realizing it.

How can a BAM become scrambled? Failing to initialize a disk is the most common way. Whenever you initialize a disk, write to a disk, or call a disk directory, the disk drive reads the BAM off the disk and stores it in its memory. Trouble can happen when two disks have the same ID number (the two-character identifier you assign when first formatting a disk with the NEW command). Disk drives use the ID number to distinguish between different disks. If you swap disks that happen to have the same ID number and then attempt to SAVE a file, the disk drive will first check the ID, conclude that the disk has not been changed, and use the BAM already in memory. Chances are pretty slim that both disks have identical BAMs, so files get overwritten or



Introducing a computer game that will bring out the railroad buff, the tycoon, the adventurer, and the kid in your kid.

TRAINS™ is one computer game that will really bring out the best in kids. Of all ages.

Because TRAINS is a Spinnaker game, which means it's a learning game that's really fun to play.

TRAINS puts kids in charge of an old-time railroad. And whether their railway empire gets bigger or goes out of business is entirely up to them.

As they juggle the challenges of picking up supplies and delivering to various industries, paying their employees, keeping the locomotive filled with coal, and making enough money to venture into new territories, kids are actually learning the economics of running a business.

They're learning to manage financial resources, and to use different kinds of information in setting priorities.

And best of all, they're having fun while they learn.

Look for TRAINS on disk at your local software retailer, and play it on your Apple®, Atari® or Commodore 64™ computer.

SPINNAKER™
We make learning fun.

scrambled. After this happens a couple of times, the BAMs themselves get messed up, and problems multiply.

To be safe, it's a good idea to initialize every time you swap disks in the drive. Without using a line number, enter:

CLOSE 15:OPEN 15,8,15,"I0":CLOSE 15 [press RETURN]

The drive's busy light will come on for a few seconds as the disk initializes. This insures that the disk's BAM is read into memory.

Another precaution when SAVEing programs on disk is to always precede the filename with a "0:" (on single-drive systems). Example:

SAVE"0:filename",8

To be really safe, you should make sure all your disks have unique ID numbers. If you sometimes use other people's disks on your system, give your own disks weird ID numbers that no one else would possibly think of, such as "Z=" or ":9" or "/*". Such symbols are legal in Commodore disk ID numbers.

For a more detailed explanation on using Commodore disk drives, see Part I of the new series starting this month, "Getting Started With A Disk Drive."

Tax Pack™

I designed Taxpack so you could do something really practical with your VIC 20.

Peter Lambert, MBA
Vice-President,
Product Development
Cosmopolitan Software

Taxpack
Powerful income tax computing software specially designed for the VIC 20.

Now you can use your VIC 20 to perform all the calculations on your Canadian T1 general tax form. *Taxpack* guides you easily through every aspect of the form with friendly prompts and a comprehensive instruction manual. This new software is available on cassette tape and will run on the standard 3.5k memory in your VIC 20 home computer.*
Taxpack lets you tackle your income tax form at your own pace. A convenient save-and-restore function lets you record and review historical results. Professional editing features assure easy and accurate data entry. *Taxpack* puts the power of tax modelling and planning for subsequent years in your hands, today.

Many happy returns
Because you can calculate and preview more tax scenarios with *Taxpack* than you'd have the patience or the time to do manually, this software can help you save tax dollars. Custom-tailored to the Canadian T1 general form, *Taxpack* will be updated every year to reflect changes in the government's income tax regulations. Innovative program design allows us to update *Taxpack* within days of the new T1's availability.

Special introductory offer
Order early and get your Taxpack for only $19.95! That's a ten dollar saving off our regular retail price of $29.95. If you're giving *Taxpack* for Christmas, we'll send you a special gift card to put under the tree.
To use your *Visa* or *Mastercard*, phone us toll free; or, send your cheque or money order with the handy mail-order form attached. We'll confirm your order by return mail. Your up-to-date *Taxpack* cassette and manual will be shipped within 15 days of the release of the 1983 T1 general form.

**The cassette also includes an expanded version of Taxpack with enhanced display features, for the VIC 20's with 8k+ memory expansion.*

To order with Visa or Mastercard call us toll-free:

1-800-268-6364

(from B.C., call 112-800-268-6364)

Satisfaction Guaranteed
We guarantee that you will find *Taxpack* an excellent software value. If you are not totally satisfied, drop us a note to say why, and return the product post paid to us within 10 days for a full refund of the purchase price.

I Want Taxpack!

Please send me ______ Taxpacks @ $29.95	$______
Discount $10 per unit for orders before Dec 31, 1983	– $______
Subtotal	$______
Nova Scotia residents only, add 10% Sales Tax	+ $______
Add $2 per unit shipping and handling charges	+ $______
Total	$______

My Name
Address
City Province
Postal Code Telephone

☐ I am buying Taxpack as a gift. Please send me a gift card. Attached please find the name and address of the person(s) to receive Taxpack.
Make Cheque or Money Order payable to:
Cosmopolitan Software Services Limited
and mail with this order form to:
Box 953 Dartmouth, Nova Scotia B2Y 3Z6 Attn: Order Desk

Computer Graphics

The Age Of Electronic Art

Fred D'Ignazio, Associate Editor

Let's imagine we have a time machine. We climb on board. We shut the door. We set the controls for Cambridge, Massachusetts, in early 1961.

The time machine whirs. Our stomachs feel queasy, as if we were on a rapidly falling elevator.

The whirring stops. The door opens.

We are in a darkened laboratory. The hulking forms of giant computers tower overhead and surround us. In the laboratory is a young man, unshaven, gazing at the screen of a computer terminal.

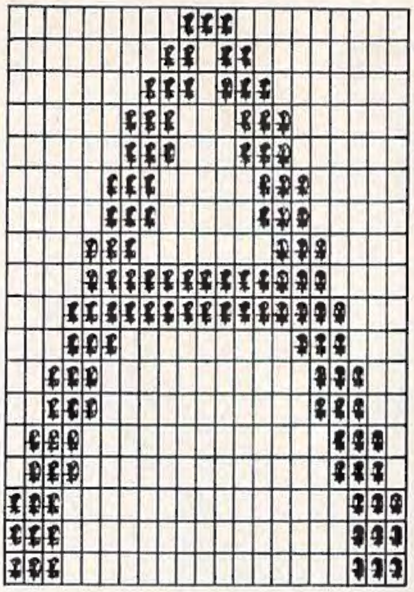
The young man is Ivan Sutherland. Ivan is a graduate student at the Massachusetts Institute of Technology in Cambridge. He sits, frozen, in front of the computer. It is 3:30 in the morning. His wife and kids are home sleeping. But here he is, dead tired, half-asleep, and eyes glazed. Yet he has to be here. It is the only time he is allowed on the computer.

Sutherland stares at the computer screen. A shape appears. Sutherland grins. "Yaaaaaa-hoooo!" he cheers. He dances around the deserted lab.

Why is Sutherland so happy? Because he has just become the first human being to teach a computer to draw.

On the screen was a straight line. That's all: just a straight line. It was special because it was the first line ever drawn by a computer – and because it opened a whole new age of electronic art.

The view out the computer "window" of an Air Force flight simulator. The ultra-high resolution computer images of the jets and the ground below are in color, in 3-D, and in motion. (Courtesy Evans & Sutherland.)



More simple graphics: a giant letter A made from several characters printed one on top of the other with a printer. Note that the A is printed on a graph-paper grid of little blocks. (Reprinted from Creative Kid's Guide to Home Computers *by permission of Doubleday & Company.)*

Sutherland was excited about what he did. But he didn't stop there; he went on to teach the computer how to draw new things. He taught the computer to change the straight line into an elastic rubber band. He had the computer bend the line, shrink it, and stretch it.

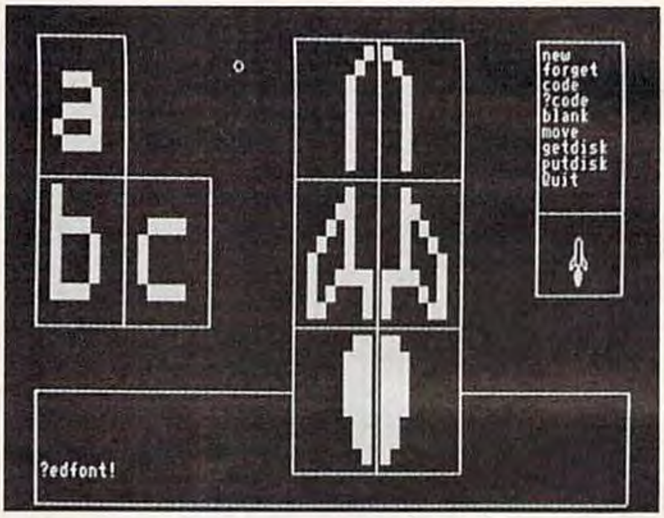
He taught the computer to draw other shapes, too: circles, triangles, squares, and polygons. He taught the computer to spin the circles, rotate the triangles, and fold the squares like pieces of construction paper.

Sutherland published his findings in a book called *The Electronic Sketchpad.* His enthusiasm and knowledge about computer art inspired a whole generation of young people to learn how to teach computers to draw pictures.

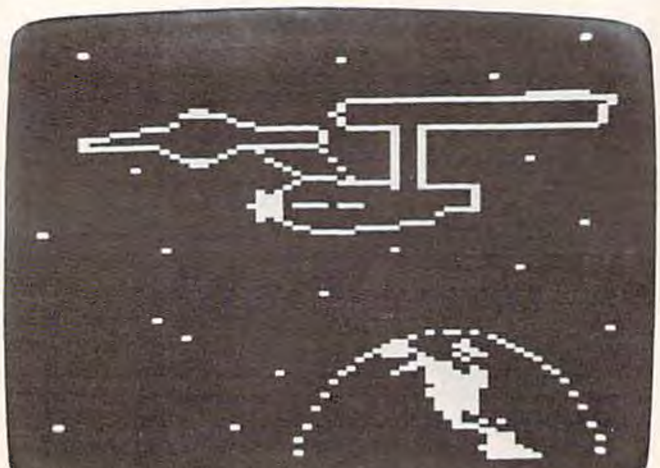
Sutherland and his friend, David Evans, moved to Salt Lake City, Utah, and founded the world's best *computer graphics* company, Evans & Sutherland.

Evans & Sutherland now builds million-dollar flight simulators for the U.S. Air Force. The simulators look like the inside of an airplane's cockpit – except that the "windows" are all computer screens. When a pilot trainee operates the controls of the plane, he or she is really controlling the computer. The computer images on the windows look like what pilots would see if they were flying a real plane.

Sutherland and his graphics computers are world-famous. Yet it all started one cold, dark morning when he taught a computer how to draw a straight line.



You can build complicated shapes by combining simple, rectangular building blocks or special graphics characters. Shown are three letters of the alphabet and a rocket. When the rocket is reduced, it becomes much more realistic. (Photo by Alice Collette. Courtesy of Rosetta Inc. Reprinted from Small Computers *with permission of the publisher, Franklin Watts.)*

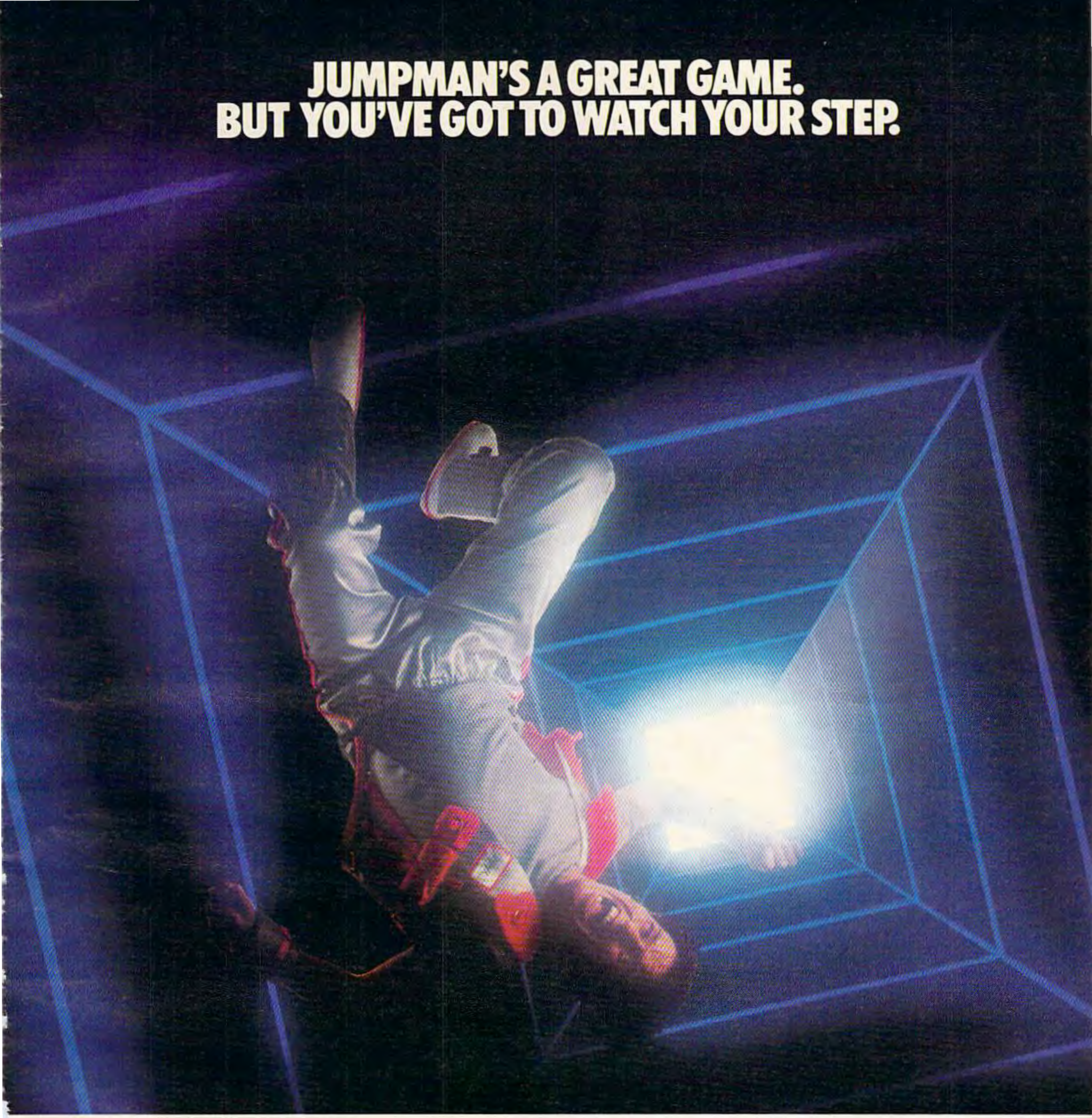


Picture of stars, Earth, and the Starship Enterprise. *A computer program lights the pixels on the TV screen. (Courtesy RCA.)*

Amazingly, your average personal computer is more powerful than Ivan Sutherland's warehouse-sized computer of 1961. Also, many of today's personal computers have graphics commands built right into their BASIC language. Sutherland had to program his computer to draw by feeding it commands written in long, snake-like strings of binary 1's and 0's. Modern computers can draw pictures with English-like commands such as PEN DOWN, DRAWTO, PLOT, and FILL.

Also, Sutherland's graphics were all in black and white. But today's computer graphics can be in color – from 16 colors on a VIC-20 and Commodore 64 up to 256 different colors on some machines.

But no matter how complicated computer graphics get, they must be built in one of three ways.



Meet the Alienators. A fiendish bunch who've planted bombs throughout your Jupiter Command Headquarters.

Your job? Use your lightning speed to scale ladders, scurry across girders, climb ropes and race through 30 levels to defuse the bombs before they go off.

That's the kind of hot, non-stop action we've packed into the award-winning,* best-selling Jumpman,™ and into Jumpman Jr.,™ our new cartridge version with 12 all-new, different and exciting screens.

Both games force you to make tough choices. Should you avoid that Alienator, climb to the top and try to work your way down, or try to hurdle him and defuse the bombs closest to you before they go off?

If you move fast you'll earn extra lives. But if you're not careful, it's a long way down.

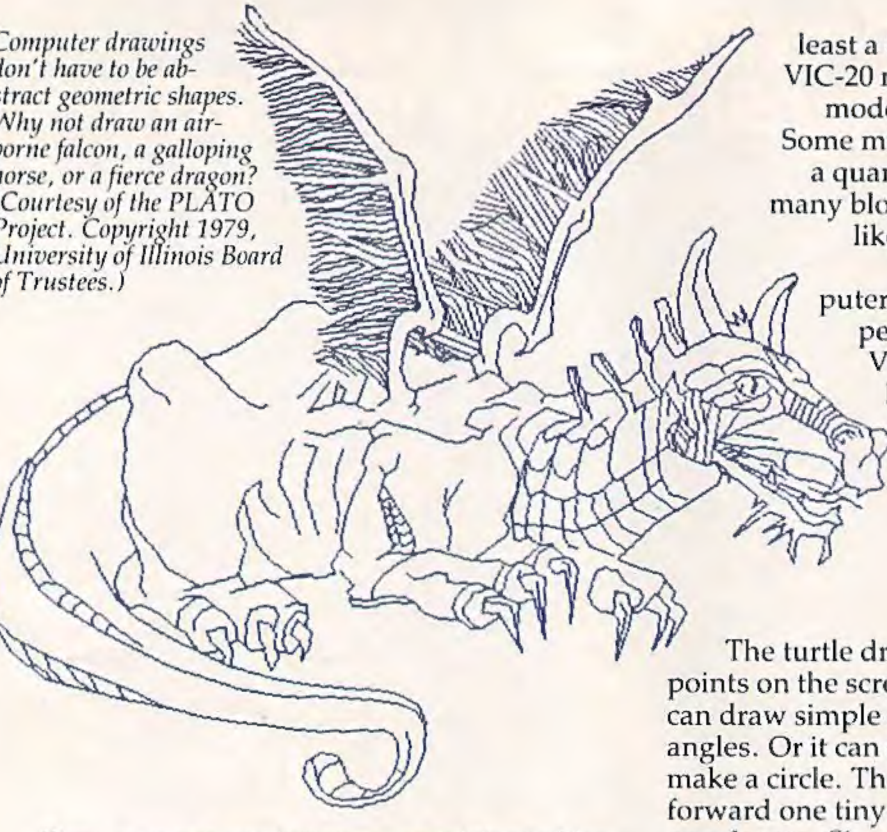
So jump to it. And find out why Jumpman and Jumpman Jr. are on a level all their own.

One to four players; 8 speeds; joystick control. Jumpman has 30 screens. Jumpman Jr. has 12 screens.

*1983 C.E.S. award winner.

EPYX COMPUTER SOFTWARE

STRATEGY GAMES FOR THE ACTION-GAME PLAYER.



Computer drawings don't have to be abstract geometric shapes. Why not draw an airborne falcon, a galloping horse, or a fierce dragon? (Courtesy of the PLATO Project. Copyright 1979, University of Illinois Board of Trustees.)

First, you can use letters or special graphics characters and combine them into some kind of shape. The shape might be a giant letter A, a monster's face, or a skyscraper. These kinds of graphics are holdovers from the days when most computer terminals used paper printers instead of video display screens.

Another way to draw pictures on a computer's TV screen is to divide the screen up into tiny blocks called *pixels* (for *picture elements*). The picture is built by filling in the blocks. It's like drawing rough pictures with graph paper and magic markers. Using this technique, you can make pictures of spaceships, human stick figures, or running horses.

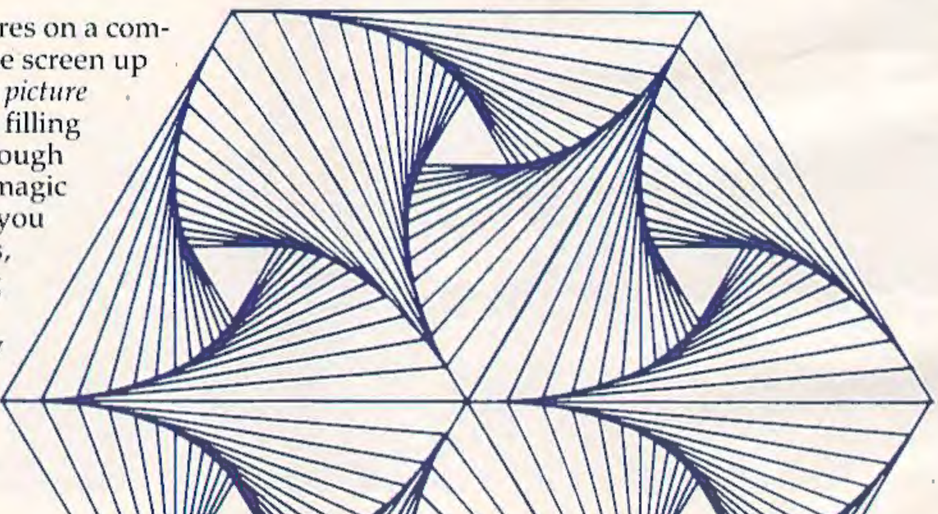
Computers that can display large numbers of very small pixels are capable of creating images with finer resolution. The more pixels, the better. If your TV screen is divided into thousands of tiny building blocks, then the pictures look smooth and realistic. On the other hand, if the building blocks are big, then the pictures appear blocky and rough. The image which opens this article was created on a computer with ultra-high resolution – millions of pixels.

Most personal computers let you draw pictures using at least a thousand blocks per screen. The VIC-20 maximum is 32,384; on the Commodore 64, up to 64,000 are possible. Some microcomputers let you use up to a quarter of a million blocks. With this many blocks on the screen, they look less like blocks and more like tiny dots.

Yet another way to create computer art is to use *turtle graphics*. Most personal computers, including the VIC and 64, can run the PILOT or Logo languages. In turtle graphics, a small, imaginary turtle (often the shape of a triangle) walks across the screen while leaving behind a trail. By issuing commands such as TURN 90 and FORWARD 10, shapes are drawn.

The turtle draws pictures by connecting two points on the screen with a straight line. The turtle can draw simple shapes such as squares or triangles. Or it can draw lots of little straight lines to make a circle. The turtle makes the circle by going forward one tiny space, then turning to the right one degree. Since it takes 360 degrees to make a circle, the turtle has to go forward and turn right 360 times.

The turtle doesn't have to stick to circles. It can make stars, snowflakes, even fierce dragons. It can make any number of complicated, beautiful



Simple shapes can be combined into beautiful, three-dimensional shapes with turtle graphics. (Courtesy of computer artist Joe Jacobson.)

WELCOME TO APSHAI. YOU'RE JUST IN TIME FOR LUNCH.

Boy, have you taken a wrong turn. One moment you're gathering treasure and the next you're being eyed like a side of beef.

You're in the Gateway to Apshai.™ The new cartridge version of the Computer Game of the Year,* Temple of Apshai.™

Gateway has eight levels. And over 400 dark, nasty chambers to explore. And because it's joystick controlled, you'll have to move faster than ever.

But first you'll have to consider your strategy.

Is it treasure you're after? Or glory? You'll live longer if you're greedy, but slaying monsters racks up a higher score.

The Apshai series is the standard by which all other adventure games are judged. And novices will not survive.

They'll be eaten.

One player; Temple of Apshai, disk/cassette; Gateway to Apshai, cartridge, joystick control.

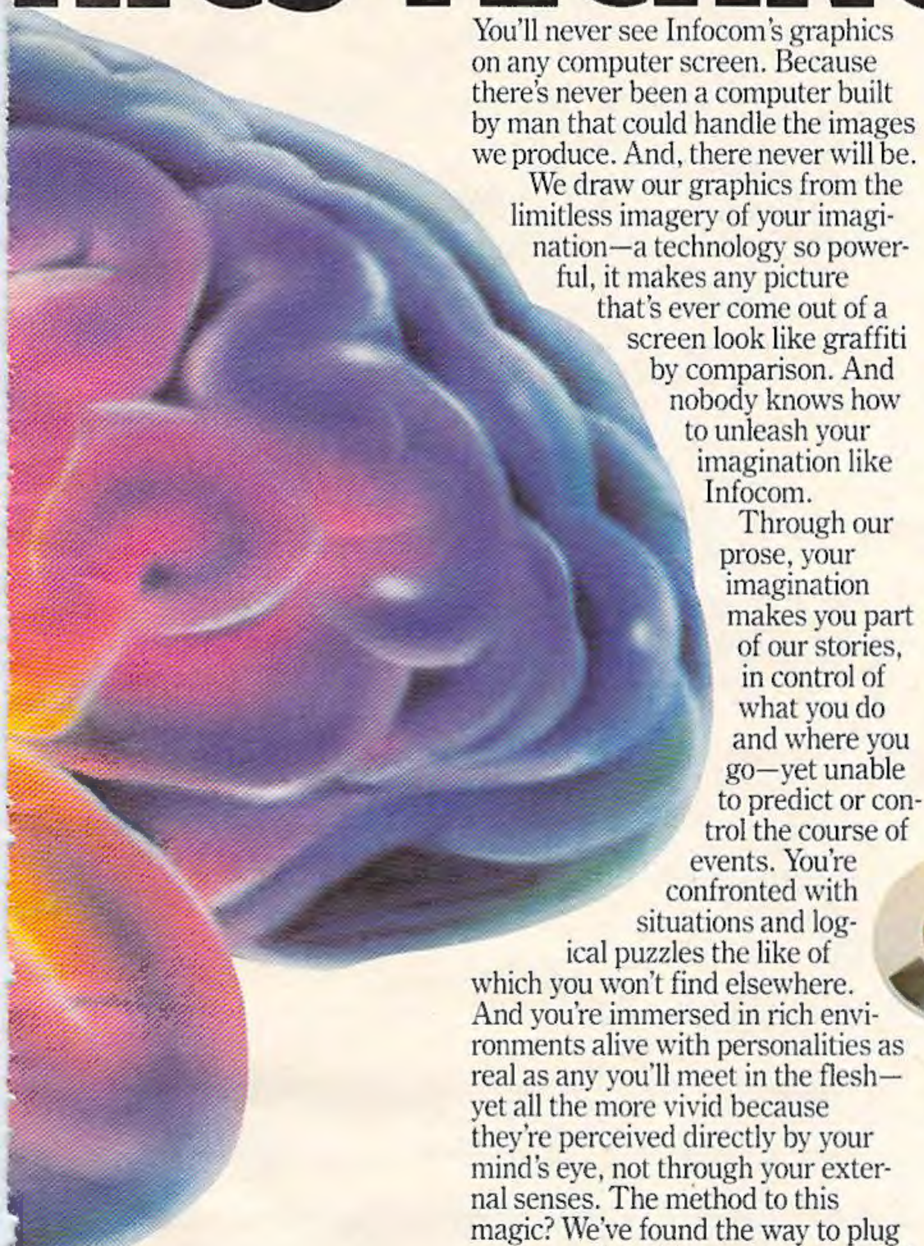
EPYX COMPUTER SOFTWARE

STRATEGY GAMES FOR THE ACTION-GAME PLAYER.

*Game Manufacturers Association, 1981

WE UNLEASH TH
POWERFUL GRAP

E WORLD'S MOST
HICS TECHNOLOGY.



You'll never see Infocom's graphics on any computer screen. Because there's never been a computer built by man that could handle the images we produce. And, there never will be.

We draw our graphics from the limitless imagery of your imagination—a technology so powerful, it makes any picture that's ever come out of a screen look like graffiti by comparison. And nobody knows how to unleash your imagination like Infocom.

Through our prose, your imagination makes you part of our stories, in control of what you do and where you go—yet unable to predict or control the course of events. You're confronted with situations and logical puzzles the like of which you won't find elsewhere. And you're immersed in rich environments alive with personalities as real as any you'll meet in the flesh—yet all the more vivid because they're perceived directly by your mind's eye, not through your external senses. The method to this magic? We've found the way to plug our prose right into your psyche, and catapult you into a whole new dimension.

Take some tough critics' words about our words. SOFTALK, for example, called ZORK® III's prose "far more graphic than any depiction yet achieved by an adventure with graphics." And the NEW YORK TIMES saw fit to print that our DEADLINE™ is "an amazing feat of programming." Even a journal as video-oriented as ELECTRONIC GAMES found Infocom prose to be such an eye-opener, they named one of our games their Best Adventure of 1983.

Better still, bring an Infocom game home with you. Discover firsthand why thousands upon thousands of discriminating game players keep turning everything we write into instantaneous bestsellers.

Step up to Infocom. All words. No graffiti. The secret reaches of your mind are beckoning. A whole new dimension is in there waiting for you.

INFOCOM™

The next dimension.

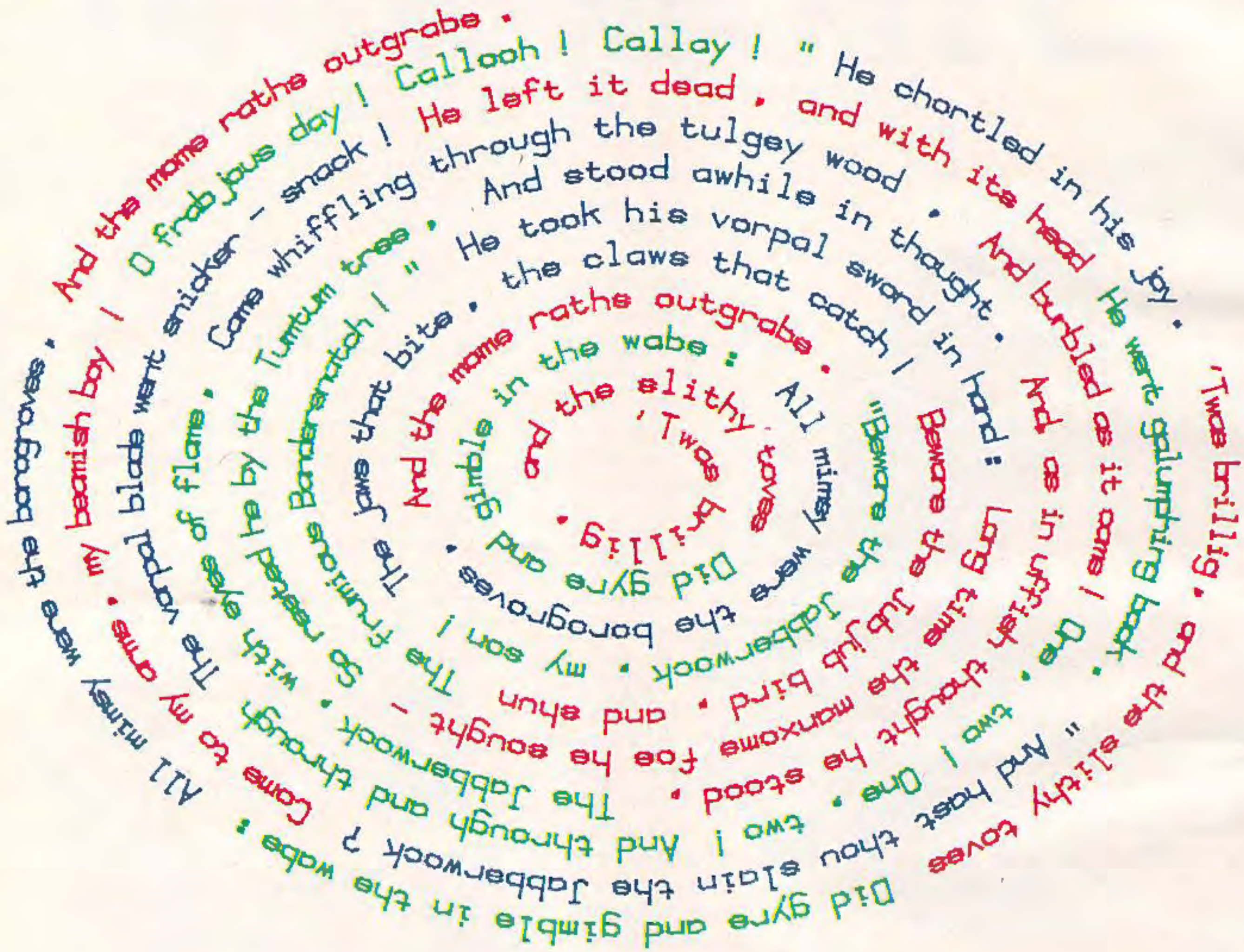
Infocom, Inc., 55 Wheeler St., Cambridge, MA 02138

For your: Apple II, Atari, Commodore 64, CP/M 8", DEC Rainbow, DEC RT-11, IBM, NEC APC, NEC PC-8000, Osborne, TI Professional, TRS-80 Model I, TRS-80 Model III shapes – all from tiny straight lines (see example printout).

The turtle usually draws with lines made up of dozens of blocks strung together. But its lines could be made up of special figures or letters of the alphabet (see printout).

Computer art will be more like drawing with a pen, pencil, or paintbrush.

One relatively new device is the *light pen*. The pen is wired to the computer. As you touch the tip of the pen to the screen and move it across the glass, it leaves an electronic "line." Some light



Jabberwocky

Figures can even be drawn with words. (Courtesy of computer artist Joe Jacobson. Reprinted from Creative Kid's Guide to Home Computers *with permission of Doubleday & Company.)*

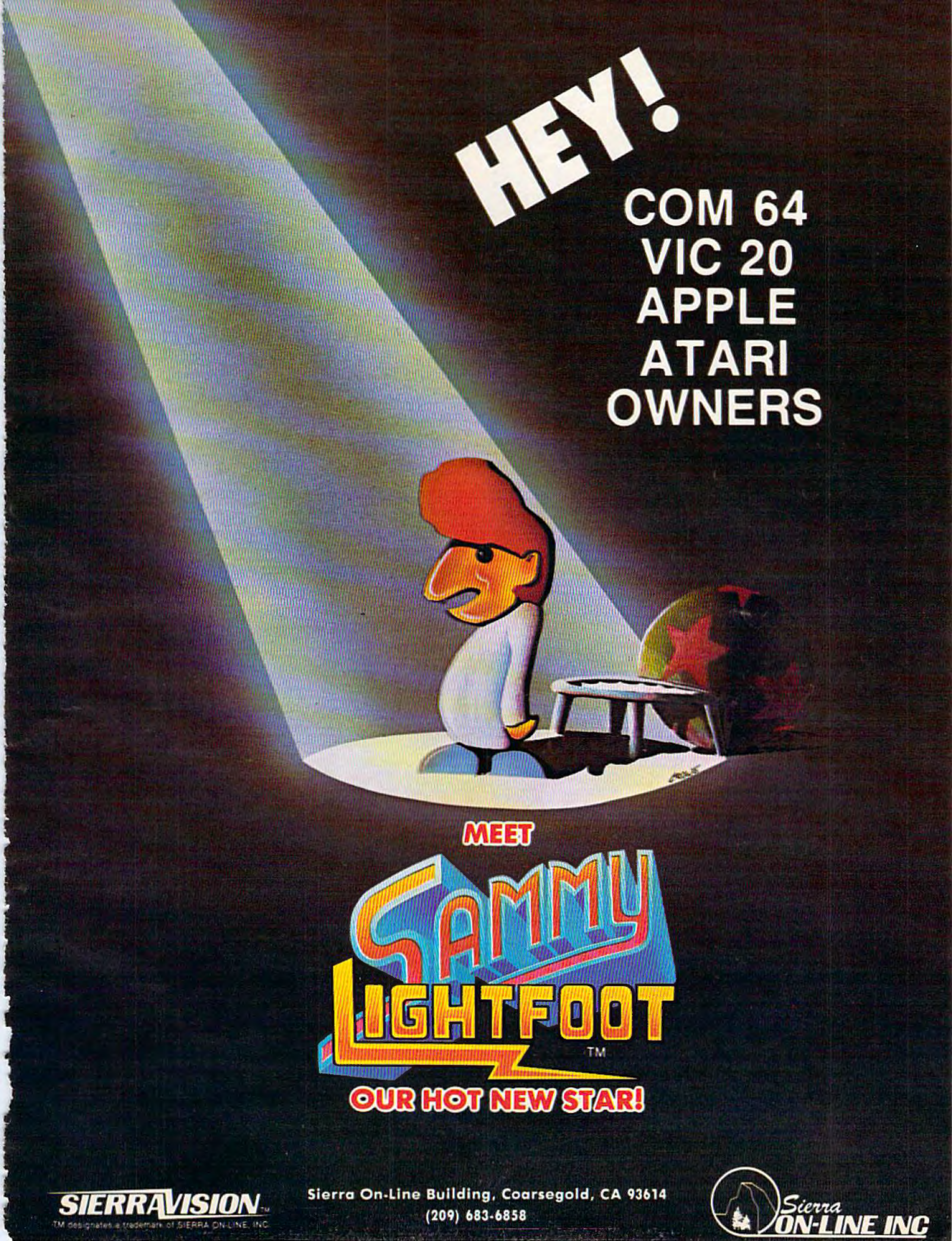
Until recently, the only way to make computer graphics was to type commands on the computer keyboard. But that's not the way traditional artists draw. They use pencils, pens, and paintbrushes. They draw the picture directly on a piece of paper. They don't have to type a PAINT command on their paintbrush, or a SKETCH command on their pencil.

Now computers are being made to work with equipment and programs that will make it as easy to draw pictures on a video screen as on paper.

pens draw in different colors. Others let you touch the screen and fill in a whole picture with a particular color. With a single touch, you could paint the sky blue, clouds white, or a robot metallic orange (see photo).

Another device is the graphics tablet. The tablet is a flat plastic rectangle or square. You mount a picture you want to trace on top of the tablet. Above your picture is a plastic arm, often with a magnifying glass on the end.

You trace the picture by moving the tip of the


HEY!
COM 64
VIC 20
APPLE
ATARI
OWNERS
MEET
SAMMY
LIGHTFOOT
TM
OUR HOT NEW STAR!
SIERRAVISION TM
TM designates a trademark of SIERRA ON-LINE, INC.
Sierra On-Line Building, Coarsegold, CA 93614
(209) 683-6858
Sierra ON-LINE INC

Drawing pictures directly on the TV screen with a light pen. (Courtesy of Matsushita Electric.)

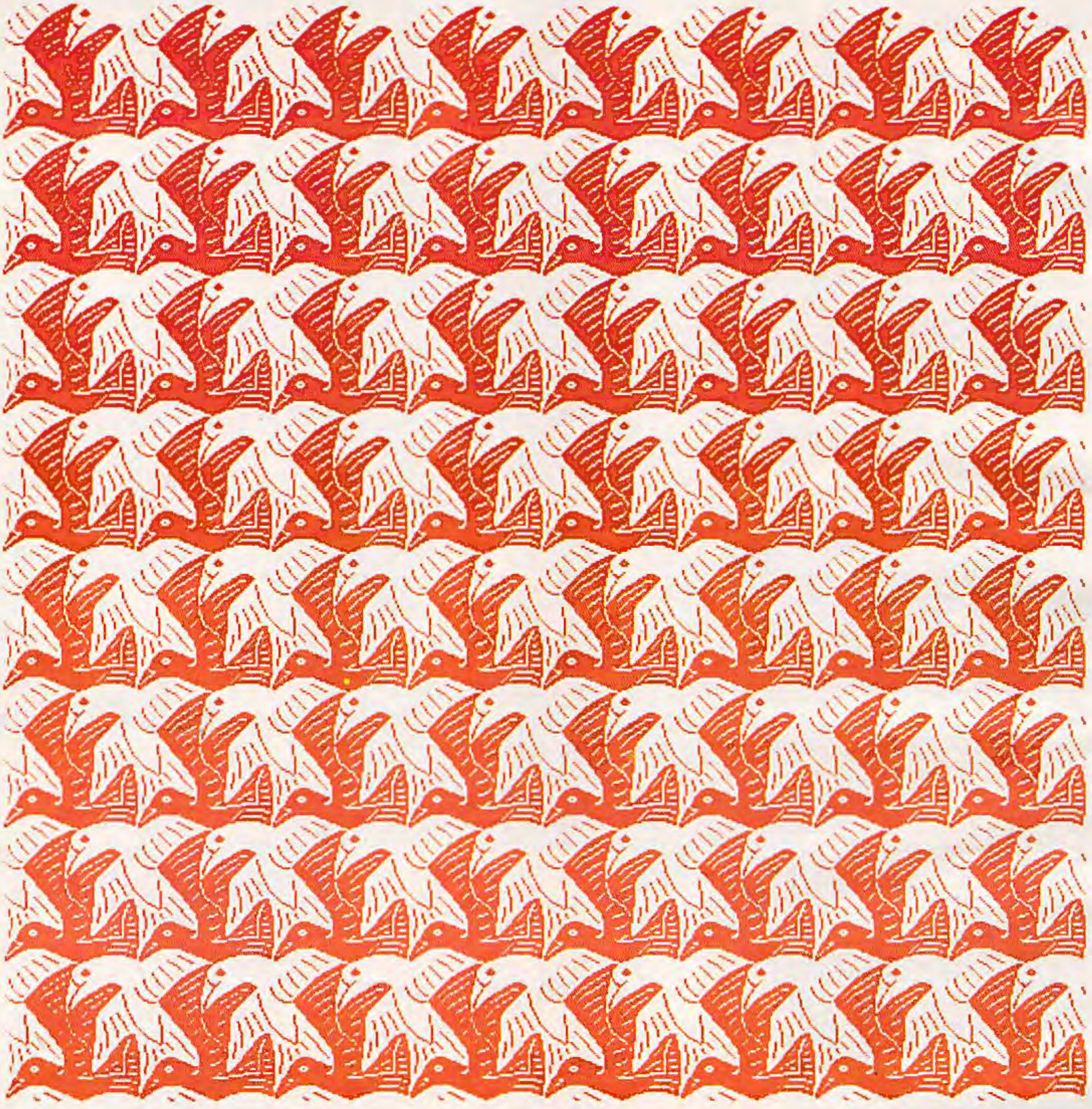
plastic arm above each line in the picture. The arm and the graphics tablet are wired to the computer. As you move the arm, it sends signals to the computer. These signals are an electronic copy of your picture, stored in the computer's memory. With just a couple of commands, you can get the computer to display the picture on the TV screen. It is fuzzier than the original, and the colors are different, but it is still very much like the picture you traced (see photo).

You can use a graphics tablet to make computer images of maps, photographs, shapes, drawings – almost anything. You also draw original pictures of your own on graphics tablets. An electronic copy is stored inside the computer and appears on the TV. On some computers, you can manipulate the picture once it is inside the computer: shrink it, expand it, change its colors, spin it around, or stretch it apart like Silly Putty (see picture made by duplicating birds).

Million-dollar mainframe computers are being used more and more to create graphics for things like flight simulators and Hollywood movies (such as last year's *TRON*). These computers are becoming so powerful that, sometimes, their pictures look like photographs of real people, real places, and real things.

Moviemakers and human artists are pro-

With a graphics tablet, you can copy maps, photographs, or diagrams into the computer by tracing them with the graphics arm wired to the computer. (Courtesy of Versawriter Inc.)

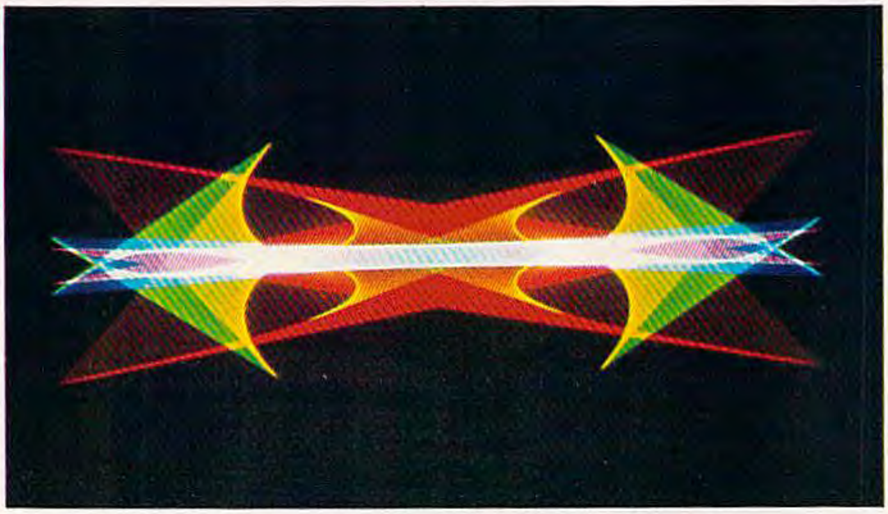


In this picture, a single pair of birds was copied into the computer. The computer duplicated the two birds dozens of times to make this beautiful pattern. (Courtesy of the PLATO Project. Copyright 1979, University of Illinois Board of Trustees.)

gramming computers to create movie and TV scenes that would be too expensive or impossible to stage with live human actors. In days gone by, a team of human artists would paint these scenes (called *mattes*). Now humans program computers to paint the scenes. And the scenes aren't frozen, like a still photo. The computer brings the scenes to life. They become realistic computer cartoons.

Computers that draw scenes for movies are far too expensive to become personal computers today. But scientists are inventing a new generation of special-purpose computer chips that will soon drastically reduce the cost of these computers. Million-dollar computers can now fit on a $5000 graphics chip. Hopefully, one day soon, these chips will be inside personal computers. The result will be the birth of a new era in computer graphics and electronic art.

nüFEKOP
P.O. Box 156, 21255 Hwy. 62,
Shady Cove, OR 97539
1-800-525-2529
Software for the VIC 20™
and Commodore 64™
3-D MAN
THE CATCH
DEFENDER ON TRI
Exterminator
VIC 20 programming
nüFEKOP



This stunning picture was made on a supercomputer. But soon, pictures like this will be possible on everyday personal computers. (Courtesy of computer artist Dr. Melvin L. Prueitt, Los Alamos National Laboratory.)

Further Reading

For an overview of computer graphics and computer art, I recommend the following books:

D'Ignazio, Fred. *The Creative Kid's Guide to Home Computers.* New York: Doubleday and Company, Inc., 1981.

D'Ignazio, Fred. *Electronic Games.* New York: Franklin Watts, Inc., 1982.

D'Ignazio, Fred. *Messner's Introduction to the Computer.* New York: Simon & Schuster/Julian Messner, 1983. (This book contains my interviews with many of the pioneers in computer graphics and computer art.)

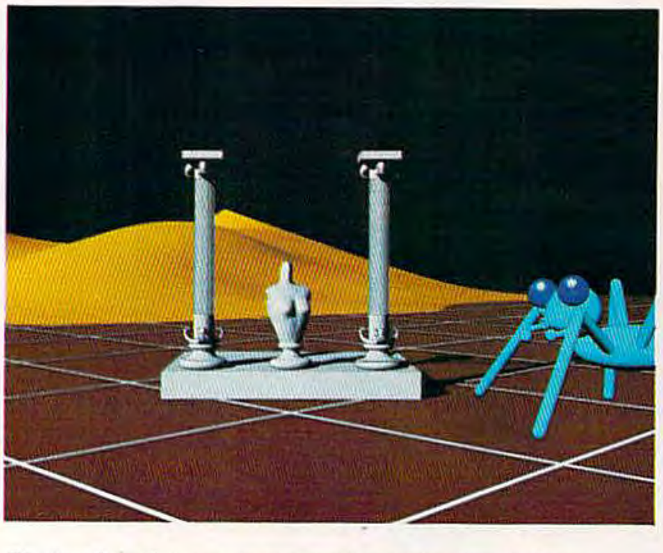
Greenberg, Donald (and others). *The Computer Image: Applications of Computer Graphics.* Reading, MA: Addison-Wesley, 1982. (This is an expensive book. But it has beautiful color pictures and is a good, clear introduction to computer graphics. I recommend it for your computer class at school or for your library.)

Leavitt, Ruth. *The Artist and the Computer.* New York: Harmony Books, 1976.

If you are ready to start creating computer art of your own, you should look at:

Myers, Roy. *Microcomputer Graphics.* Reading, MA: Addison-Wesley, 1982. (Lots of Apple II BASIC examples.)

Thornburg, David. *Picture This!* Reading, MA: Addison-Wesley, 1982. (Lots of examples in Logo and Atari PILOT.)



Electronic artists are programming computers to draw cartoon-like scenes and cartoon characters and creatures. (Courtesy of MAGI Inc.)

The hardest thing for computers to draw is people. Now scientists at Ohio State University and elsewhere are working to program computers to draw realistic pictures of human beings in action. (Courtesy of Dr. Charles Csuri, Ohio State University.)

Calc Result™

The Commodore 64™ Spreadsheet that puts you a million miles ahead

CALC RESULT...The one spreadsheet guaranteed to turn your Commodore into a powerful financial tool. Offering you every feature found on other more expensive programs for much less the cost.

Flexible...you can view four different areas at once

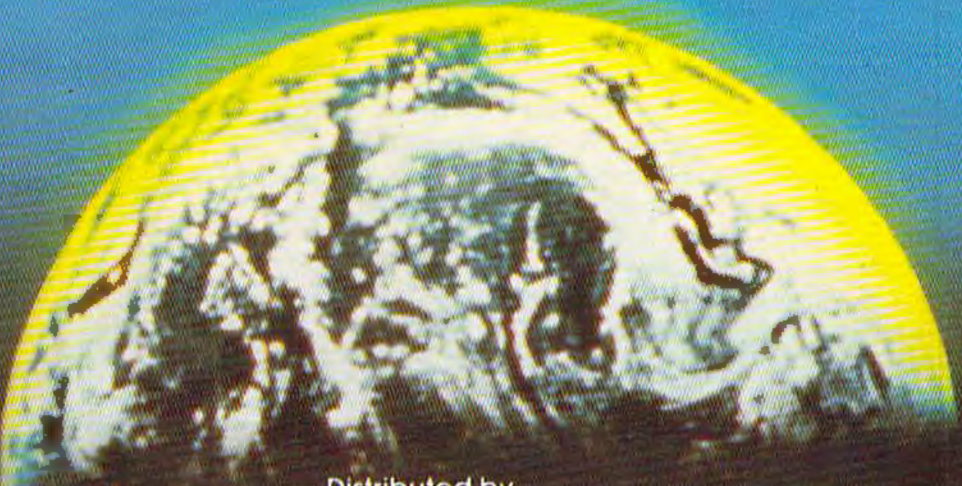
Versatile...customize your own print formats

Distinctive...display beautiful color graphics

CALC RESULT *Advanced* is a three-dimensional spreadsheet with built-in HELP function and 32 pages of memory. For the Commodore 64 $149.95. For the CBM™ 8032 $199.00.

For first time users CALC RESULT *Easy* gives you a fast way to perform financial calculations—easily. For the Commodore 64 $79.95.

For a down to earth demonstration of either version visit your local dealer today.



handic software ab

-a company in the Datatronic group-

Distributed by

Computer Marketing Services, Inc.

300 W. Marlton Pike Cherry Hill, NJ 08002

609 795 9480

The Inner World Of Computers

Part 1: Binary Numbers

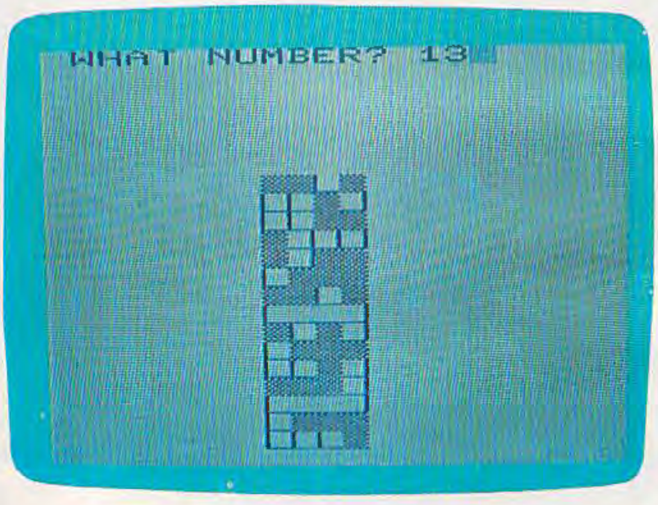
Tom Prendergast

Do you ever wonder what happens after you type RUN? What goes on inside the computer? How a machine can "think" just by manipulating numbers? This new series shows how computers work by explaining computer math in a nontechnical way. It's especially recommended for those who are following our monthly column "Machine Language For Beginners."

I hated math in school. I skipped more math classes than some people skip gym. Numbers made me numb. Especially when the teacher spent every period filling up the blackboard with funny little chalkmarks – then erasing the whole board before explaining what he'd done.

That's why I wouldn't buy a computer. They scared me. A computer is for numbers, right? And who needs a computer when you can buy a calculator that fits into your shirt pocket and costs a whole lot less! Then I bought a word processor to replace my typewriter – and discovered I'd bought a computer!

I discovered something else after I'd been running my computer-disguised-as-a-word-processor for a couple of months: that I shouldn't have been scared of computers in the first place. Because computers don't use numbers! Sure, they *do* numbers. Computers do addition, subtraction, multiplication, division – even square roots and sines and cosines and all that advanced mathematical stuff – but computers don't "understand" numbers any better than I did, or do.

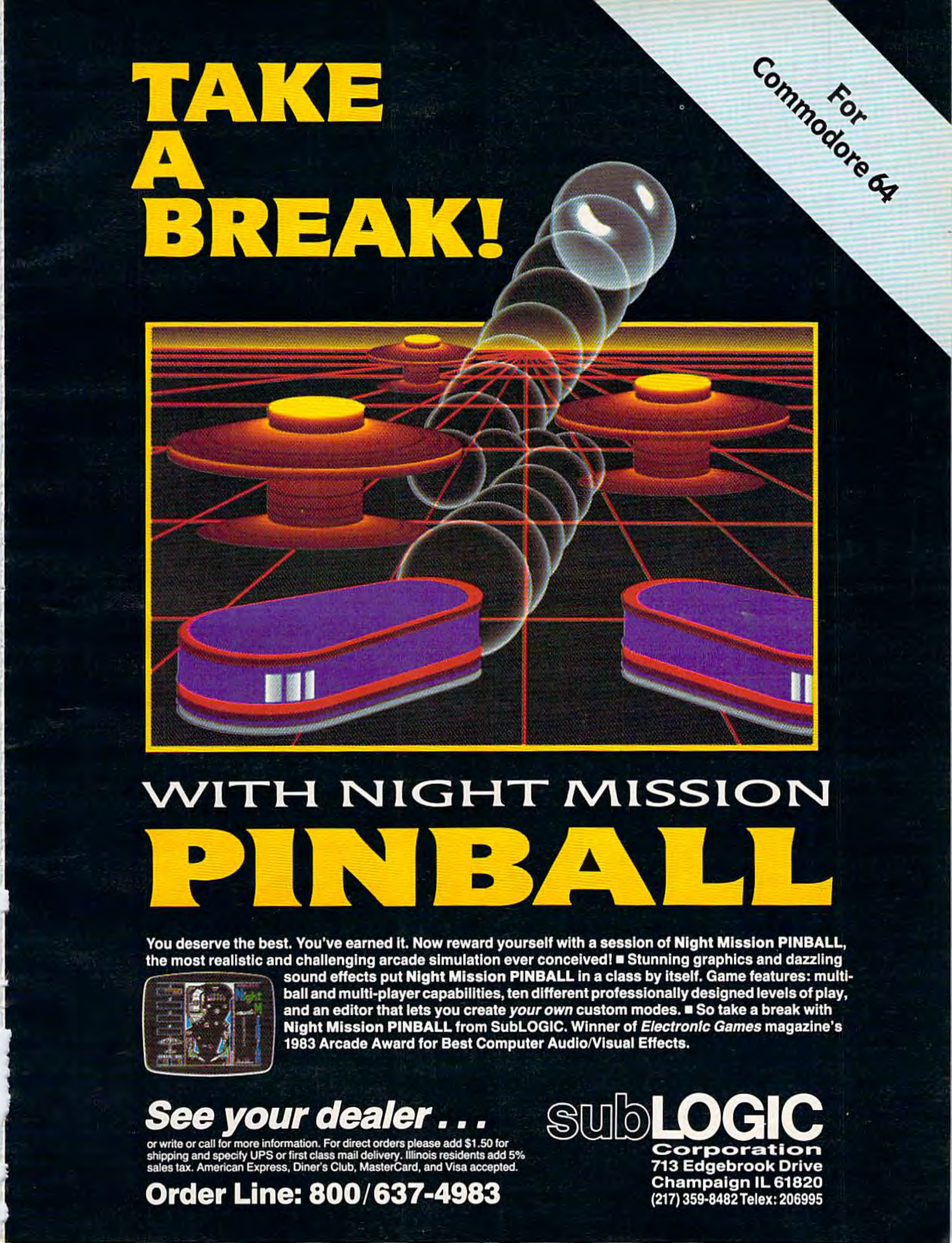


"Binary Castle" is a simple program that lets you practice recognizing four-bit binary numbers (VIC version).

Maybe you think it's adding 2 + 2, but all it's doing is setting little switches ON and OFF according to certain patterns.

We input a sequence of PATTERNS into the computer's memory – that's called programming – and when the program is run, these patterns are taken one-by-one and the computer's thousands of electronic switches are set according to these patterns. I like to think of it as little ELFS (ELectronic FingerS) setting those switches, but hard-headed number types insist it's *binary*.

"Here we go!" I can hear the groans. "Now he's gonna explain binary with rows of light bulbs turned on and off."


For Commodore 64
TAKE A BREAK!
WITH NIGHT MISSION
PINBALL
You deserve the best. You've earned it. Now reward yourself with a session of Night Mission PINBALL, the most realistic and challenging arcade simulation ever conceived! ■ Stunning graphics and dazzling sound effects put Night Mission PINBALL in a class by itself. Game features: multi-ball and multi-player capabilities, ten different professionally designed levels of play, and an editor that lets you create your own custom modes. ■ So take a break with Night Mission PINBALL from SubLOGIC. Winner of Electronic Games magazine's 1983 Arcade Award for Best Computer Audio/Visual Effects.
See your dealer . . .
or write or call for more information. For direct orders please add $1.50 for shipping and specify UPS or first class mail delivery. Illinois residents add 5% sales tax. American Express, Diner's Club, MasterCard, and Visa accepted.
Order Line: 800/637-4983
subLOGIC
Corporation
713 Edgebrook Drive
Champaign IL 61820
(217) 359-8482 Telex: 206995

Well, yes. I'm going to explain binary, and an easy way to learn binary numbers up to 15 – almost all the binary you'll ever need to know. But forget the light bulbs. All you get from turning a bunch of dumb light bulbs on and off is a lot of heat. Maybe light bulbs explain binary, but they don't explain computers.

Because a computer doesn't understand binary – or octal, or decimal, or hexadecimal – or any kind of numbers. But let's not split hairs. Whether a computer uses binary, or *we* use binary to tell the ELFS how to set a computer's switches, it amounts to the same thing. And while you don't have to know what's going on inside a computer to run one – any more than you need to know what's going on inside a clock to tell time – it's the binary manipulation of those switch-patterns that puts the magic into your computing. Let me repeat that.

Binary puts magic into your computing.

Binary also comes in handy if you decide to learn machine language someday, or even to understand how PEEK and POKE work in BASIC.

Position Is Everything In Life

Binary is easier if you take it a few bits at a time. Ones are the only digits that have any value in binary, so 1 equals one and 0 equals nothing. Not that zero just sits there and does nothing. The invention of zero was probably the greatest invention of all time, bigger than the wheel. A 1 floating around all by itself could be just a 1, or a 10, or 100, or 1000.

There's a difference, right? Zero acts as a "place holder" to show where we've positioned the 1. And that's important because a binary 1 *doubles* in value every move to the left – rather than multiplying by 10, as in our familiar decimal system. That's why, in binary, 10 equals 2, 100 equals 4, and 1000 equals 8.

We take the place system so much for granted in decimal (where every digit increases 10 times in value as it moves left) that we're liable to forget that when we go to another number system. So let's repeat:

Binary values double as we move to the left.

Study the list below for a moment to see the pattern:

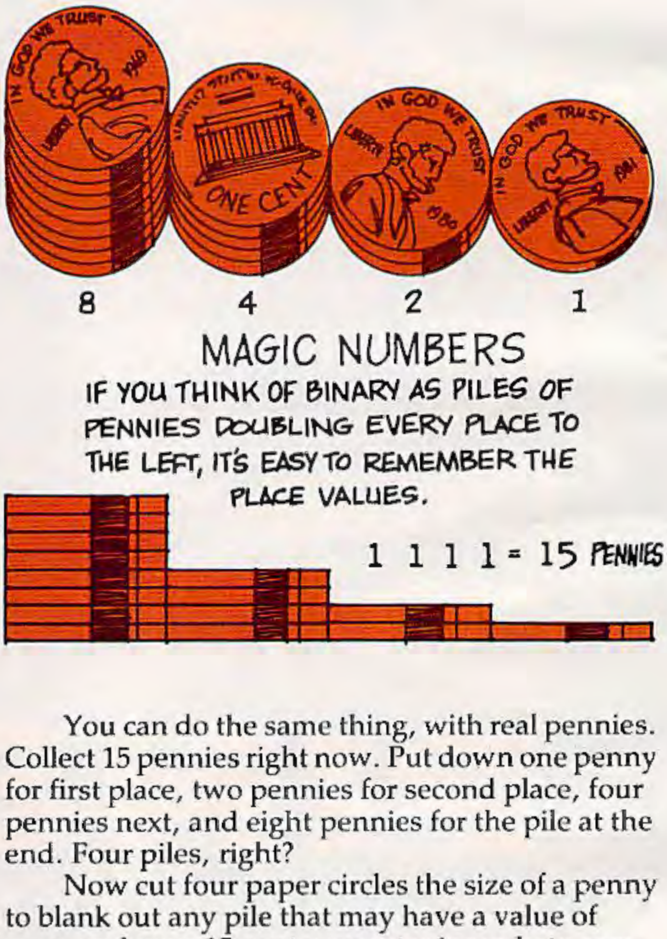
Binary = Decimal	
1 = 1	
10 = 2	(2*1)
100 = 4	(2*2)
1000 = 8	(2*4)
1010 = 10	(8 + 2)
1100 = 12	(8 + 4)

Those are the even numbers (except for the 1) and here are the odd numbers:

Binary = Decimal	
11 = 3	(2 + 1)
101 = 5	(4 + 1)
111 = 7	(4 + 3)
1001 = 9	(8 + 1)
1011 = 11	(8 + 3)
1101 = 13	(8 + 4 + 1)
1111 = 15	(8 + 4 + 3)

The 15-Cent Computer

I don't know about you, but my eyes glaze over when I look at tables of numbers. Words I can read, but numbers don't mean a thing unless I can attach something solid and meaningful to them. So what's more solid than money? Here's how piles of pennies show binary place values:



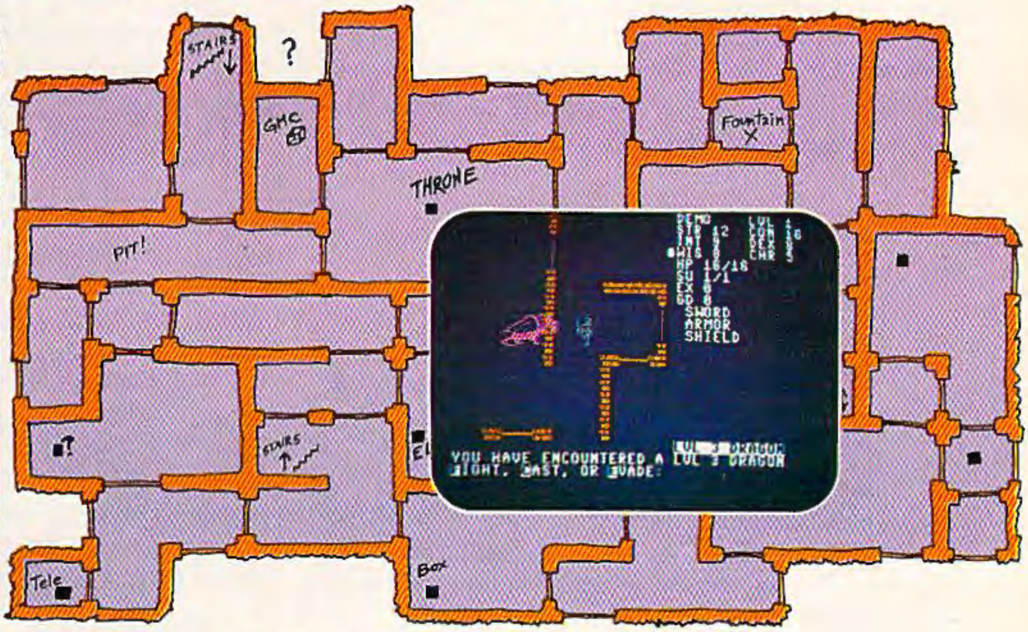
You can do the same thing, with real pennies. Collect 15 pennies right now. Put down one penny for first place, two pennies for second place, four pennies next, and eight pennies for the pile at the end. Four piles, right?

Now cut four paper circles the size of a penny to blank out any pile that may have a value of zero, and your 15-cent computer is ready to compute. (I glued my stacks of pennies together with white glue so that I wouldn't have to count out 8, 4, 2, and 1 each time, and pasted the paper circles on the bottoms so that I could set any pile to zero just by turning it upside down – but you can skip this refinement until you see whether you like the idea.)

Programming The 15-Cent Computer

Two seems to be the binary number most difficult

Be Amazed!



Telengard: How low can you go?

We've created a subterranean monster. Fifty stories low.

That's the number of levels in the TELENGARD dungeon.

Each labyrinthine level holds hundreds of dark chambers and tomb-like corridors for the mighty adventurer to explore. It goes without saying that a shifting collection of hideous monsters with unpredictable behavior patterns can make life in the TELENGARD maze quite interesting—and frequently quite short!

Using wits, magic and true grit, your character delves deeper and deeper into the depths of TELENGARD in this realtime fantasy role-playing game. Available on cassette for Commodore® 64, Atari® Home Computers (40K), TRS-80® Mods. I/III (32K) and PET® 2001 (32K) for a ghoulish $23.00. 48K diskettes for Apple® II, Atari®, Commodore® 64 and TRS-80® available also, for $28.00.

AND FOR THOSE WHO DON'T DIG UNDERGROUND GAMES . . . There's **B-1 NUCLEAR BOMBER**, a nail-biting solitaire simulation of a manned B-1 on a mission over the Soviet Union. Your plane is equipped with six Phoenix Missiles, a one megaton warhead and orders to retaliate! Cassette for Commodore 64, Atari Home Computers (32K), TI99/4 & 4A (16K), VIC-20 (16K), Timex/Sinclair 1000 (16K), and TRS-80 Mods. I/III (16K) are available for an explosive $16.00. Diskette versions for Apple (48K), TRS-80 (32K), Atari (24K) and IBM (48K) just $21.00.

NUKEWAR: Defend your country by massive espionage efforts, or by building jet fighter bombers, missiles, submarines and ABM's. Your cold and calculating computer will choose its own strategy! Cassette for Commodore 64, VIC-20 (16K), TRS-80 Mods. I/III (16K) and Atari Home Computers just $16.00.

FOOTBALL STRATEGY: Animated action on a scrolling field. A head-to-head challenge or solitaire as you select the best offensive or defensive plays in response to your opponent. Commodore 64, Atari Home Computers (32K) and TRS-80 Models I/III/IV (16K) cassette for $16.00. Atari (32K), IBM (64K) and TRS-80 Models I/III/IV (32K) diskettes available at $21.00.

T.G.I.F.: Thank Goodness It's Friday! Avalon Hill's new party game for one to four players recreating an often-not-so-typical week in the lives of the working class. Half the fun is just making it from Monday to Sunday. Commodore 64, Atari Home Computers (40K) cassette for a meager $20.00. Atari diskette (48K) for $25.00.

AVAILABLE WHEREVER GOOD COMPUTER GAMES ARE SOLD or call Toll-Free: 1 (800) 638-9292 for the name of a store near you. Ask for Operator C.

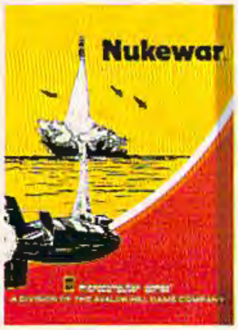
microcomputer games®

a Division of the

Avalon Hill Game Company

4517 Harford Road, Baltimore, MD 21214 U.S.A. Earth

® Trademarks of Tandy Corp., Apple Computers, Commodore Business Machines, Warner Communications and International Business Machines.



AH QUALITY

for people to grasp (because it looks like our decimal 10). So let's make that our first "program."

Zero the 8- and 4-penny piles and the single penny with the blank paper circles (if you've got the paper circles pasted on, just turn the piles over so that the blank side is up). This leaves the second-place pile as is, the only pile that has a value. Your pennies should now look like this:

Now pick up the second-place pile and count the pennies. Two of them, right? If you think of the second binary place as always having a value of two, it's easy to visualize 10 (0010 with "leading zeroes") as a two!

Practice some more binary numbers, making sure you keep the 8-pile, the 4-pile, the 2-pile, and the single penny in that order. If a pile has a value, that value is added to any other values. So, for instance, 0000 – all zeroes – would be 0 + 0 + 0 + 0, or zero; 0001 would be 0 + 0 + 0 + 1 = 1; 0011 would be 0 + 0 + 2 + 1 = 3, and so on. With every pile showing a value (1111), the number would be 8 + 4 + 2 + 1 = 15.

Here are all the binary numbers to 15:

0000 = 0	**0101 = 5**	**1010 = 10**
0001 = 1	**0110 = 6**	**1011 = 11**
0010 = 2	**0111 = 7**	**1100 = 12**
0011 = 3	**1000 = 8**	**1101 = 13**
0100 = 4	**1001 = 9**	**1110 = 14**
		1111 = 15

The 15-cent computer is OK for binary up to 15, but what about the really big numbers? You won't need them. The VIC and the 64 automatically process decimal into binary. For *bit masking* and other tricky maneuvers, it's nice to know the binary for 255, 128, 64, 32, and 16, but we'll get into that next month. I'll also show you an easy way of converting any decimal number to binary.

For now, though, practice your 4-bit binary so that you'll be able to write or instantly recognize any binary number from 0 to 15.

Putting The Hex On Binary

Hexadecimal, or *hex* as it's usually called, is that funny combination of numbers and letters with a dollar sign, like $F2 for instance. The dollar sign is here to tell you that what follows is a hex number and not the name of a plane or some creature from outer space, and the letters are there because hex uses A, B, C, D, E, and F for the values 10, 11, 12, 13, 14, and 15. (In hex, $10 = 16.)

Hang on now, because we're coming down the home stretch.

Hex is shorthand for binary.

I used to think the people who design computers dream up all those crazy number systems just to make it hard for us nincompoops. Good for your character, you might say. But hex saves you from going completely nuts trying to remember all those binary 1's and 0's when you're doing machine language because hex is shorthand for binary. Another beautiful thing about hex is that every four bits of binary is a *direct* conversion to hex. And that's the reason for keeping our binary to the 15-and-under limit for now.

Take decimal 65535, for instance, the highest memory address on your 64 or VIC. 65535 is 1111111111111111 in binary, but $FFFF in hex. Much easier to remember, right? Let's nibble on that for a bit (a *nybble* is four bits, half a *byte*). First we separate all those 1's into easier-to-handle 4-bit nybbles:

1111	**1111**	**1111**	**1111**
F	**F**	**F**	**F**

Then, presto! All we have to do is put the hex values for each nybble directly underneath, and that's our conversion: 65535 (decimal) = $FFFF (hex). Hard to believe it's that simple, isn't it?

Sure, the far left nybble's F has a value of 61444, the F beside that has a value of 3840, the F next to that has a value of 240, and the first-place nybble's F has a value of 15 – but the computer doesn't know that. A computer doesn't understand numbers, remember? It just sets switches according to certain binary patterns. When a binary byte goes through the system, our little ELFS (ELectronic FingerS) turn on a switch wherever there's a 1, but when they see a hex F, they turn on four adjacent switches.

Although you can convert binary nybbles to hex, most programmers do it the other way around: they use hex to visualize the binary patterns they want to input. $FF (1111 1111), for instance, is a pattern that turns every switch in the byte on; $F0 (1111 0000) would turn on the four left switches – and turn off the four right switches; $0F (0000 1111) turns off the four left switches and turns on the four right switches.

As you can see, you can do a lot of tricks if you input directly in hex, but you need a *machine language monitor* for that – such as the *VICMON* or *HES MON* cartridges, or the monitor program in *Machine Language For Beginners* (by Richard Mansfield; COMPUTE! Books). Without a monitor, you'll have to POKE in decimal. But more about that next month when I'll show you how to convert decimal to binary/hex.

* Popular Computing, November, 1982
† Apple Softalk, April, 1982

Everyone's talking about The Home Accountant.™

Is it because it's the #1 bestselling home finance package in the world? Or because it's extremely thorough and powerful and easy to use? Or because it's great for home and business use? Or because it has up to 200 budget categories and handles up to 5 checking accounts?

Yes. But there are a lot more reasons why people buy The Home Accountant. And why you will, too.

Because The Home Accountant can literally save you hours of time. And take the headache out of handling your finances. Whether it's setting up a budget, cataloging your expenses, balancing your checkbooks or handling your credit cards and money market funds. For personal or business use.

The Home Accountant will even print net worth and financial statements. Not to mention being a lifesaver at tax time. Especially when you're able to transfer information onto Continental's The Tax Advantage™ program and figure out what you owe. Quickly.

In short, The Home Accountant is the most effective software program there is for managing your money. And managing it easily.

Stop by your Continental Software dealer today and pick up The Home Accountant. You'll see what everyone's talking about.

The Home Accountant is available for Apple II/IIe, IBM PC/XT, Atari 400/800/1200XL, Osborne,® TRS-80 Models III/4, Commodore 64, Texas Instruments Professional, Zenith Z-100/110, Compaq and KayPro computers. Actual budget capacities will vary with each computer.

For your free 48 page booklet, "Tips For Buying Software," please write Continental Software, 11223 South Hindry Avenue, Los Angeles, CA 90045, 213/417-8031, 213/417-3003.

Continental Software
A Division of Arrays, Inc.

In the meantime, when you get tired of playing with your 15-cent computer, why not try building a binary castle?

Binary Castle

This is a little game you can quickly type into your VIC or 64 and have some fun with while you continue to sharpen up your binary. In this program, square graphics have been substituted for the pennies so that you can use the binary as blocks to build a castle that reaches high into the sky (to the border at the top of the screen).

That's if everything goes smoothly. But if you give even one wrong answer to the Wicked Witch of the Hex when she asks you what binary number the block represents, your castle will collapse and you'll lose all your blocks.

When you run the program, a 4-bit binary block will appear at the bottom of the screen, and you'll be asked what binary number it represents. If you answer – or guess – correctly, a new block will be placed on top, and again you'll be asked what binary number it is.

If you keep answering correctly, your castle will build higher and higher. If you do reach the sky, the game restarts and you can begin building another castle.

So, make one mistake at any time – answer with the wrong number – and the castle is reduced to "bits" with sound effects and lightning flashes. You then have your choice of giving up computing forever or building the castle from the ground up again. (You can always shut the game off until you simmer down, too.)

Incidentally, the blocks build on top of each other the way a part of the computer called the *stack* stores BASIC variables in the VIC and 64. So you'll not only be practicing your binary, you'll also be learning a little about what goes on inside your computer while the program is being run.

See program listings on page 198.

DECK THE HALLS
and your VIC 20
with ARCADE - STYLE games from MICRODIGITAL !
available NOW for Commodore 64
WATCH OUT!
Your mission, should you decide to accept it, is to capture tokens and keep a sharp eye peeled for the gremlins that are chasing you. A fast, action packed maze-and-chase game with high-resolution graphics and excitement-building sound.
SNAKMAN
available NOW for Commodore 64
EXCITING! JOYSTICK-CRUNCHING!
A new generation of grid game with super color graphics and arcade - like sound. You are a lonely little painter who must completely paint increasingly complex grids before the nasty grid chasers catch you! There are 256 different grids to challenge you.
GRIDDER
available NOW for Commodore 64
THRILLING! FANTASTIC!
A space game filled with exciting graphics and life-like sound! You've got to land your galactic fighter craft beyond the defenses of a hostile enemy while lasering and bombing through tunnels, caverns, and a city maze, and outmaneuvering rockets, wiggles, and meteor showers.
SKRAMBLE!
available SOON for Commodore 64
PINBALL
WIZARD
JUST LIKE THE REAL THING!
An action-packed pinball game featuring 1 or 2 players, realistic gravity, flippers, high-resolution graphics, and super pin ball sounds. You pull the plunger, flip the flippers, light the lights, and rack up the points. But watch out, you can TILT!
FREE VIC 20/64 POCKET Reference Guide Send SASE to TECH2
FAST and SMOOTH! Programmed in machine language!
Commodore 64™ and VIC 20™ are registered trademarks of Commodore Business Machines, Inc.
See your local retailer today or call MICRODIGITAL toll free at 1-800-833-7384 or in NY 716-872-0518.
TECH2 SOFTWARE, INC. • P.O. Box 1110 • Webster, NY 14580. Dealer inquiries invited.

Getting Started With A Disk Drive

Part I: The Basics

Charles Brannon, Program Editor

A disk drive can make loading and saving programs much easier and amplify the speed and power of your applications. However, a disk drive is more than just a fast cassette recorder — its greater sophistication can sometimes be confusing. This series explores the advantages of disk-based computing, and will explain everything covered in the *VIC-1540* or *VIC-1541 User's Manual*, plus more.

Maybe you're tired of searching through a box of tapes, then waiting five or ten minutes for a long program to load. Perhaps you're using your computer and tape drive to keep track of an extensive data base, such as a record collection, and you're discovering that you can find an item faster yourself. You may even be plagued by tape errors and yearning for a reliable substitute.

On the other hand, the relatively low price of a Commodore disk drive (maybe more than your computer cost) may seem exorbitant just for higher speed and ease of use. And if you don't mind waiting ten minutes for a game to load, you probably don't need a disk drive. If, however, the tape drive is the weakest link in your system, you'll reap a substantial upgrade from the investment.

In any case, this series will make you aware of the unique advantages and the subtle problems of the Commodore 1540 and 1541 disk drives. We'll give you the information you need to get started, as well as tips to make the most of your drive.

Let's delve beneath the white or tan plastic case of your Commodore drive. Knowing how something works lends you insight when you're stumped.

First, let's look at a typical diskette (we'll use the terms *disk* and *diskette* interchangeably unless there's a need for distinction). Refer to the cutaway view in Figure 1. The disk itself is like a thin, limp (hence "floppy") 45 rpm record, except it's enclosed in a plastic sheath. Floppy disks come in diameters of 8 inch (the original size) and 5¼ inch (called *minifloppies*). Minifloppies are most popular for personal computers. Recently, several companies have introduced *microfloppies*, but so far no one can agree whether to standardize on the 3, 3¼, or 3½ inch size.

Disks themselves are made of flexible mylar coated with the familiar magnetic oxide of recording tape. A great deal of work has gone into improving the floppy disk, but the basic manufacturing process is the same.

The large disk-manufacturing companies (Scotch, Verbatim, BASF, Maxell, and others) stamp out disks from large sheets of coated mylar. All disks, single- and double-sided, single and double density (we'll explain these terms later) are stamped from the same sheet. The manufacturers then test the disks, grade them like eggs, and sort them into the various classes. It's somewhat disturbing to know that the average single side/single density disk (the lowest grade) had to fail several tests to fall into this grade.

However, the much greater demand for lower-priced single side/single density disks re-

quires that some of the higher-grade disks be relabeled to satisfy this demand. Therefore, there is a good chance that your disk will be of higher quality than it is graded on the label.

When you insert the disk in a disk drive and close the door, the drive hub clamps down on the disk's center ring (see photo). Centering is not perfect. A common technique to improve the centering is to insert the disk while the motor is spinning (enter LOAD "$",8).

The disk drive's read/write head, similar to a tape recorder's record/play head, must make good contact with the disk's surface. When the door is closed, a pressure pad bears down on the disk from above and presses the disk against the head. That's why the disk needs to be floppy, so it can flex with the pressure. Since the head is underneath, the drive writes to the underside of a disk. Be aware of this when handling a disk.

The moving head is mounted on a linear track that allows it to slide forward and backward across the surface. This is similar to linear-tracking record turntables. In fact, the record analogy is very useful in understanding how a disk drive works.

If your record player ran as fast as a disk drive (300 revolutions per minute), your favorite albums would sound like a high-pitched squeal. This should give you an appreciation of how fast a disk drive is compared to a cassette tape, which moves at only 1⅞ inch per second. The high rpm's of a disk drive account for much of its speed, but are not the only factor. Equally important is the way a disk drive finds what it's looking for on a disk. Unlike a tape recorder, a disk drive can move its read/write head directly to the location where the data is recorded.

With a tape recorder (for either a stereo or a computer), you must search sequentially through the tape for a certain selection (program or song). In other words, the tape winds past a stationary record/play head, so the head has to wait until the right section of tape arrives. This method, called *sequential access,* is inefficient. You can press Fast Forward to speed up the winding, but finding the right spot is a haphazard process if you're not aided by a tape counter. Even with a tape counter, you will find that the numbers vary widely from recorder to recorder.

On a record player, however, you just lift the tone arm and move it to the desired track. You can move the tone arm directly to any track on the record, skipping over unwanted ones. A floppy disk also has tracks to which the disk drive's read/write head can move directly. This is *random access.*

Disk tracks are somewhat different than record tracks, however. A record really has only one very long track which starts at the rim and spirals toward the center. On the other hand, disk tracks are concentric circles. Each track is further subdivided into arcs called *sectors*. Each sector holds a *block* of information. The sector size varies on different machines, but is 256 bytes on the Commodore 1540/1541 disk drive.

Most disks have 40 tracks, and each track usually has the same number of sectors. An average single density disk can hold roughly 100K (100,000 bytes). This varies according to how many sectors are stored per track. Double density disks squeeze in more sectors per track. This requires a better grade of disk and requires greater reliability.

Inside look at a 1540/1541 disk drive. The small white square bisected by a horizontal line is the read/write head; poised above it is the felt pressure pad. A disk is partially inserted.

Commodore has taken a clever approach to density. The 1540/1541 disk can hold 170K with single density storage. It does it by using a varying number of sectors per track (while using only 35 tracks). The outside tracks of a disk are larger and can proportionally hold more sectors than the inner tracks (see Figure 2). This also solves the problem some drives have when they try to read or write to the inner tracks, where the sectors are crammed together.

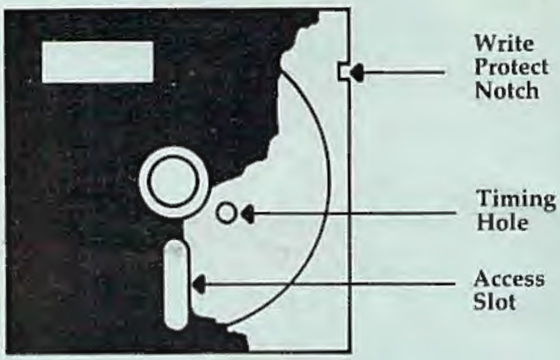
Incidentally, this is a good time to distinguish between *soft* and *hard sectoring*. Hard-sectored disk drives require disks with a series of holes that are read through a timing hole (see Figure 1). The holes tell these drives where each sector is. Soft-sectored drives, however, find each sector by timing how long it takes for the disk to revolve. This also can vary from 300 rpm according to how loosely or tightly the disk fits in the jacket. With this system, the single timing hole suffices (some drives — those for Atari computers, for instance — disregard the timing hole altogether). Drives that use hard sectoring (multiple holes) cannot

use soft-sector disks, but you can use both types on your soft-sectored 1540/1541 drive.

Fortunately, you don't have to remember all this when using the drive, since the disk drive's internal Disk Operating System (DOS, pronounced "doss") takes care of managing sectors. All you do is tell the computer whether you want to read or write information and the name of the *file* involved.

Many application programs do this for you, too. In BASIC, you just give the SAVE or LOAD commands (more on this later). A word processor just asks you for the filename and takes it from there. But you'll still be responsible for keeping track of your disks and following necessary care and maintenance rules.

Figure 1: Cutaway View



You should be careful to protect your disk drive investment. Keep the disks within a reasonable temperature range. To paraphrase one disk manufacturer, "If it's comfortable for you, it's more than comfortable for your disk." Keep your disks away from extreme cold (like your freezer, or a mishap in the winter) and extreme heat (your glove box in the summer, or a heat source such as a radiator or a ventilation slot).

Handle floppy disks carefully. Although disks can sometimes bear being bent in half and still function (if you're lucky), don't take the term "floppy" too literally. Any stress can flake off the magnetic coating and cause read errors.

One more warning is especially important — since disks store information magnetically, keep them away from magnetic fields and radiation. The most common — and hazardous — source of strong magnetic fields and radiation in a home computer system is the TV or monitor. Never rest disks on top of a display device, and don't store or use them within two feet of one.

Magnetism is very sneaky. It's hiding under the case of many appliances in the form of a transformer. Don't store disks near power supplies, refrigerators, printers, etc. Keep all magnets miles away from your equipment. And don't place disks near a telephone. Every time a phone rings, it sends out waves of magnetism. A disk resting under a telephone (besides being uncomfortably compressed) is a sitting duck for a friendly phone call.

Manufacturers tell us to treat disks as if they were eggs. In fact, disks are very resilient to abuse. Data processing people delight in relating favorite horror stories about how an apparently unrecoverable disk was salvaged. Here's my favorite:

I was preparing to load a very useful BASIC utility program when I noticed someone had spilled a soft drink all over this prized, uncopyable disk. The gooey mess had jelled, like glue, and the disk wouldn't even turn inside the jacket. My heart sank, as I realized the disk was irretrievably lost.

My boss came in response to my cries and appraised the situation. He cradled the mangled disk in his hands and declared: "This disk can be saved." I shook my head in knowing disbelief and surrendered the magnetic victim.

I watched my boss as he undertook what I figured was a mission of folly. First, he took a good, working, but blank diskette. He carefully slit the edge of the jacket (see Figure 1) and removed the blank disk from the envelope.

He then did the same with the damaged disk and carefully placed the syrup-covered victim on some paper towels. Discarding the previously removed blank disk, he brought the hopeless disk to the sink. To my surprise, he held it under a stream of water, and washed it with detergent! After carefully rinsing off the suds, he left it to dry on the paper towels.

To make a long story short, he inserted the newly cleaned disk inside the envelope he had earlier readied. A piece of tape sealed the slit.

You guessed it — the disk worked! I inserted it into the disk drive and it loaded perfectly. I was amazed!

To be fair, though, I would have to say we were very lucky.

Why was the recovery of the disk so important? The reason is that you've banked a lot of information on a disk. A disk can hold 170K, so you have a *lot* to lose if that disk crashes (fails).

This is a real threat to a small business. Insurance will pay for loss of equipment caused by fire or theft, but the information stored on the destroyed disk is worth hundreds of times the cost

Into the violent world of video games, Romox™ introduces

Two of the Friendliest, Fastest, Fun-packed Family Games Ever!

TYPO.™ Chased by an evil spaceship, your only chance of escape is through "knowledge." Knowledge of typing and spelling. The faster you type, the better your chance of eluding the enemy. Spelling homework can be fun with this fast-paced game that not only develops word power—but increases your typing speed with every fun-filled hour as you master the typewriter-style keyboard of personal computers. **Cartridges for the Atari® 400/800/1200XL, Commodore Vic 20™, Commodore 64™, Texas Instruments 99/4A.™**

ANTEATER.™ Ant or Anteater—who will be king of the hill? Somewhere along the twisting, turning tunnels of this marvelous maze game the answer will be found. Until then, gamesters of every age will be caught up in the non-stop action and strategy of Anteater, the game that's crawling with arcade excitement. **Cartridges for the Atari® 400/800/1200XL, Commodore Vic 20™, Commodore 64™, Texas Instruments 99/4A.™**

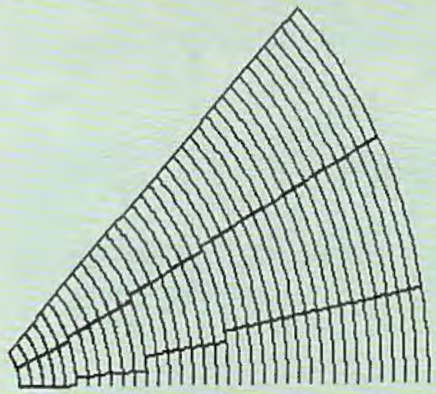
ANTEATER and TYPO are only two of the exciting recreational and educational games from Romox™. See our complete selection at your Romox dealer.

Romox Inc., 476 Vandell Way, Campbell, CA 95008.

Romox is a trademark of Romox Inc. Atari is a trademark of Atari, Inc. VIC 20 and Commodore 64 are trademarks of Commodore Business Machines, Inc. TI 99/4A is a trademark of Texas Instruments.

ROMOX™

Figure 2:
Track/Sector Layout – 1/8 Surface



(Sectors are subdivisions of concentric tracks.)

of the disk. Not only do you lose your investment in expensive commercial software, but you lose the countless person-hours it takes to restore the lost information. Some data processing facilities have had to reenter tens of thousands of questionnaires.

Even if a home computer user loses only his or her games, that's still a catastrophe when you figure that disk-based games can cost at least $30 each. There are some insurance policies that pay claims for lost data, but some disks are irreplaceable (manufacturer out of business, discontinued software, etc.).

With so much potentially priceless information on a disk, we should, indeed, treat it like fragile china.

Luckily, there's one thing you can do to save yourself in case you lose your disk library: make backup copies of all your important disks. Most computer systems have some way of duplicating a disk. If it is your own programming, you can just SAVE the program to several disks. To be really secure, you should store the backups in a different location, perhaps even in a fireproof safe. Many a programmer has breathed a heavy sigh of relief upon discovering that there was indeed a backup copy of that irretrievable program.

Next month: Copy Protection And Piracy

Look at these Features

- Fully screen-oriented
- Horizontal and vertical scrolling
- Terminal mode – never seen before on a wordprocessor
- Supports Commodore disk and cassette handling
- Imbedded commands

Bliztext

Wordprocessor for Commodore 64

BLIZTEXT is a trademark of ELCOMP PUBLISHING, INC.

Commodore-64 and VIC-20 are trademarks of Commodore Business Machines.

Dealer and Distributor inquiries are invited.

BLIZTEXT – SUPER WORDPROCESSOR for the Commodore-64

– ON SALE NOW! –

- Fully screen-oriented, up/down, left and right scrolling – Upper and lower case
- More than 70 commands
- Full I/O compatibility with Commodore peripherals Upper and lower case
- Works with practically every printer on the market, user definable printer control commands
- INCLUDE command allows handling large files on up to 4 diskettes or on cassette.
- Build in terminal software for electronic mail and networking. Telecommunications mode, upload and download, save on disk or cassette.
- Dynamic formatting, Imbedded commands
- Single keystroke for disk directory and error channel
- Program comes on disk or cassette
- Double line spacing, left and right margin justification, centering, page numbering, and practically everything one expects from a good wordprocessor.

AVAILABLE NOW!

Order #4965 $89.00
Manual only (62 pages) $29.95

MACROFIRE
Editor/Assembler for the Commodore-64
ON SALE NOW
AVAILABLE IMMEDIATELY

One outstanding tool, consisting of 3 powerful elements combined into one efficient program!

1.) Fully screen-oriented Editor (more than 70 commands)
2.) Very fast assembler with macro capability
3.) Machine Language Monitor

Assembly can be started from the editor. Translates in 3 passes. More than 1,000 lables, screen oriented/no line numbers, scrolling, includes disk files.
Practically everything the serious machine language programmer needs everyday!

Manual only $19.95
Order #4963 $89.00

THE GREAT BOOK OF GAMES, VOL.I,
by Franz Ende

46 programs for the Commodore 64
Introduction to graphics and sound. How to program your own games. Walking pictures, animation, high resolution graphics, programming tips and tricks, hints and useful subroutines for the beginner and advanced programmer. This book is a MUST for every C-64 owner.
Come and get it – It's yours for only $ 9.95
Order #182 128 pages $9.95
Programs from the book on disk.
Order #4988 $19.95

MORE ON THE SIXTYFOUR, by H.-C. Wagner
How to get the most out of your powerful Commodore 64. Very important subroutines, tricks and hints in machine language for your C-64. How to modify DOS. How to connect a parallel and serial printer. How to design your own terminal program for communication and networking. Dig into I/O for cassette and disk.
Order #183 $9.95
Programs from the book on disk
Order #4989 $19.95

NEW PRODUCTS

Watch out for our new books, software and add-ons to come soon. ON SALE NOW! -- ORDER TODAY!

How to program in 6502 Machine Language on your C-64, by S. Roberts (Introduction)
Order-#184 $12.95

Commodore-64 Tune-up, Vol. I, by S. Roberts
How to expand and customize your C-64.
Order #185 $12.95

Small Business Programs for the Commodore-64
by S. Roberts
How to make money using your C-64. Mailing list, invoice writing, inventory, simple wordprocessing and much more.
Order #186 $12.95

Hardware Add-Ons:

Parallel printer interface KIT	Order #4990 $ 19.95
Direct Connect Modem KIT	Order #4991 Ask f.price
Universal Experimenter Board	Order #4970 $ 9.95
Expansion Board, space for four experimenter boards(board only)	Order #4992 $ 29.95

For your VIC-20
Tricks for VICs $ 9.95
Universal Experimenter board

HOFACKER

PAYMENT: check, money order, VISA, MASTER CARD, Eurocheck, ACCESS, Interbank
Prepaid orders add $3.50 for shipping (USA)
$5.00 handling for C.O.D.
All orders outside USA: add 15 % shipping, California residents add 6.5 % sales tax.

ELCOMP PUBLISHING, INC
53 Redrock Lane
Pomona, CA 91766
Phone: (714) 623 8314
Telex: 29 81 91

HOTWARE

A Look At This Month's Best Sellers And The Software Industry

Kathy Yakal, Editorial Assistant

Commodore 64 Entertainment

This Month		Last Month
1	*Jumpman* (Epyx)	1
2	*Frogger* (Sierra On-Line)	3
3	*Gridrunner* (HesWare)	7
4	*Fort Apocalypse* (Synapse)	—
5	*Temple of Apshai* (Epyx)	4
6	*Sword of Fargoal* (Epyx)	—
7	*Telengard* (Avalon Hill)	—
8	*Supercuda* (Commdata)	—
9	*Pegasus Odyssey* (Commdata)	—
10	*Pakacuda* (Commdata)	—

Commodore 64 Home/Business/Utility

This Month		Last Month
1	*WordPro 3 Plus/64* (Professional)	1
2	*Turtle Graphics* (HesWare)	—
3	*Quick Brown Fox* (Quick Brown Fox)	—
4	*Inventory Management* (Timeworks)	—
5	*Money Manager* (Timeworks)	4
6	*TOTL.Text* (TOTL)	10
7	*Household Finance* (Creative Software)	8
8	*HES Writer* (HesWare)	7

Commodore 64 Educational

This Month		Last Month
1	*Facemaker* (Spinnaker)	2
2	*Kids On Keys* (Spinnaker)	—
3	*Word Race* (Don't Ask)	—
4	*Dungeons of the Algebra Dragons* (Timeworks)	—

VIC-20 Entertainment

This Month		Last Month
1	*Gridrunner* (HesWare)	3
2	*Amok* (UMI)	—
3	*Escape MCP* (Commdata)	—
4	*VIC Men* (Star Tech)	—
5	*Ape Craze* (Commdata)	—
6	*Choplifter* (Creative Software)	1
7	*Apple Panic* (Creative Software)	—

VIC-20 Home/Business/Utility

This Month		Last Month
1	*Quick Brown Fox* (Quick Brown Fox)	8
2	*HES Writer* (HesWare)	2
3	*HES Mon* (HesWare)	3
4	*TOTL.Label* (TOTL)	6
5	*Household Finance* (Creative Software)	1
6	*TOTL Time Manager* (TOTL)	—

VIC-20 Educational

This Month		Last Month
1	*Primary Math Tutor* (Commdata)	—
2	*Touch Typing Tutor* (Taylormade)	2
3	*Square Pairs* (Scholastic)	—
4	*English Invaders* (Commdata)	—

Continued on page 56.

THIS PRINTER COST JUST $119.95*
Introducing the new Alphacom 42. High quality dot matrix printer for Commodore and Atari home computers.
o Print program listings, charts graphics, letters - most anything you create with your computer.
o Full 40-column width.
o Ultra quiet operation.
o Fast -- two lines per second.
o No maintenance -- 90 day limited warranty.
o Complete with paper and instructions.
For the name of the dealer nearest you call toll free 800-538-7047 (in California 800-325-4395).
*Intelligent interface cable for Commodore or Atari sells separately.
Alphacom™

As Commodore hardware prices continue to drop, the software market spreads out even more. New owners are discovering old favorites that had begun to lose their positions on our HOTWARE list. This month, we see a resurgence in popularity for some of these programs, and some surprise debuts by others.

Commodore 64 HOTWARE

Epyx continues to maintain its presence in the entertainment category. *Jumpman* claims the No. 1 position for the third month in a row; *Temple of Apshai* and *Sword of Fargoal* take fifth and sixth. Though it did not make the list, *Upper Reaches of Apshai* sales reportedly picked up again. HesWare's *Gridrunner,* after falling off the list two months ago, is back up in the No. 3 spot. And Commdata returns with three games in positions eight, nine, and ten: *Supercuda, Pegasus Odyssey,* and *Pakacuda.*

Appearing for the first time are Avalon Hill's *Telengard,* a multilevel adventure game, and *Fort Apocalypse,* by Synapse, an underground helicopter rescue adventure.

Four of the eight best sellers in the home/business/utility category are word processors. Holding the No. 1 position for the third month in a row is Professional Software's *WordPro 3 Plus/64. Quick Brown Fox* has appeared on the list before in the VIC-20 business category, but appears this month for the first time in the No. 3 position for the Commodore 64. *TOTL.Text* and *HES Writer* continue to fare well.

Turtle Graphics, which came in third on the September list and dropped off last month, is back again — this time in the No. 2 spot.

Spinnaker again claims the top two positions in educational software with *Facemaker,* last month's second-place finisher, and a new entry, *Kids On Keys.* Debuting in third and fourth place are *Word Race,* by Don't Ask Software, and *Dungeons of the Algebra Dragons,* from Timeworks.

The educational software market is beginning to spread out as lots of new programs are being introduced and tested by consumers. It's difficult to dominate in a situation like this. It will be interesting to see if the leaders can continue to hold their positions.

VIC-20 HOTWARE

Creative Software's *Choplifter,* which has held the No. 1 position in the entertainment category for the last three months, has finally been knocked out of first place by HesWare's *Gridrunner,* which has also been on the list since HOTWARE premiered in August. Commdata, which made a comeback on the Commodore 64 list, takes third and fifth place with *Escape MCP* and *Ape Craze.* Star Tech debuts with *VIC Men; Amok,* by UMI, and *Apple Panic,* by Creative Software, make return appearances.

Quick Brown Fox regains its No. 1 spot in the home/business/utility category after disappearing from September's list. The remaining positions, with the exception of Creative Software's *Household Finance* in fifth, belong to HesWare and TOTL Software. TOTL's *Time Manager* appears for the first time.

Commdata and Scholastic are fast becoming contenders in the VIC-20 educational category. Commdata takes first and fourth positions with *Primary Math Tutor* and *English Invaders.* Scholastic debuts with *Square Pairs* in third, and Taylormade's *Touch Typing Tutor* moves up a notch to claim second.

Honorable Mention: Terminal Software

We don't have a specific program to rank in this area, but telecommunications is fast becoming a common use for home computers. Several sources mentioned to us this month that terminal software packages are very popular. None has taken the lead so far.

The Changing Marketplace

Our HOTWARE sources are valuable to us in two ways. First, the figures. In order to produce a credible, balanced list, we get monthly unit sales figures that range from ten units a month to thousands. Of course, the greater the monthly volume, the more weight they carry in our calculations.

But this is not to say that our sources who do not move a great deal of software every month are not as important. What we learn from them is how consumers feel about what is going on. Distributors generally do not spend a lot of time talking to individual users. But owners of small computer retail stores do — they know what frustrates, aggravates, and delights their customers. And they must respond to their customers' needs quickly in this market.

Our sources who manage small computer retail stores are responding to the changing demands in several different ways. Some have retreated from the software business entirely in the face of discount competition from mass merchandisers. Some have dropped only their lower-end hardware and software, and returned to concentrating on business and higher-end personal systems. One source moved her business to another part of the country to see if the market was any more receptive there. And one source in the Midwest, who got the jump on the low-end market in her area, is moving into larger quarters and offering both sales and service.

Commodore® owners:
"THE FUTURE IS HERE . . ."

Dealer And Distributor Inquiries Invited

Will your printer interface pass the Commodore® printer test? We don't think so!! Ours will.

The CONNECTION™ is truly the ultimate parallel interface for the VIC20™/COMMODORE 64™. This fully intelligent interface plugs into the disk (serial) socket just like the standard printer and you can easily assign it any device number. It will provide virtually TOTAL EMULATION of the Commodore® printer including all standard graphic characters (normal or inverse), column tabbing, dot tabbing, graphic repeat, dot addressable graphics, cursor up/down mode, and more. It responds to all of the standard commands (PRINT#, OPEN, CLOSE, etc.) to insure software designed for the Commodore® printer will operate with the CONNECTION™. In the TOTAL TEXT MODE, it will work with virtually EVERY PARALLEL PRINTER with standard Centronics configuration. All this plus:

1) A 2K Printer buffer
2) Full LED Status indicators.
3) Complete Built in self test
4) Printer reset switch
5) Adds Skip over perf, margin set, programmable line length, program list format commands to your printer.
6) No need for extra cost, special tape loader for graphics.
7) All features easily accessed from software
8) ASCII conversion, TOTAL TEXT, EMULATE, and TRANSPARENT Modes

To take full advantage of your printer's special features, please specify the printer type. Available for STAR MICRONICS, STX80, EPSON, OKI, NEC, PROWRITER, BANANA, SEIKOSHA, RITEMAN, GEMINI10X and others. ONLY $119.00 Complete (Additional ROMs are available if you should ever change printers)

MasterCard VISA

NOTE: We solicit hardware and software items for the VIC20 & CBM64. Royalties, license fees, or outright purchases can be negotiated. Commodore & VIC20 are trademarks of Commodore Business Machines.

micro ware DI

1342B RT 23
BUTLER, NJ 07405 201-838-9027

TYMAC SAYS,
"We'll Pay For Your Mistake"*

That's right, by providing you with a $50 credit toward our Connection™ interface. After receiving thousands of calls that asked "BUT WHAT DO WE DO WITH OUR OLD ONE?", we now have the answer. Send in your old interface plus a good reason why you want ours, and we will send you the Connection (LIST $119 less $50 Credit) for $69.00. This is strictly a one time offer and TYMAC CONTROLS CORP may WITHDRAW it at any time. It is only for END USERS that are dissatisfied with their interface and is invalid to all DEALERS and DISTRIBUTORS. Offer good only in the USA until 1/84. Offer void where prohibited or taxed. Connections sent prepaid MASTERCARD, VISA, or COD, plus shipping and handling. Send old boards to TYMAC OFFER, BOX 31, RIVERDALE, NJ 07457

**Payment is in the form of credit toward purchase.*

Dale Disharoon & Jim Bach

The Programmers Behind *Hey Diddle Diddle*

Kathy Yakal, Editorial Assistant

Microcomputers have enormous potential as educational tools. Some schools have been using them for years; home computer owners are beginning to explore the educational possibilities of their machines, too. Here's a look at a very popular educational program – Spinnaker Software's *Hey Diddle Diddle* – and the programmers behind it.

Jim Bach (left) and Dale Disharoon jamming in the backyard.

Up to now, the toughest competition for commercial software has been in the area of games and business programs.

Up to now, that is. In the last few months, companies that had previously ignored the educational market are vying for positions in the educational categories of best-seller lists, and new companies are starting up to fill the gaps in the market.

A good example of the latter is Spinnaker Software Corporation, recently organized to publish entertaining educational software for home use. Spinnaker currently has several programs that are being hailed by many retailers, distributors, and users as the best they've seen yet. Two programmers are responsible for one of Spinnaker's biggest successes, *Hey Diddle Diddle:* Dale Disharoon, 27, and Jim Bach, 17.

Tired Of Teaching

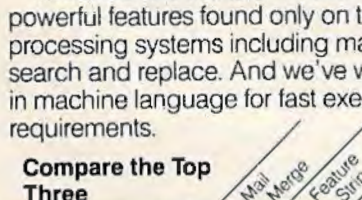
Disharoon's involvement with educational software began about three years ago when he was teaching a combined class of kindergarten, first, and second grade at a small school in Manchester, California. The school had bought an Atari 800 for the students to use; Disharoon bought an Atari 400 to learn programming on his own.

Soon, he started writing computer programs that replaced some of the more repetitive lessons he had to teach every year. To Disharoon, the process of creating a lesson was valuable, but he

If our word processing software is so great, why are we giving it away?

It's our way of introducing you to DATA 20.

As the leader in price/performance peripherals for Commodore 64™ and VIC 20,™ we want you as a loyal customer. So to encourage you to sample our add-ons, we're giving away our WORD MANAGER software when you purchase any DATA 20 DISPLAY MANAGER or VIDEO PAK to expand your Commodore to 80 columns.



What's so great about WORD MANAGER?

The table tells the tale. This DATA 20 exclusive gives you powerful features found only on the most expensive word processing systems including mail merge, block move, search and replace. And we've written our word processing in machine language for fast execution and low memory requirements.

Compare the Top Three	Mail Merge	Feature Strip	Works with Any Printer*	Tape & Disk Capabilities	Full Screen Editing	80-Column Format
WORD MANAGER	•	•	•	•	•	•
WordPro®	—	—	—	—	•	—
Quick Brown Fox™	—	—	•	—	—	•

*Most standard serial or parallel printers require interface, not supplied.

Best of all, WORD MANAGER is exceptionally easy to use. A self-adhesive feature strip for function keys makes most commands one-key simple, eliminating awkward prompts and menus.

What you see is what you print.

With our professional 80-column format, your printed product duplicates screen output precisely. You see centering, left justification and other features as they will print. So you'll save time, save paper, and get the exact results you want!

WORD MANAGER is really something—for nothing! Get it free with your DISPLAY MANAGER or VIDEO PAK from DATA 20. See your dealer or write: DATA 20 CORPORATION, 23011 Moulton Parkway, Suite B10, Laguna Hills, CA 92653.

DATA 20 CORPORATION

Price/Performance Peripherals

grew tired of teaching some of the same lessons time after time.

Disharoon realized he had a knack for creating good educational programs due to his background in teaching and his newly acquired programming skills. He began to sell his creations, first to the Atari Program Exchange (an Atari-managed collection of user-written programs), then to The Learning Company, and, finally, to Spinnaker Software.

Tired Of Teachers

Enter Jim Bach, a teenager from upstate New York who was pretty discouraged with the traditional educational system. So discouraged, in fact, that he dropped out of high school, and then was asked to move out of the house by a father who couldn't understand his son's passion for computers and lack of interest in traditional schooling. Ironically, it was Jim's father who had sparked his interest in computers by buying him an Apple three years before. "Everything I've done has been inspired by my father," says Bach.

Bach's father, incidentally, is Richard Bach, author of the best-selling book *Jonathan Livingston Seagull.*

Jim Bach and Disharoon met when Bach was working for an office supply store in Iowa, writing and reviewing programs. Disharoon had been looking for someone to translate his programs, and Bach was looking to get into the commercial software business.

"I'm in training right now," explains Bach. Disharoon creates the concept and initially programs a new educational game on his Atari, then Bach translates the program for Commodore computers.

"When I was learning about Commodore, I got a Toronto phone book and called every Butterfield in the book until I found [Commodore expert] Jim Butterfield," says Bach. "He was extremely helpful in my learning process."

Recalling A Childhood Game

Disharoon got the idea for *Hey Diddle Diddle* from a school game he had played as a child. Someone would write a familiar poem on a large piece of paper and cut it up so each line was separated. Students would queue up in front of the class, each holding one line of the poem, and the class would rearrange them so the poem was restored to the right order.

"The game actually helped students in three different areas of understanding: reading, logic, and poetry," says Disharoon.

In Disharoon's version, *Hey Diddle Diddle* contains 30 eight-line poems that the player has to rearrange. A line lights up on the screen, and the player uses a joystick to move it to the correct

A screen from Hey Diddle Diddle *on the Commodore 64.*

place. When the verses are finally rearranged, a picture illustrating the poem (drawn by Santa Barbara artist Robin Bush) appears on the screen and a song plays.

Computers As Teachers

Disharoon's most recent project for Spinnaker was *Alphabet Zoo,* introduced at the Summer Consumer Electronics Show held in Chicago in June. "Basically, it helps accomplish what kindergarten and first-grade teachers spend a whole year doing: teaching the alphabet and how to read," says Disharoon.

Yet he does not believe that computers can or should replace human teachers. "When I was teaching, I liked to sing and dance with the kids and help them get to know each other," he says. "There are very important social aspects of education that computers cannot handle."

Bach agrees that computers may be capable of taking over many of the repetitive tasks now performed by teachers. "Before that happens, though, computers have to be a lot less technical and foreign," he says. "They will have to contain a tremendous amount of knowledge and actually be able to teach. Most educational programs that are available now, with the exception of CAI [Computer-Aided Instruction], don't actually teach – they just let you practice something you already know."

Bach says he might have stayed in school longer if educational computer technology was more sophisticated, and if he had been allowed more freedom to learn by using computers.

What's Educational?

Three elements are necessary to make an educational program, according to Disharoon: it must

teach, be fun, and be easy to use. "Some educational software is too game-like and not educational enough. And some is just like a book on a computer — all text." Disharoon tries to achieve a happy medium.

He also tries to write games for children that he himself likes to play. "After you've written a program, you end up playing it hundreds of times while you're refining it," he says.

Disharoon's next project is an adventure game in which the player can program the actual environment of the game. It will use a combination of text and graphics; all the rooms, creatures and other variables can be created and edited by the player. The game will be suitable for ages 12 and up.

Does Disharoon consider this an entertainment or educational game?

"The act of anything where a child creates is educational," he says.

Programmers

Have you written an exciting game? A utility that makes programming easier? An educational program for children? Any kind of useful home application program? If so, *COMPUTE!'s Gazette* wants to hear from you.

Chicken Little

Jeff Wolverton

"Chicken Little" is an action game for the unexpanded VIC-20 and Commodore 64. The author, who is 15, wrote the original program for the VIC. VIC users should be sure to read the section headed, "Special Instructions."

Probably everyone is familiar with the childhood tale of Chicken Little. It is, of course, about the little chicken who was hit on the head by an acorn and then jumped to the conclusion that the sky was falling. (Though that reaction may seem a bit neurotic, the story caught on.)

The game "Chicken Little" is a simulation of the tale. But in this re-creation, the sky really *is* falling. Your goal is to keep too many fragments from tumbling to the ground. You'll have to be nimble on your feet and adept at hurling stones to survive unscathed.

How To Play

In the VIC-20 version, you control Chicken Little with four keys: I=Up, M=Down, J=Left, and K=Right. Because the program scans the keyboard with the PEEK statement instead of GET A$, you can hold down a key for continuous movement.

The Commodore 64 version uses a joystick plugged into port one instead of the keyboard. Chicken Little's vertical movement is restricted to the bottom four lines of the screen, and horizontal movement to the screen's width. The top line is reserved for the score and number of chickens left (you start the game with three chickens). The rest of the screen is filled with 30 randomly placed pieces of sky.

Detailed instructions are included in the program, but I'll summarize them here.

Basically, pieces of sky — blown around by wind — fall down at you randomly. To protect yourself (and to save the world), you must hit them with stones. On the VIC, press the space bar to throw a rock; on the 64, press the joystick fire button. You can also heave rocks at stationary pieces of sky before they start falling. If more than eight pieces of sky get by you and hit the ground, the sky has fallen and the game is over.

Each time you clear the screen of sky fragments, you advance to the next level and the sky falls faster. Also, with each new level, the number of pieces that can fall before the game ends is decreased by one. On the unexpanded VIC, it's possible — if you're very good — to run up a score so large that you'll overflow the memory. But even if you're that good, it would take a long time.

Special Instructions

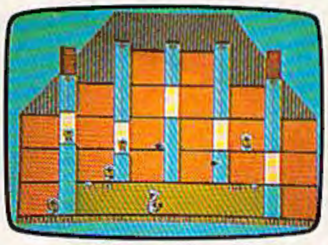
The VIC version of Chicken Little is broken into

First Star Has 4 New Games.

Fernando Herrera, designer of ASTRO CHASE™ and our design team again define "State of the Art." Superior graphics, real-time animations,™ multiple screens, intermissions, arcade-quality sound, innovative gaming, challenge and excitement—we deliver it all!

THE BAD NEWS? You can't play them all at once.

BOING!™
Designed by Alex Leavens & Shirley A. Russell
Atari VCS 2600



BRISTLES™
Starring Peter the Painter
Designed by Fernando Herrera
Atari Home Computers
Commodore Computers

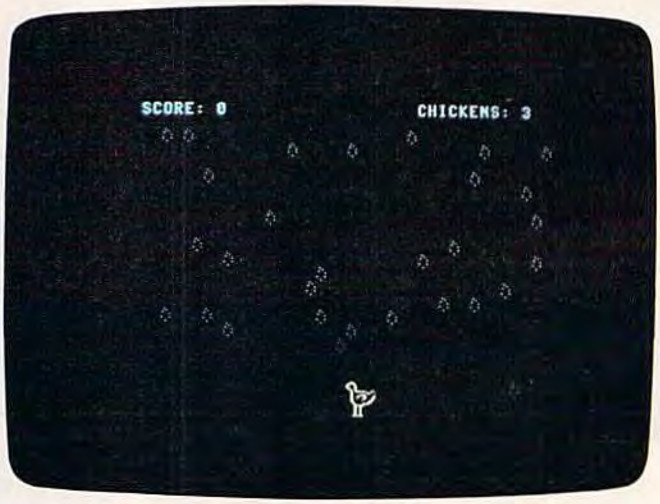
FLIP and FLOP™
Designed by Jim Nangano
Atari Home Computers
Commodore Computers

PANIC BUTTON™
TRS-80 Color Computer by Paul Kanevsky
Vic-20 Home Computer by Wayne Lam

FIRST STAR SOFTWARE INC

"When being first counts...we're number one!"
22 East 41st Street, New York, NY 10017
Distributor and Dealer Inquiries Invited/212 532-4666

The sky is falling in large chunks toward Chicken Little (VIC version).



"Chicken Little" on the Commodore 64.

two parts to save memory. The first part (Program 1) prints instructions and loads a custom character set of 12 characters. If you're using tape, it then automatically loads and runs the next part (Program 2), which contains the actual game.

Tape users: Type Program 1 and SAVE it on cassette. Then type NEW, enter Program 2, and SAVE it on the same tape immediately after Program 1.

Disk users: SAVE both programs on the same disk. RUN Program 1. When the screen says, "Press Play On Tape," press RUN/STOP instead (but *not* RUN/STOP-RESTORE), and LOAD Program 2. Type RUN and the game will start.

Because of the Commodore 64's larger memory, it was possible to combine the 64 version into a single program that requires no special loading.

See program listings on page 182.

New VIC-20* Releases!

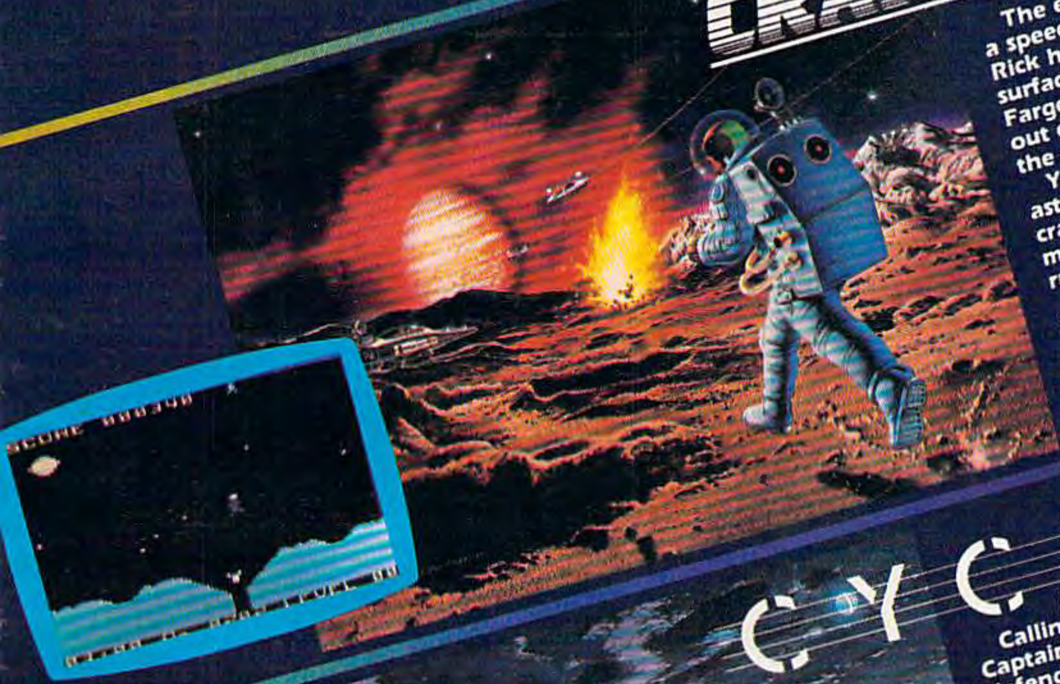
More fun than humans should be allowed.

CRATER RAIDER™

The earth is directly in the path of a speeding asteroid. Fearless Raider Rick has just stepped onto the alien surface with orders from Captain Fargo himself to blast the asteroid out of existance before it smashes the world.

You must guide Rick over the asteroid surface. He must leap over craters. Avoid deadly aliens and meteorites. And do it all without running out of oxygen.

In the end, you may succeed in helping Rick place his bomb. But a fiery death awaits him if you lack the skill and nerve necessary to get him back to his ship before the awesome blast!



CYCLON™

Calling all future fighter pilots. Captain Fargo needs you! If you can defend your ship against the attack of enemy star-cruisers, control of the sky is yours. But beware! Creatures of the night will come at you from all sides . . . firing at you with light missles and white-heat vapor rays.

You can take only a few hits or you'll be destroyed. Only a steady eye and rapid-fire reactions can save you and your ship. The challenge is yours. Defeat is agony.

Created by Alan Pavlish

• Introducing two new red hot releases from the Game Brains™ at Boone: Cyclon™ and Crater Raider!™ Like all Boone games they feature: Rapid action • Super smooth animation • Intense sound and color • Multiple levels of difficulty • And professional Hi-Res graphics. And when you buy a Boone product you get more than just a game. We have **poster offers.** Memberships to Captain Fargo's **Fan Club.** And updates on all new releases • Cyclon™ and Crater Raider™ are at your dealers now! Demand the best. Demand Boone.

BOONE

2900-A Bristol Street, Suite 102, Costa Mesa, California 92626 U.S.A.

Martian Prisoner

Adventure Game For VIC And 64

Alan Poole

"Martian Prisoner" is a mini-adventure game for the unexpanded VIC-20 and Commodore 64. If you've never played an adventure game before, this is a good introduction. Unlike most computer games, text adventures have no graphics and do not require fast reflexes — instead, they test the player's patience and cunning.

Without warning, the Martians have suddenly started a devastating war against Earth. They have captured you and are holding you prisoner in a cell on a Martian space cruiser headed toward Earth. The cruiser also carries a secret weapon that can neutralize all of Earth's defenses. Your task is to destroy the Martian ship and escape in a lifecraft before the Martians can complete their sinister mission.

Like Radio Dramas

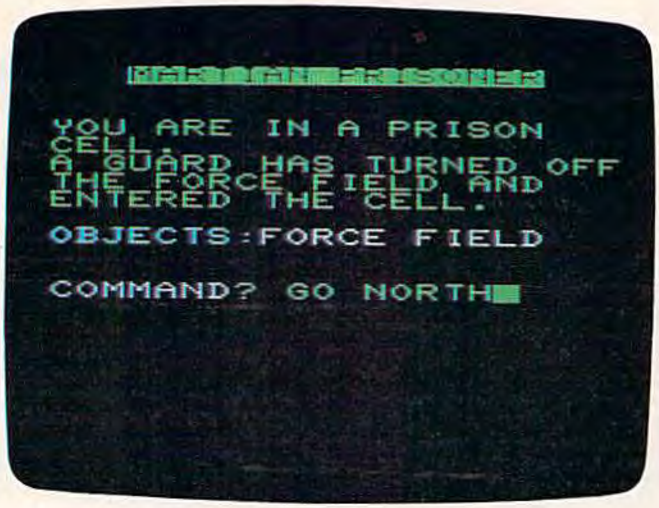
"Martian Prisoner" is a mini-adventure game, about the most that can be squeezed into an unexpanded VIC. Adventure games require you to solve puzzles and explore a simulated world inside the computer. The computer will describe what you see and what happens, and you tell the computer what you want to do. Instead of using screen graphics, adventure games rely on text descriptions and your imagination. It's like the difference between old-time radio dramas and television; despite the visual impact of video, the mind can still imagine a scene more exciting than a camera can picture.

In Martian Prisoner, you start off in the prison cell of the Martian space cruiser. Besides the cell, the cruiser contains several other rooms. It's up to you to explore the rooms and find a way to destroy the ship. In each room, the computer will describe your surroundings and list the objects in

WE INTERRUPT THIS AD TO BRING YOU THIS IMPORTANT PROGRAM...

Now you could add **your** program to the rapidly expanding list of MMG Software with MMG's **Write Your Own Program Contest!** Each month a program will be chosen and developed from the entries and suggestions sent in by **you** and best of all, the winners will receive royalties from the sale of their entry. You don't have to be a computer whiz to enter, so see your local retailer who carries ATARI products for details. FINAL FLIGHT is only one of the many exciting and useful programs for the ATARI system now available from MMG Micro Software that are designed with you in mind. At MMG, we believe that people are as important as the programs they use.

MMG micro software
P.O. Box 131 Marlboro, NJ 07746
(201) 431-3472



"Martian Prisoner" is a clever text adventure game that fits in an unexpanded VIC-20.

the room. The computer then waits for you to type a command, consisting of one or two words.

For example, you could type GO NORTH to move north. If there is a book in the room, you would type GET BOOK to pick it up. Type INVENTORY at any time to see a list of the objects you are carrying. All commands and nouns can be abbreviated to the first three letters. You can list your INVENTORY by typing INV, for instance.

Although Martian Prisoner is a short adventure game, you must solve several puzzles to win. It's a good way to prepare for the more elaborate adventure games available commercially for Commodore 64s and VIC-20s with expanded memory.

(Editor's Note: If, after hours of play, you're still stumped and desperate for the solution, write to COMPUTE!'s Gazette. Depending on the demand, we may publish a blueprint of the Martian cruiser and the solution to the game in a future issue.)

Starting a game of "Martian Prisoner" on the Commodore 64.

VIC Emulator For 64?

Well, sort of. Program 2 allows one version of Martian Prisoner to work on both the VIC-20 and Commodore 64. To adjust the 64's 40-column screen to simulate a 22-column VIC, 64 users should type in Program 2 in addition to the game listing. *VIC users should not use Program 2.* Program 2 creates a machine language program which forces the 64 to PRINT within a 22-column format. It will not support any other VIC functions, however. Martian Prisoner will work on the 64 without Program 2, but words will break in strange places and the text will be harder to read.

Be sure to save Program 2 on tape or disk before running it. When you type RUN, the routine activates itself. Should you need to reactivate it at any time, enter SYS 828. (By the way, this program might work on other all-text VIC programs, too.)

Tape Copies

If you would like to save the time of typing in Martian Prisoner, send a blank cassette, self-addressed mailer with postage, and $3 to:

Alan Poole
4728 King Road
Loomis, CA 95650

See program listings on page 185.

Get the players' games from CDS!

WITCH WAY

Can Wally fight his way to the top and save his true love??? The way looks simple enough, but . . . uh oh . . . LOOK OUT FOR THE WITCH!!!

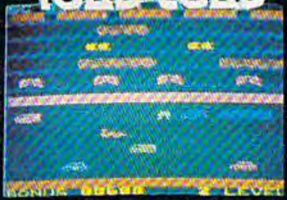
It's a multi—level, multi-screen boggler! Super game power requires an 8K expander!

BUG SPREE

Strap on your lasers, squad, because this one is going to stretch you to your limits!

Changing screens and fast-paced, non-stop action keep you long on fun and short on fingernails!

ROAD TOAD



Can our hero make it across the freeway without getting that run-down feeling?

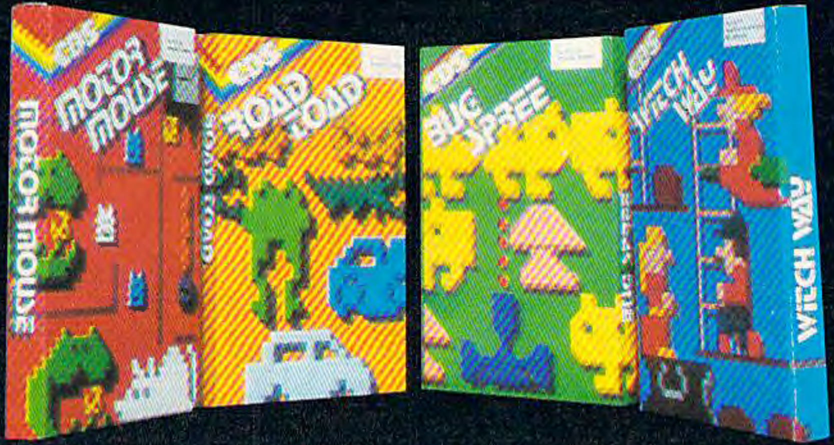
And if he does . . . does he know he can't swim?!?!

It's the thrill of victory and watch out for the alligators!

MOTOR MOUSE

Keep your eyes open — there's a tasty treasure in cheeses just ahead! But . . . there's hungry cats around every corner!

One wrong move, and you're hickory dickory dead!



The CDS winners for the Commodore VIC 20. (Also some for Commodore 64) Suggested retail $29.95 US (cassette). Ask for them at your Commodore store. Or write to us. Dealer inquiries also welcomed.

Commercial Data Systems Ltd.,
730 Eastview Avenue, Regina, Canada, S4N 0A2
(306) 525-3386

FRED D'IGNAZIO,
ASSOCIATE EDITOR

There's A Creature In My Computer!

The Lobsters Under My Bed

When I was a kid I used to go to sleep at night with my hands pulled up inside the sleeves of my pajamas, and my feet tucked inside two layers of socks and a pair of slippers. I did this to hide my fingers and toes from the lobsters that lived under my bed.

These lobsters weren't just average creatures. First, they didn't need to live in water. Instead, they could somehow survive under my bed — along with lint, dust, dirty clothes, copies of *Mad* magazine, science fiction books, and potato chip crumbs.

Second, if they got hungry, they didn't look for regular lobster food. Instead, they liked to munch on crumpled, smelly socks. (There were lots of those under my bed.) But their favorite food was fingers and toes — *dirty* fingers and *dirty* toes.

I went to bed at night convinced that lobsters really did live under my bed. I was afraid that if I fell asleep and accidentally let my hand or foot slip over the side of the bed, one of the lobsters would leap out, pinch it off, gobble it up, and disappear back under the bed.

The lobsters had never been known to attack clean fingers and clean toes. But I never considered taking a bath. Instead, I bundled up my toes and fingers, and slept in bed all scrunched up like a sunburnt spider. If a lobster wanted to make a meal out of me, it was going to have to work for it.

I shared my bedroom with several lobsters. But we weren't alone. There was also a nightmarish creature who lived underneath my

dresser. He would come out from under the dresser when my mother turned out the hall light. He always hid in the shadows. In fact, he *was* a shadow. Real slithery, dark, and tricky. He was all body. No head.

And then there was the creature that lived in my closet behind my dress shirts and Sunday school pants. I called him the Closet Beast. He was one of those shy creatures. He only came out at night when I wanted nothing to do with him. During the day he probably killed time pretending to look like a bow tie or the pair of brown dress shoes I hated.

Between me and all the creatures, the bedroom was crowded. I wished that some of the bedroom creatures would move out. But if they had they would have bumped into the creatures in the other parts of the house. The worst of these was the Ghoul who lived in the cellar, underneath the stairs.

I hoped and prayed I would never meet the cellar Ghoul. But one night I came very close. It all happened because I was a sleepwalker. I was so bad that my mother had to bolt all the windows each night before she went to bed. She was afraid I might climb out one of them and try to sleepwalk on the two-foot ledge that rimmed the roof.

Thanks to my mother I never did any sleepwalking on the roof. But I did sleepwalk a lot inside the house. And I sometimes ended up in some pretty strange places.

One night I woke up and immediately knew something wasn't right. I had my pillow and was wrapped up in my blanket like a mummy, but I wasn't in my bedroom any longer. I was someplace else, someplace very, very dark. And damp. And moldy smelling.

I rubbed my fingers on something hard underneath me. I realized I wasn't in bed. I was on a dusty concrete floor — the basement floor. And I was right next to the stairs where the Ghoul lived.

As dark as the basement was, the space under the stairs was even darker. I couldn't see anything, but I could sense that I was not alone. Something was there with me. And it was coming closer.

I screamed. I screamed again. And again.

I woke up the whole house with my screaming. Moments later, the basement light came on. My parents came flying down the stairs and found me huddled under my blanket, wailing like a ninny.

When they dug my head out of the covers, I pointed toward the stairs. My parents investigated. They didn't catch the Ghoul. But, they did find, hiding under the stairs, a very scared kitty cat.

There's A Creature In My Computer!

I used to see creatures in every shadow or dark corner of my life. I saw so many creatures because I had a crazy imagination.

How about you? Do you have a crazy imagination like mine? Do you see ghosts in wisps of smoke? Do you see sleeping giants inside craggy mountains? Do you see fang-toothed monsters staring up at you out of gutters and hollow stumps of trees? Have you seen the skinny creature who lives inside your medicine cabinet — the one that feasts on stale toothpaste?

With a little imagination you can see creatures everywhere. And, with a little imagination, you can create a creature inside your computer. The creature (he, she, or *it*) might even be living there now. You just have to bring it to life.

The Ghosts In The Machine

Your computer is a perfect place for a creature to live. After all, it's already full of ghosts. The ghosts are other people's programs.

Some people think that programs are just abstract lists full of information and commands. These people are wrong. A program is — or can be — much, much more.

Every program that is written has a personality. Most computer programs written in the past had dull personalities. But they don't have to be dull.

Where does a program's personality come from? It comes from its creator, the person who thought it up and typed the commands into the computer.

The program is a reflection of its creator's imagination.

If the person has a dumpy, dull sort of imagination, then the program will be dull. It might have the personality of a stuffed shirt or toad. Most business programs have toady personalities.

On the other hand, if the person's imagination is creative, weird, and funny, then the program will be creative, weird, and funny, too. (Does this remind you of a few game programs you have played?)

Programs are the ghosts inside your computer. So why not turn them into real ghosts, goblins, ogres, zombies, dragons, and other creatures? You can take the creatures that live inside your imagination and load them into your computer. To create the creatures you just write a program. To bring them to life you just type RUN.

Turn On Your Imagination

Warning: If your imagination is having a bad day, you'd better stop here and wait. The creature we're going to create this month is 99 percent imagination and only 1 percent program. The creature is simple, but it can still seem real — *if* you use your imagination.

A Simple Creature

Turn on your computer and type:

20 PRINT "GRRRR!!" [Press the RETURN key.]

You have just created a creature inside your computer. You don't know what it looks like. You don't know if it wears a ski cap and orange polka-dotted socks, or how many warts are on its nose. But you do know two things: it's there and it's not very friendly.

To see if I'm right, type RUN (and press RETURN). What does the creature do? It says:

GRRRR!!

Not too friendly, is it?

What happens if you add a new line to the creature's program? For example, type:

30 GOTO 20

Now type RUN. What happens? This time you see:

```
GRRRR!!
GRRRR!!
GRRRR!!
GRRRR!!
GRRRR!!
GRRRR!!
GRRRR!!
GRRRR!!
```

Now you've created a creature that is *really* unfriendly! (To stop the creature from growling, press the RUN/STOP key.)

So far, the only way to get your creature's attention is to type RUN. But you might want to say something to the creature. To do this you have to teach the creature to *listen*. To make it listen, type:

10 INPUT A$

Change line 30 to say GOTO 10. Your whole program now looks like this:

```
10 INPUT A$
20 PRINT "GRRRR!!"
30 GOTO 10
```

Type RUN.

The program begins running, and the creature wakes up. He is looking at you. He is waiting for you to say something. (The computer has printed a "?" on the display screen. Imagine that the creature is sprawled in a dungeon inside the computer. He is just waking up. He looks dazed, and has a big "?" over his head.)

This is your first chance to say something to the creature. In fact, he won't make a move until you say something.

But what do you say to a creature?

You can try insulting him by saying something like:

YOUR FEET SMELL!

Or you might try giving him a command like:

DON'T EAT ME!

Or, you can try to be friendly and ask the creature a question, such as:

DO YOU LIKE PIZZA?

Think up a message, type the message, then type RETURN. What is the creature's answer? He says:

GRRRR!!

He says "GRRRR!!" because it's the only thing he knows how to say. He's a very dumb creature. No matter what you tell him, he always growls. He's a real grump.

To make him say something else, you have to teach him. What sort of new things can you teach your creature to say? What sort of things can you teach your creature to do?

Next Time: New Creatures

This time we created a very simple creature. Next time we'll see how we can create a creature that surprises you. He'll make scary creature sounds. And he'll have a creature face.

I'll help you build creatures and turn them loose on other members of your family. But I'd really like to see what creature you can come up with on your own.

Write a short program and make a creature. Then, no matter how crazy the creature is, send it to me. Send it to:

Fred D'Ignazio
2117 Carter Road, SW
Roanoke, VA 24015

Dream up strange, funny, and unusual creatures, then turn them into programs and send them to me. I'll print the best programs in this column.

Be like a magician pulling rabbits out of a hat. Pull the creatures out of your imagination. Then pop them in the computer and bring them to life.

You can send me any kind of creature at all.

Except for just one kind.

Don't send any lobsters.

Munchmath

Gerald R. Anderson

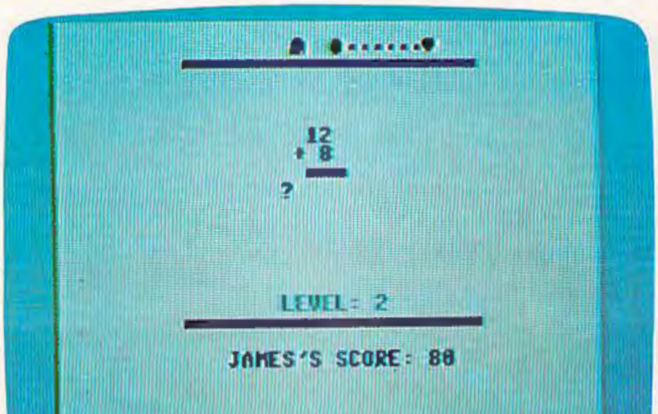
"Munchmath" is an above-average math drill program that entertains as it teaches. Because of its multiple difficulty levels, it is suitable for a wide range of ages. There are versions for the unexpanded VIC and Commodore 64.

To keep a young person's interest, an educational program has to be fun to play. "Munchmath" presents an arcade-style character that relies on the player's correct answers to math problems to stay ahead of a ghost that is trying to gobble him up.

The program begins by asking for the player's name, the type of problems wanted (addition, subtraction, multiplication, or division), and the starting level of difficulty. Problems are then presented on the screen for the player to answer. Each correct answer scores ten points and moves "Munchie" one step closer to the power prize. The ghost, however, stays in hot pursuit only three steps behind. After 15 correct responses, Munchie eats the power prize and the tables are turned. Munchie chases the ghost across the screen, eventually catching him and scoring a bonus of 100 points. The difficulty level then advances one notch higher and new problems are presented.

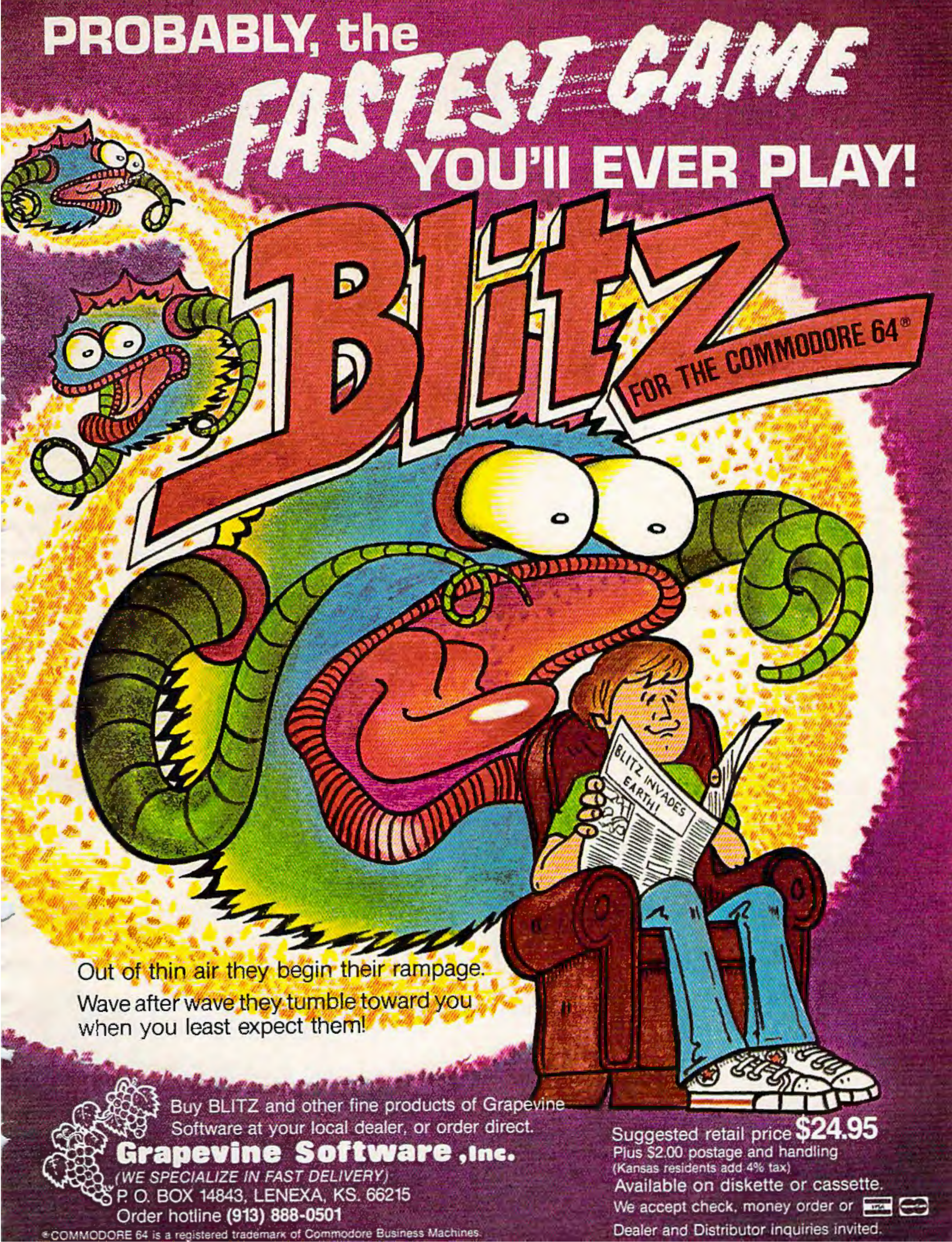
The ghost moves into action when the player gives a wrong answer. First, the correct answer is displayed for the player to study. Then the ghost advances one step closer to Munchie. Three incorrect answers and the ghost catches poor Munchie and gobbles him up. This results in a loss of 50 points and a return to the next lower level of difficulty.

If a Q is typed in response to a problem instead of a number, the game stops. A scoreboard is printed which shows the number of problems the player has been given, the number answered



Practicing simple addition with "Munchmath," 64 version.

Practicing easy multiplication with the VIC version of "Munchmath."



PROBABLY, the
FASTEST GAME
YOU'll EVER PLAY!
Blitz
FOR THE COMMODORE 64®
BLITZ INVADES EARTH!
Out of thin air they begin their rampage.
Wave after wave they tumble toward you when you least expect them!
Buy BLITZ and other fine products of Grapevine Software at your local dealer, or order direct.
Grapevine Software, Inc.
(WE SPECIALIZE IN FAST DELIVERY)
P. O. BOX 14843, LENEXA, KS. 66215
Order hotline (913) 888-0501
®COMMODORE 64 is a registered trademark of Commodore Business Machines.
Suggested retail price $24.95
Plus $2.00 postage and handling
(Kansas residents add 4% tax)
Available on diskette or cassette.
We accept check, money order or
Dealer and Distributor inquiries invited.

correctly, the number answered incorrectly, and the percentage of correct answers. The player may then choose to resume the game or to end play.

The program has been extensively tested by my six- and eight-year-old daughters, as well as the neighborhood children, and its appeal holds up very nicely.

Munchmath fits neatly into an unexpanded VIC. (There's also a version for the 64.) It uses custom characters, so you should remove any memory expanders you may have on your VIC, unless you know how to rearrange the memory.

If you want to avoid the drudgery of typing in the program (VIC version only), send a blank cassette, a stamped, self-addressed envelope, and $3 to:

Bob Wallace
6649 Thunderhead Lane
Placerville, CA 95667

(Since I'm in the Navy and frequently gone, my friend Bob Wallace has volunteered to make copies.)

Program Description

Here's a breakdown of the program (VIC version):

Lines 10-35: Initialization and delay subroutines.

Lines 40-50: Answer-checking.

Lines 55-60: Print titles computer-style.

Lines 65-145: Generate problem and print it in proper format.

Lines 150-180: Ghost catches Munchie. Generate sound effects, subtract 50 points, and reduce difficulty level.

Lines 185-235: Munchie reaches the power prize and chases the ghost. Bonus of 100 points, advance to next level.

Lines 240-255: Move Munchie and Ghost.

Lines 260-275: Print level and score. Clear old answer from screen.

Lines 280-300: Print scoreboard at end of game. Restart or end program.

Lines 305-345: Special characters.

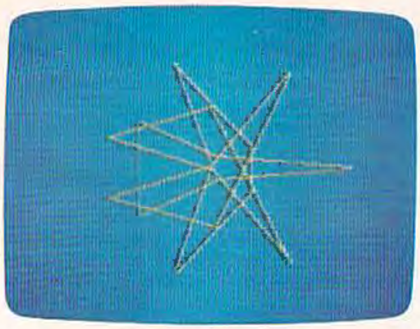
Lines 350-385: Titles.

Lines 390-445: Get player's name, choice, and level.

See program listings on page 186.

Young People

COMPUTE!'s Gazette wants to know what today's young people are doing with computers. We want our readers to know, too. If you've written an interesting program for the VIC-20 or Commodore 64, share it with us.



VIC Super Expander Graphics

Kenneth Knox

Want to see some stunning graphics on your VIC? If you have a Super Expander cartridge, type in this short program and you might be surprised to see what effects are possible.

Here's how you can show off the amazing graphics which can be created with the VIC-20 using the Super Expander cartridge.

Polar figures, in which the points are part of a circular function, may be formed in many ways. In lines 120, 140, and 150 of this program, we see one way to generate unusual point curves. Chords (lines connecting points of a point figure) are drawn using selected points. This program combines these functions in an unusual way. When you type RUN, fascinating patterns are drawn continuously until you press RUN/STOP-RESTORE. Pressing RUN/STOP alone will freeze a pattern on the screen.

How It Works

Lines 40 - 70 — Initialization and graphics setup.

Lines 100 - 170 — Put points (x,y) into memory in an array and plot the points.

Lines 200 - 220 — Draw the patterns as a series of lines with selected points of the point figure.

Lines 230 - 250 — Utility parts of the program.

Several modifications are interesting. Replace line 40 with :

40 B = 500 : L = 0

The new point figures now look much the same but are drawn differently. The line figures vary also.

Another option is to change line 100 to:

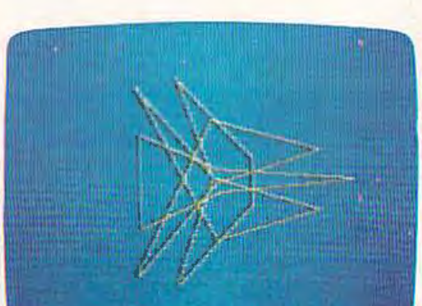
100 FOR Z = 12 TO 20

This will create more complex point figures with symmetrical features.

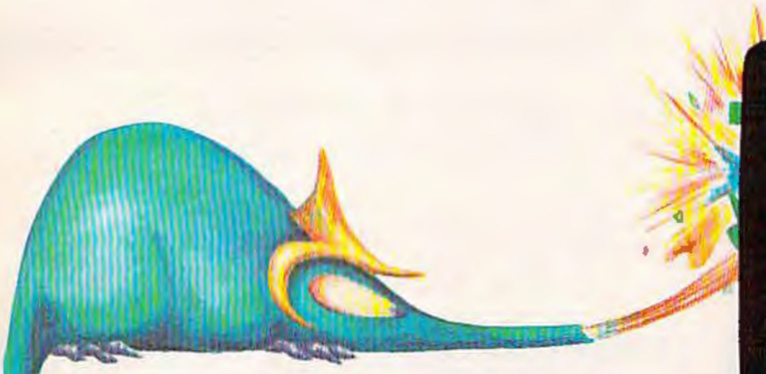
There are no unusual techniques here, so it should be easy to recreate these figures on any computer with high-resolution graphics.

See program listing on page 195.





64 Aardvark Attack



Last month, COMPUTE!'s Gazette published "Aardvark Attack," an educational game for the unexpanded VIC-20 originally written by Todd Heimarck. This month, we have an updated version for the Commodore 64, translated by Mark Sugiyama, Programming Assistant. Here's a summary of how the game works for those who missed our last issue.

An "A-bomb" drops toward city number 3 in "Aardvark Attack" for the Commodore 64.

Mutant aardvarks from Andromeda are attacking Earth with unique weapons — alphanumeric bombs. There are 26 types of bombs, and each requires a different defense on your part. Typing the letter M, for example, sets up the defense against M bombs. The same tactic applies to the other 25 letters of the alphabet.

The aardvarks are attacking Earth's ten largest cities, numbered 0 through 9. Once you set up the defense, you have to decide which city is being attacked.

Keyboard Practice

"Aardvark Attack" isn't designed to teach touch-typing, but it will give you practice on the keyboard if you're just learning to type. For youngsters, it can teach letter and number recognition.

Experienced typists should enjoy Aardvark Attack, too. It uses the entire keyboard and includes variable speeds that will challenge anyone.

Rules Of Play

Your job is to type the correct letter, and then the correct number. The letter of the bomb about to drop appears in a "radar window" in the upper-right corner of the screen. When it appears, type it. If you miss, try again. When you get the right letter, the falling bomb reverses colors. Then you must look at the row of numbers representing cities at the bottom of the screen and type the matching number to destroy the bomb.

The selectable speeds range from easy to nearly impossible. The faster games award more points.

Parents or teachers can do the typing for very young children who are just learning their letters and numbers. The child can call out the letter or number for them to type.

See program listing on page 191.

ANNOUNCING A NEW GAME SO ORIGINAL YOU NEED INSIDE SECRETS JUST TO SURVIVE...MUCH LESS WIN!

Now In Every Dragon Hawk Package: Free Strategy Cards.

Great new game — great new way to play. You are the Dragon Hawk, soaring to attack — and escape from — a host of flying monsters. Each time you press the trigger on your joystick, the hawk's wings flap, lifting you into position to dive, talons extended.

One pounce and another phoenix bird or flying iquana is reduced to a mere floating feather. But if you fail to get above your enemies... zap! You've had it! And you've got to avoid the massive lightning bolts, too.

Finally, on the seventh level, you come face to face with the dragon himself. But you won't be alone.

To get you there faster and make playing Dragon Hawk more fun than any Commodore 64 game ever, you'll have help at your finger tips. Strategy cards with key tips on crucial parts of the game are included free in your package.

Get your claws on Dragon Hawk right away. It's a thrill so new and different you may never come down.

You'll find Dragon Hawk for the Commodore 64 with disk drive at your local dealer now, or write to UMI direct.

umi

United Microware Industries, Inc.
3503-C Temple Avenue
Pomona, CA 91768 (714) 594-1351

64 Timepiece

Joe DiNicola

In the premier issue of COMPUTE!'s Gazette (July 1983) we published "VIC Timepiece," a clock program originally written for the unexpanded VIC-20 by Joseph D. Wright. A reader, Joe DiNicola, has translated the program to work on the Commodore 64. Following is an adaptation of the original article.

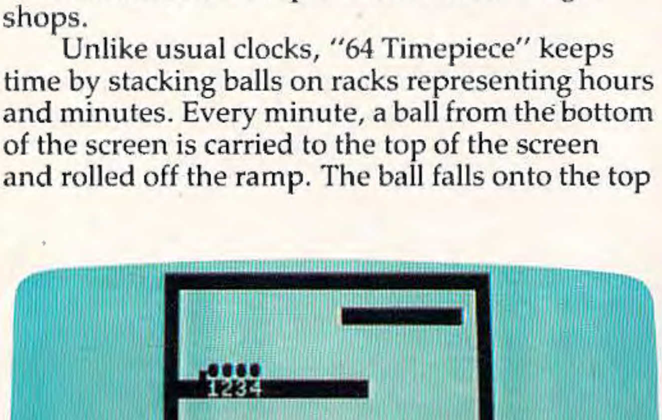
Here's something for your computer to do so it won't get bored when you're not using it. The program is patterned after the plastic rolling-ball clocks often sold in department stores and gift shops.

Unlike usual clocks, "64 Timepiece" keeps time by stacking balls on racks representing hours and minutes. Every minute, a ball from the bottom of the screen is carried to the top of the screen and rolled off the ramp. The ball falls onto the top rack, where it represents one minute. When the rack accumulates five balls, four of them return to the bottom of the screen and one rolls onto the middle rack.

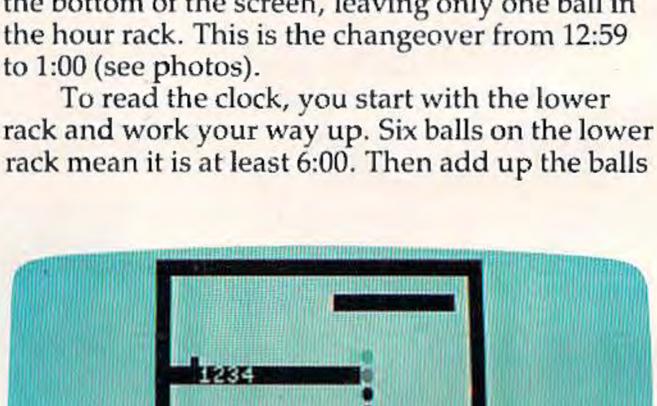
Each ball on the middle rack represents five minutes. When this rack fills up with 11 balls – and when the top rack overflows again – all balls but one return again to the bottom of the screen. The remaining ball rolls to the lower rack, where each ball represents one hour.

When the hour rack fills up with 12 balls – and when the two upper racks overflow – a chain reaction is triggered as all the racks empty into the bottom of the screen, leaving only one ball in the hour rack. This is the changeover from 12:59 to 1:00 (see photos).

To read the clock, you start with the lower rack and work your way up. Six balls on the lower rack mean it is at least 6:00. Then add up the balls

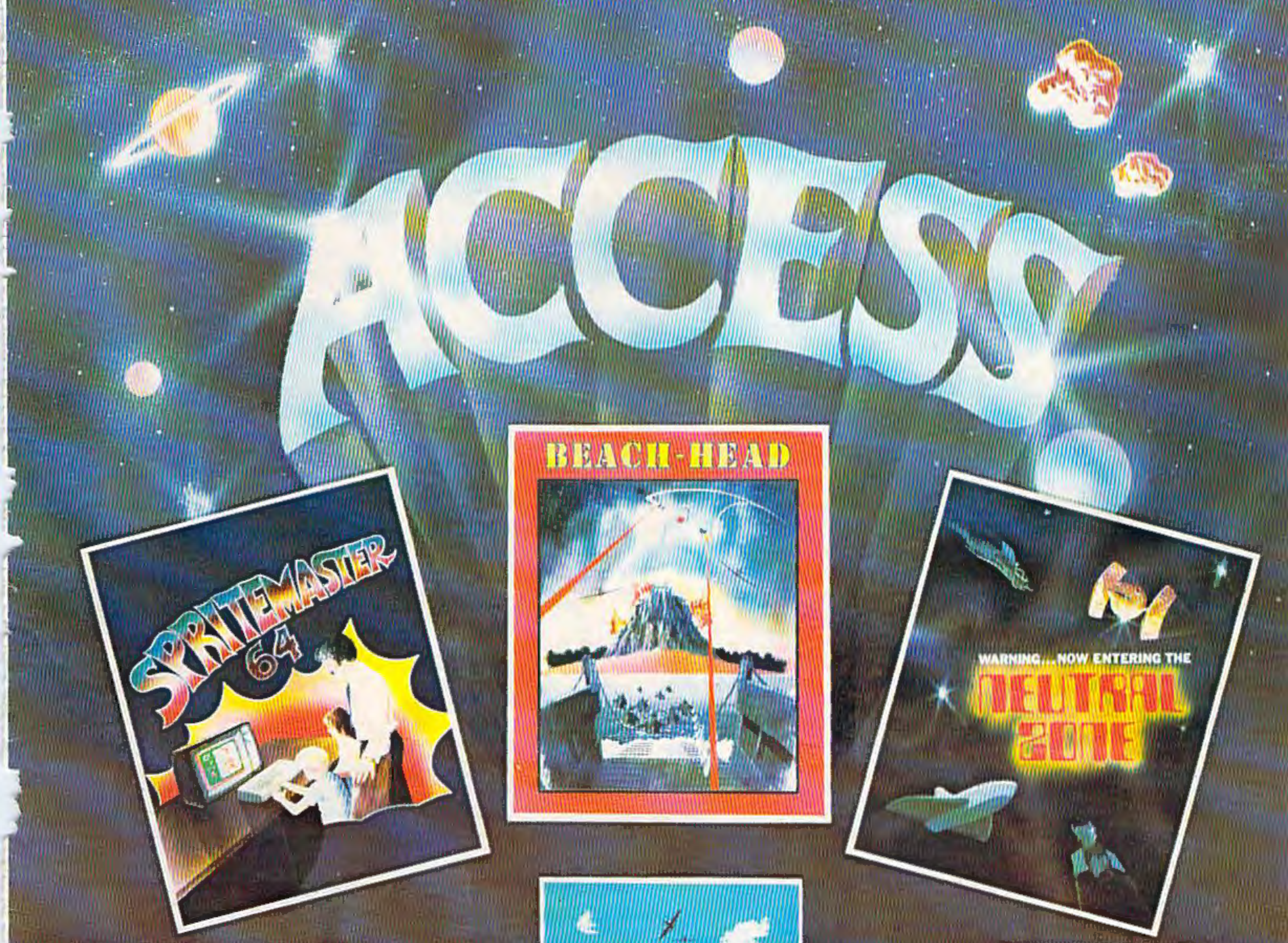


It's 12:59 – all the racks are filled with balls.



Balls from the top rack start tumbling down, triggering chain reactions on the lower racks...

COMMODORE 64™ SOFTWARE



SPRITEMASTER™ is not just another sprite editor. It's the finest utility available for multicolor sprite animation and game programming. It will have you making full color animated objects in just minutes. People running, birds flying or tanks rolling are a snap with Spritemaster. It will **automatically** append your sprites to other programs. It's easy to use and understand and comes with a full 21 page instruction manual and samples of animated sprites to get you started. (Suggested retail price... $35.95)

GENERAL QUARTERS! BATTLE STATIONS! As chief commander of land and sea forces in the Pacific, your mission is to obtain a quick naval victory, and invade enemy territory with land forces. **BEACH-HEAD**™ is a 100% machine language game and offers multi-screen action with high resolution, three dimensional graphics. (Suggested retail price... $34.95)

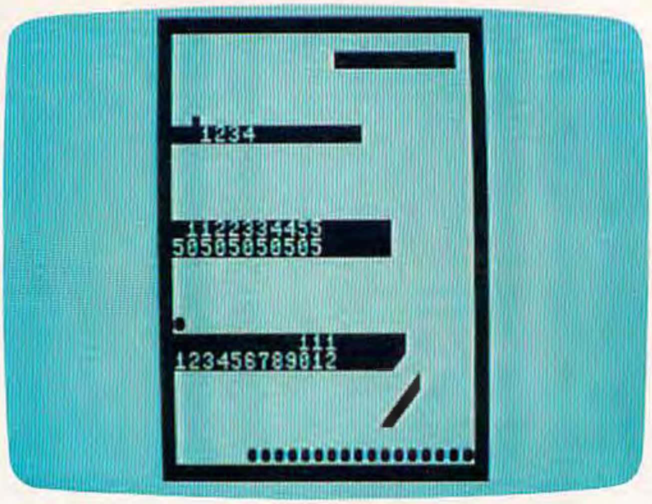
NEUTRAL ZONE™ takes you to the outer edges of the galaxy, to ALPHA IV, a long range early warning station whose mission is to detect alien intruders from other galaxies.

NEUTRAL ZONE™ is the ultimate in high resolution, fast action, arcade quality games. It is written in 100% machine language and features smooth scrolling of the 360 degree panorama. The realism is unbelievable. (Suggested retail price... $34.95)

SPRITEMASTER™, **NEUTRAL ZONE**™ AND **BEACH-HEAD**™ ARE AVAILABLE AT YOUR LOCAL COMMODORE DEALER ON EITHER DISK OR TAPE.

ACCESS SOFTWARE INC

925 EAST 900 SOUTH, SALT LAKE CITY, UTAH 84105
TELEPHONE (801) 532-1134



...until all the racks are emptied, except for one ball on the hour rack – indicating 1:00.

on the two minute racks. If the middle rack is full, and if there are three balls on the upper rack, it is 6:58 (55+3=58).

When you first run 64 Timepiece, it asks you to set the time. Just follow the instructions and enter the correct time as a three- or four-digit number *without a colon*, such as 658 for 6:58.

See program listing on page 189.


COMMODORE 64 PROGRAMS
BUSICALC 3
BUSINESS
CALCULATION
PROGRAM
$129
The "TRUE 3-D"
Electronic Spreadsheet
for the Commodore 64
BUSICALC 1
BUSINESS
CALCULATION
PROGRAM
$69
A "BEST BUY"
Electronic Spread Sheet
for all Commodore's
BUSICALC 2
BUSINESS
CALCULATION
PROGRAM
$99
The MOST CAPABLE
Electronic Spread Sheet
for the Commodore 64
WE'RE HERE!

Connect The Dots

Janet Arnold

"Connect The Dots" is an entertaining graphics program for young children who can locate numbers and letters on the keyboard. Originally written for the unexpanded VIC, we've added a version for the Commodore 64.

As teachers at a small private school, my husband and I saw the children anxious to get their hands on our VIC-20 computer whenever we brought it to class. Unlike many adults, who are hesitant to use it or even refuse to touch it altogether, the children jockeyed for their turn at even the dullest programs we loaded.

I wrote "Connect The Dots" to provide my own children and my preschool/kindergarten students with a "game" that could entertain while reinforcing their skills at the same time.

Making Dots Into Pictures

Here's how it works. The child is given a four-item menu from which to choose the picture he or she wishes to draw. The greater the number of dots, the longer it takes to complete the picture.

A grid appears on the screen. Some of the squares contain markings. Tell the child to look for the solid dot, because that's what must be matched with the coordinates. When the prompt "Number?" appears at the top, show the child how to press the correct number coordinate and hit RETURN. Answering the next prompt, "Letter?", will probably take longer unless the child is familiar with the keyboard.

A wrong number-letter combination is answered with a low "uh-oh" sound and the words, "Try again."

After a correct answer, the computer draws a line connecting the dots and plays an amusing sound effect. A short timing loop delays this just long enough for the child to look from the keyboard back to the screen to enjoy this reward.

The finished drawing is accompanied by a short tune and the remark, "Good job! Draw again?" Hitting a Y calls up the menu again. An N ends the program.

Working With The Child

When introducing this activity to a child, a few additional explanations may be necessary. Be sure to explain the difference between the number 0 — point out the slash — and the letters O and Q.

A tot whose visual discrimination is immature might reverse letters. Connect The Dots can give that child enjoyable practice in overcoming this. If you notice a child confusing 7 and L, for instance, ask, "Is that line walking on the ceiling or on the floor?"

Of course, preschoolers and some kindergartners who are still learning their numbers and letters will enjoy naming them aloud to you.

Children with short attention spans should try the pictures with fewer dots. Even then, be prepared to help them along or to complete it for them. This isn't all bad, because the time spent with children at the computer can enrich your relationship and will tell them that their activities are important to you.

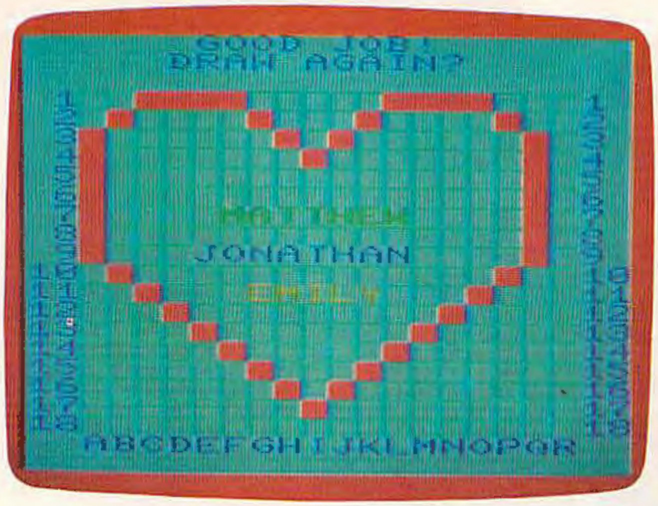
There's no time limit in Connect The Dots, so don't rush the child. This will be a welcome relief to the child who equates computers with tense, timed, shoot-or-be-shot action.

If some children's eyes have trouble following the grid from the dot to the coordinates, show the children how to trace with their fingers directly on the screen.

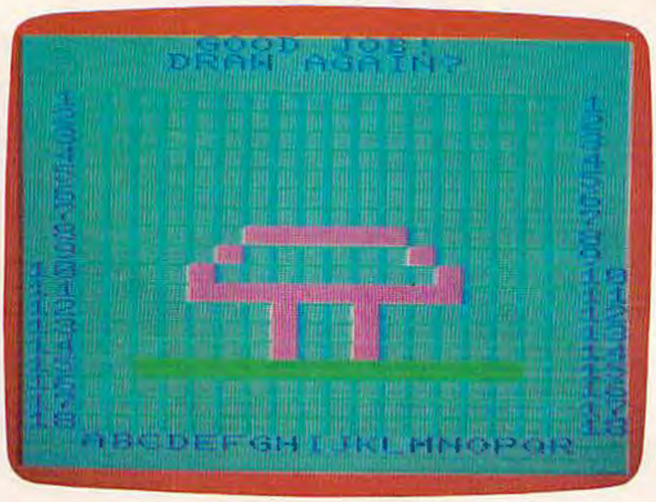
Details Of The Program — VIC Version

It is important to type this program exactly as shown. Too many extra spaces might cause VIC users to run out of memory.

The fourth selection on the menu is a heart inscribed with my children's names. Substitute your own message by changing line 650.



Connecting the dots to draw a heart on the VIC-20. (You can modify the program to substitute any names in the heart.)



A completed mushroom, one of four pictures included with "Connect The Dots" (VIC-20 version).

Following is a line-by-line program description for the VIC version, giving the starting line number of each section:

1 — Title and instructions.
10 — Menu.
20 — Draw grid.
90 — Search DATA for starting point of chosen picture.
100 — Read 4 pieces of DATA per dot and POKE dot.
110 — Ask for dot's coordinates.
140 — Response for wrong answer.
200 — Response for right answer.
500 — Set up butterfly.
550 — Set up mushroom.
600 — Set up dog.
650 — Set up heart.
900 — Response for completed picture.
1000 — DATA for butterfly.
1500 — DATA for mushroom.
2000 — DATA for dog.
2500 — DATA for heart.

Designing Your Own Pictures

Part of the fun of this program is designing your own pictures. My five-year-old, Jonathan, contributed the mushroom found in Connect The Dots by coloring in squares of graph paper.

To substitute a picture of your own, design one using the accompanying grids. They correspond to the square grids in the VIC and 64 versions of the program. Remember that most of your design should consist of a continuous line as in dot-to-dot pictures. Anything else must be POKEd in when the picture is first set up.

For the purposes of this article, let us assume that you have a VIC and have drawn a clown to

Figure 1: Grid For Designing Pictures On The VIC

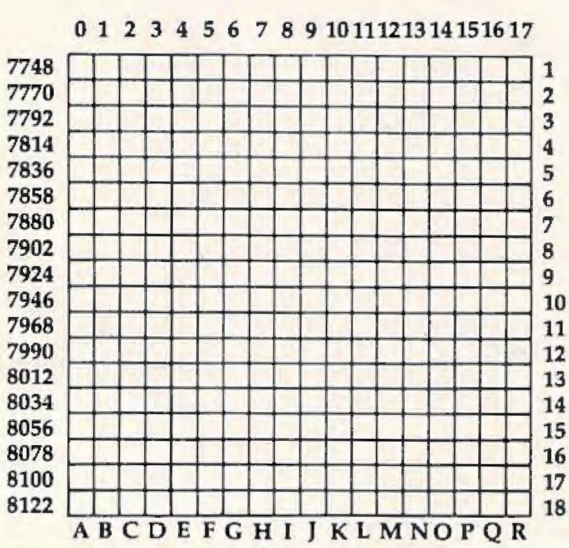
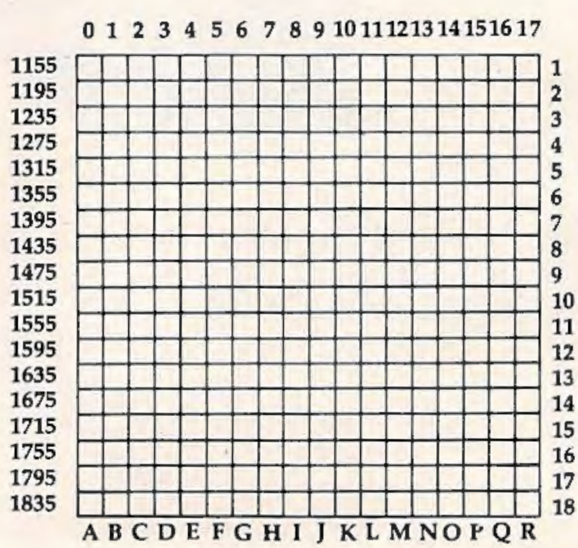


Figure 2: Grid For Designing Pictures On The 64.



ADVERTISEMENT

Growing computer industry expands authors' choices

The rapidly expanding personal computer industry offers greater opportunities for the software programmer and author in search of a publisher.

Yet the growth poses its own problem - the choice of a publisher.

Here is a list of questions to consider when looking for the publisher best-suited for your product:

-How large is the publisher's distribution network? A publisher with international connections can offer more exposure than companies limited to regional or national sales.

-How will your product be marketed and advertised? No matter how good the program is, if people don't know about it, it won't sell. Look for a publisher with a marketing budget large enough to give individual attention to the program.

-Does the publisher market programs for more than one computer? The days of limited selection in hardware are long gone. Limiting programs to one or two computers can limit sales and profits. Authors can increase their share of the marketplace by looking for a publisher devoted to converting programs to a variety of popular computers.

-Does the publishing house lend technical support to authors? Some publishers only accept programs ready for the marketplace. A lot of good ideas are lost in the long run. The publisher that offers assistance invests a greater stake in the product, the author and the success of the product.

-Does the publisher offer complete product support to consumers? In these times of consumer awareness, the company that has established a network to answer customer questions about its products fares better than those who do not offer this support.

Each of these services leads to greater sales which in turn lead to greater profits for the individual programmer.

Sierra On-Line, Inc. is committed to paving the way for an author's success.

Sierra On-Line's product line is distributed worldwide with production facilities in the United States, Japan, Australia, the United Kingdom and South Africa.

Sierra On-Line employs a well-financed, in-house marketing and advertising staff with a knack for creating tailor-made campaigns for products.

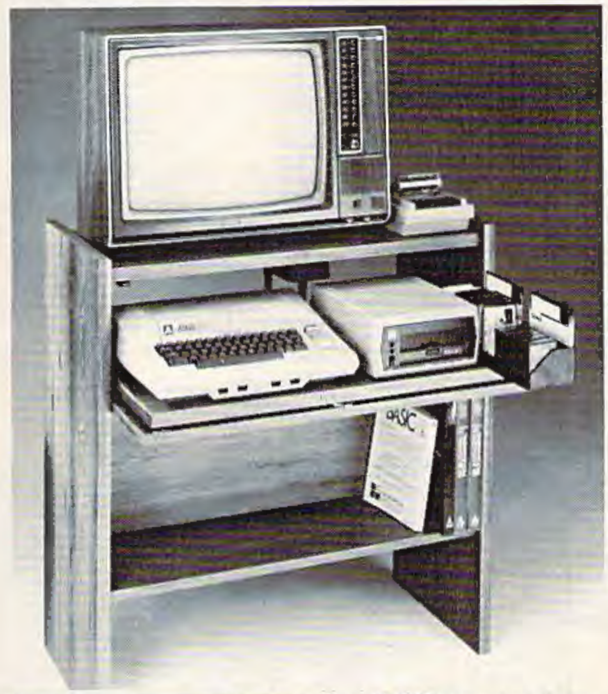
Each program is evaluated by experts, who may suggest enhancements to improve the product and to increase its appeal to customers.

Further, Sierra On-Line isn't limited to a single computer. The company closely monitors computer trends and makes existing products available for the most popular lines - all to the author's benefit.

A packet for authors with more information about the software submission process and our company is available by writing Sierra On-Line, Inc., Sierra On-Line Building, Coarsegold, CA 93614, or by contacting David Siri or Howard Luthy by phone at (209) 683-6858.

ARE YOU A SMART BUYER?

For **$89.95** this is a smart buy if you're looking for a place to store your computer, peripherals, and accessories without spending a fortune.



The CS 1632 computer storage cabinets compact yet functional design fits almost anywhere while housing your computer monitor, joysticks, software, books and peripherals all for only $89.95.

The slide out shelf puts the computer at the right height and position for easy comfortable operation.

The fold up locking door keeps unwanted fingers off the key board when not in use.

To store joysticks just turn them upside down and slide them into the inverted storage rack.

Twist tabs on the back of center panel allow for neat concealed grouping of wires, while power packs rest hidden behind center panel on shelf.

The slide out software tray has room for 14 cartridges or cassettes and up to 30 diskettes. Most brands of software will fit between the adjustable partitions with a convenient hook for the spare key at rear.

Stand fits Atari 400 & 800, Commodore 64 & VIC 20, Ti 99/4A and TRS-80.

Cabinet dimensions overall 36" high x 33-7/8" wide x 16" deep.

Cabinet comes unassembled. Assembly requires only a screwdriver, hammer, and a few minutes of your time.

Choice in simulated woodgrain, of warm golden oak or rich natural walnut finish.

HYTEC SYSTEMS

To order CS1632, send $89.95 to:
P.O. Box 446 West Linn, OR 97068
For Fast Phone Orders Call
Toll Free **1-800-547-3100**
Inside Oregon Call **(503) 635-6667**

Name ______________________

Address ______________________

City ______________ State ________ Zip ________

☐ Golden oak finish ☐ Natural walnut finish

☐ My personal check, cashiers check or money order is enclosed.

☐ Bill my VISA # ______________ Exp. Date ________

☐ Bill my Mastercard # ______________ Exp. Date ________

Card Holders Signature ______________________

Immediate shipment if in stock. If personal check is sent, allow additional 2 weeks. Prices subject to change. Shipment subject to availability. Cabinet shipped unassembled in 2 cartons. Ships UPS frt. collect FOB Portland, Oregon.

replace the dog in the listed program. Substitute the title "CLOWN" for "DOG" in line 11. This changes the menu to read "C=CLOWN".

Lines 600-610 POKE in the dog's tail and a starting square (SQ). Use these lines to POKE in your clown's nose, for example. (Hint: Since children are always asked the coordinates of a *solid* dot, use an open O or you will confuse them.)

To compute the screen memory location of the nose, add the four-digit row number to the left of the grid to the column number above the grid. This same number + CD is your color memory location.

POKE in your starting square — use code #160, a reversed space — and assign SQ the value of the screen memory location of that starting square.

Now just figure your DATA. The computer reads four pieces of data per dot: screen memory location (A), color of the line to be drawn (B), number-letter coordinates of the dot (E$), and the direction that the line will travel to reach the dot (S). Figure each as follows:

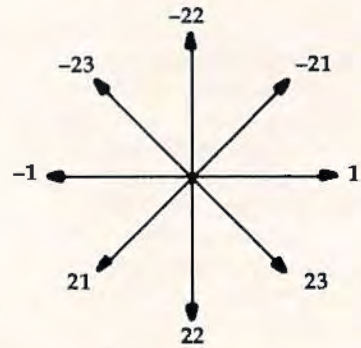
First, compute the screen memory location of the dot as explained earlier.

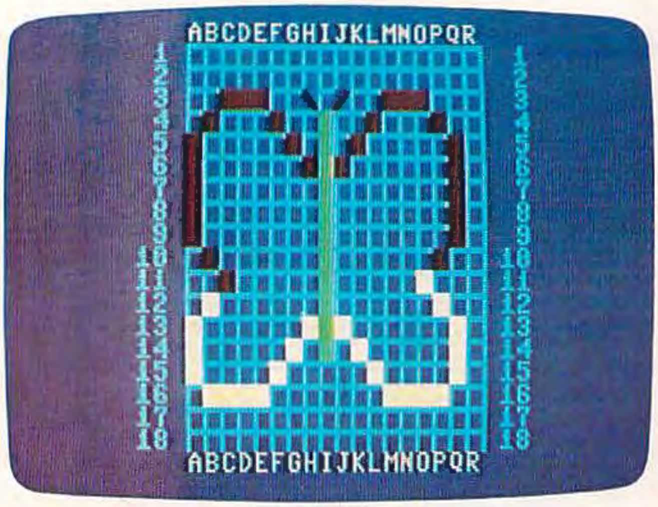
The second number is the color code of the line to be drawn. The color code is always the number of the color's computer key minus 1 (e.g., black=0, red=1, etc.).

Third, look at your grid to find the number-letter coordinates of the dot. The number comes first and is found on the right side of the grid. Follow this with the letter. Do not separate the number and letter with a space.

The last number is a STEP value. This number tells the computer in which direction the line should be drawn. For instance, a line moving from left to right travels one space at a time, so its STEP value is 1. From right to left, the line moves backwards one space at a time, making its STEP value -1. A line traveling diagonally up to the left has a STEP value of -23 on the VIC (or -41 on the Commodore 64), since the computer skips back 23 (or 41) spaces before POKEing the next square.

Use this diagram to figure STEP values for the VIC:





A finished butterfly – Commodore 64 version.

A horse – 64 version.

Use this diagram to figure STEP values for the 64:

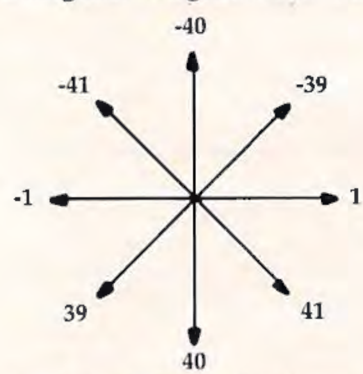


Figure each dot's DATA in the same manner. Separate each piece of DATA with a comma. You must insert your new DATA into the proper line numbers, so check the program explanation listed earlier. Since you are replacing the dog with your clown, your DATA will go in lines 2000-2030. Be sure to leave the first piece of DATA — C — in line

2000. This is the DATA that the computer searches for to set the DATA pointer.

If you would rather spend your time playing with your child than typing in this program (VIC version only), send $3, a blank cassette tape, and a self-addressed, stamped mailer to:

Janet Arnold
620 Alger
Owosso, MI 48867

See program listings on page 192.

REVIEWS

Colorcraft Graphics Animator For VIC And 64

Shelby Goldstein

ColorCraft for the VIC-20 and Commodore 64 computers is an entertaining, educational graphics program that is ideal for the computer novice or young student. (We reviewed the VIC-20 version.) The emphasis is on graphics and animation, but *ColorCraft* provides an excellent introduction to the computer in general. It covers topics from setting up the computer and loading tapes to using a screen editor.

The first-time user will quickly become familiar with the computer's graphics keys, function keys, and the CONTROL key. *ColorCraft* also teaches about the computer's special editing features, including the CLR/HOME, INST/DEL, and cursor movement keys. In addition, the concepts of screen wrapping, repeat keys, screen centering, memory, and mass storage are introduced.

Graphics Commands

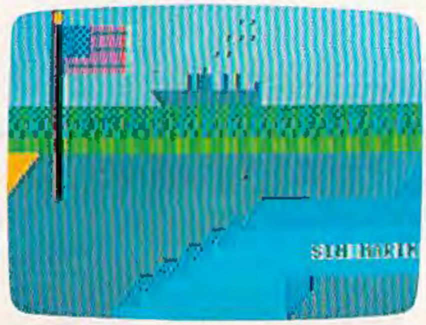
Animation is created with *ColorCraft* by designing different pictures or text displays on several different screens (pages) and then flipping through them quickly. Text and pictures are created and animated with *ColorCraft* using five commands: Form, Edit, Run, Save, and Load. A sixth command, Copy, is available on VIC-20s with memory expansion and on the 64. Commands are executed by pressing the appropriate letter on a command line displayed along the top of the screen.

The first option, Form, allows you to change the page size to various dimensions. As the page size decreases, the number of available pages increases. Memory expansion, of course, allows VIC users to have more pages of any size. For example, an unexpanded VIC will hold up to five pages of the smallest page size, 5 x 10 (5 lines by 10 characters), while 16K will increase that number to 50 pages of the 5 x 10 size, or 12 pages of the largest size, 21 x 22. Animation requires several pages, so smaller-sized pages would be the most common choice, especially on an unexpanded VIC.

In addition to controlling page dimensions, the Form command lets you select the border color. This is done simply by pressing the f3 special function key. There are eight colors from which to choose: black, white, red, cyan, purple, green, blue, and yellow.

Most work with *ColorCraft* is done in the Edit mode. The editor is entered by pressing E while the command line is on the screen. In the Edit mode, you can choose the cursor color, change the character set, and turn reverse printing on and off. You can write text or create graphics with the graphics keys.

One particularly nice characteristic of the editor is the advanced drawing command called Direction. This option allows you to set the direction that the cursor will move. For example, Direction 7 is down; it commands the cursor to move downward with each subsequent character typed instead of moving in the normal direction (to the right). Eight directions, including diagonals, are possible. This feature is particularly useful for drawing shapes requiring straight lines.



Two examples of graphics screens created with the Commodore 64 version of ColorCraft.

Editing Graphics

Revisions are easy with the editing keys. Characters or spaces are added or erased with the INST/DEL key. The CLR/HOME key can be used to erase an entire line or the entire screen. On VIC-20s with memory expansion, you can also delete or insert lines by pressing CTRL-D or CTRL-I, and you can center lines with CTRL-C.

While in the editor, you choose the screen color by pressing the f3 function key. There are 16 colors available. These include the eight colors mentioned above plus orange, light orange, pink, light cyan, light purple, light green, light blue, and light yellow.

One of the most important features of the editor in creating animation is the timing selection. The Timer determines how long each page is displayed. Your pictures or text can move very slowly or very quickly, and each page can stay on the screen a different length of time. The time is set for each page with the f5 function key. The larger the timing number, the slower the page-flipping. The Timer works with the Timing Interval, which is the constant that determines the overall speed of your *ColorCraft* program. The two can be combined to set as much as 45 seconds display time per screen.

The final feature of the editor uses the f1 function key to move to the next or previous page. This is especially helpful for editing the various pages of your sequence, and for previewing your animation by flipping through the pages. This also is enormously helpful if you have memory expansion, because the f1 and C keys can duplicate an entire page as many times as you wish. This makes animation much easier – you can duplicate one screen several times and then make just slight changes to each screen.

Easy Animation

After you've created your pages and finished editing them, you can sit back and view your accomplishments with the Run command.

Several features can be executed while the sequence is running: f7 turns on the windows that display your page numbers, line numbers, position numbers, direction, timing and color; f3 slows down the screen displays, and SHIFT-f3 speeds them up; f5 runs your program in reverse. The space bar freezes the display and starts it moving again.

Your work can be saved on tape or disk. A print program is included in the *ColorCraft* manual that allows you to access your *ColorCraft* files from BASIC. This program is written for the Commodore printer only, but if you know a fair amount of programming, you can get it to work on other printers.

The flip side of the program tape contains samples created with *ColorCraft*. These are helpful in illustrating some of the possibilities of the program. In addition, the accompanying manual is one of the most lucid manuals I have ever seen. It is written in an easy-to-read style with many detailed explanations and examples. Chapter VII takes you step by step through a sample program that uses all of *ColorCraft*'s editing features and illustrates simple animation. The manual even includes a ten-page glossary of terms used in the program, as well as many common computer terms.

ColorCraft is an excellent program for a first-time computer user or as an introduction to computer graphics. It is not intended for creating advanced graphics. There is a limit to the sophistication you can achieve when restricted to the VIC and 64 graphics characters. Also, the program is very limited on an unexpanded VIC.

But as with many graphics programs, the weakest link is the user, not the program. If you can't draw on paper, you probably can't create great works of art on a computer. You can, however, learn a great deal from this program and have a lot of fun, too.

ColorCraft
Sim Computer Products, Inc.
P.O. Box #7
Miquon, PA 19452
$24.95 VIC-20 cassette
$29.95 Commodore 64 cassette
$34.95 Commodore 64 disk

Writers...

COMPUTE!'s Gazette is looking for well-written, clearly explained articles for beginning and intermediate users of VIC-20 and Commodore 64 personal computers. If you have an idea for a feature article or tutorial, submit a manuscript or send us a query letter.

Fantastic Sale!

HES Special Programs For VIC-20 Computer

Call Today!! Call Today!! Call Today!!

SYNTHESOUND Sale $19.95

• Fantastic Sound & Music • Piano Keyboard

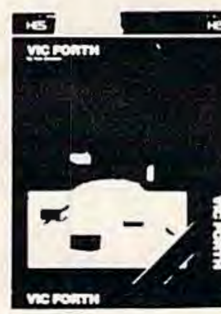
Now your VIC-20 can be used as a state-of-the-art music synthesizer and special effects generator! A very user friendly manual guides you through sessions of learning and development of your synthesizer "style"! You can program Synthesound to sound like many different instruments like bagpipes, organ, bass and many others. Synthesound brings technical musical abilities to any VIC-20 user. Prepare to step into the exciting world of the futuristic computer musician.

- VIC keyboard used as a 2 octave, fast attack keyboard.
- 3 voices with 4½ octave range.
- 4 low frequency ocillators.
- 4 envelope shaping controls (Attack, Decay, Sustain, Release).
- No additional hardware required.
- Spectacular special effects.
- 4 voice/20 note sequencer for repeating melodies or rhythms.
- High resolution piano keyboard display.

List $44.95 / **Sale $19.95**

VIC FORTH LANGUAGE

Faster than BASIC — easier than Assembly language — Language of Robots.



VIC FORTH is an exciting new cartridge for your VIC-20 computer. You now have a language that is more powerful than BASIC and easier to program than assembler! VIC FORTH is a nearly complete implementation of the "FORTH INTEREST GROUP" (fig.) version of FORTH. The VIC FORTH editor follows closely the standard FORTH editor described in the book, "Starting FORTH", plus it has many, many additional features.

- Full screen editor which has 16 lines of 64 characters each (standard FORTH Screen) through use of a horizontal scrolling window.
- User definable words; this means a function not already supported can be created by you and added to VIC FORTH for future use.
- Full VIC sound and color capabilities.
- Runs in standard 5k VIC.
- Output to printers and disks and tape machines.
- Up to 24k additional memory can be added.
- 31 characters for variable names.

List $59.95 / **Sale $39.95**

TURTLE GRAPHICS

• Create colorful pictures and graphics • Learn how computers compute.

Turtle Graphics is an exciting and fun way to discover computer programming concepts. Turtle Graphics is an easy-to-learn computer language. It enables the beginning programmer to master the computer in just a few minutes. By telling an imaginary turtle how to roam about on the VIC's screen, the Turtle Graphics programmer can create colorful and artistic pictures. As one becomes more and more adept at teaching the turtle to paint using the VIC's rich graphics and color palette he or she is painlessly learning all of the basics of computer programming. This not only develops computer awareness, but enriches one's math, logic, communication and artistic skills as well.

- Turtle Graphics has over 30 different commands.
- Programs may be listed on a printer and saved on or loaded from tape or disk.
- Turtle Graphics is menu driven for ease of use.
- Trace mode to help the beginning programmer follow the logic of his program one step at a time.
- Well written, easy-to-follow manual with many examples included.

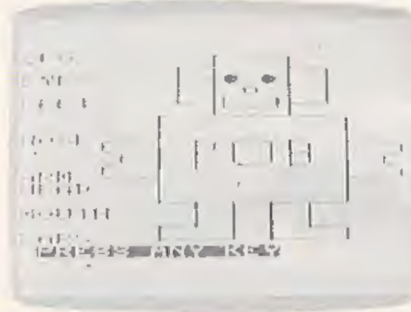
List $39.95 / **Sale $29.95**

We Have All HES Products At Super Sale Prices!!

PROTECTO
ENTERPRIZES (WE LOVE OUR CUSTOMERS)
BOX 550, BARRINGTON, ILLINOIS 60010
Phone 312/382-5244 to order

Preschool Educational Software For VIC-20

Donald Elman



Rob the robot looks OK now, but he could suddenly redraw himself and be without ears. Players must type in the name of the missing part in My Body.

Most commercial programs available for VIC-20 owners are various games designed for children 7 to 70. Very little educational software of any kind has been introduced for the VIC. Even more scarce are programs designed to entertain and educate the *Sesame Street* generation of children under the age of six.

Finally, a handful of commercial programs designed for preschoolers is beginning to hit the market. As more of these become available, parents and teachers should look for certain features that make the programs worthwhile learning tools. I have found four areas of particular significance:

Ease of use. Loading convenience, clarity of documentation, and uncomplicated input/response methods for the child. These are very important for young children, who are easily frustrated and whose ability to solve problems is not likely to be well developed.

Sophistication of graphics, color, and sound. A child's responsiveness is enhanced by an animated screen. A program that uses many of the VIC's special features in this way will promote the child's interest.

Educational features. Matching the program to the child's learning level is essential. How does the program handle positive reinforcement, prompts or corrections following errors, and increasing difficulty as the child masters one level of a problem?

Sustained interest potential. This partly depends on the absence or presence of the three characteristics mentioned above. You need to look for programs that can continue to teach and to hold the child's interest.

Here is a summary of some available software for preschoolers:

Early Games For Young Children, CounterPoint Software (cassette), $29.95. Documentation: A brochure with loading instructions and brief descriptions of each "game."

There's more packed into this single cassette than nearly any other commercial VIC tape that I have seen. This group of programs was recently adapted from earlier versions for the Apple, Atari, and IBM. They work well on the VIC. On one side of the tape are nine separate programs that can be individually loaded and run on an unexpanded VIC. On the other side of the tape is a huge, single program that incorporates all nine of the shorter programs under one master menu. However, this comprehensive program requires 16K of memory expansion and takes a full five minutes to load from the tape. Once loaded, the long version allows the user to jump around at will among the nine subprograms.

Instead of presenting all the choices on the screen at once, the 16K version's rotating menu displays a symbolic representation of one program at a time, changing to the next one every two or three seconds. By pressing any key, the user instantly stops the menu and starts the subprogram whose symbol is showing. The menu itself can become a sort of game, as the child learns to anticipate the sequence of symbols and to pounce on a key as soon as a favorite picture is shown. You can return to the menu from any subprogram by pressing one of the function keys.

Numbers. A large block number from one to nine appears on the screen. Below are the instructions "MATCH THIS NUMBER." If the correct key is pressed, a short, random-note tune is played, and a new number appears. If wrong, there is a very brief "uh-uh" sound. No corrections are given, and the child can keep trying different keys until the right one is pressed.

Count. Either one, two, or three blocks appear on the screen. Pressing a number key results in either the correct or incorrect feedback described above. After several correct answers, the level of difficulty automatically increases, with the maximum number of blocks shown going up to six, and

eventually to nine.

Add. A simple addition problem is displayed using colored blocks instead of digits. As in Count, the level of difficulty gradually increases as the child gives a string of correct answers, with the highest sum being nine. The feedback is the same as described above.

Subtract. This uses the same presentation format as Add. The largest operand is five.

Letters. A large letter appears on the screen. Below is the instruction "MATCH THIS LETTER." Correct and incorrect responses receive the same feedback as in *Numbers*.

Alphabet. This is similar to Letters, except the child must press the key of the letter that alphabetically follows the one on the screen. After a wrong answer, the child is prompted by a brief display of the entire alphabet.

Names. First a parent or older child must type in a name (or any word). Then the word disappears from the screen, and the young child must retype it correctly.

Draw. In this program, the child can create a low-resolution picture on the screen by pressing various keys. Instead of using a single key for each direction, the keyboard is divided into eight regions. The first character-sized, square dot appears in the center of the screen. Pressing a key in the upper left of the keyboard, such as 1 or 2, will add an adjacent dot above the left of the first one. Pressing a key near the bottom-center of the keyboard, such as B or N, will add a dot directly below the last one, and so on. Pressing the space bar changes the color of the succeeding dots.

Shapes. Four large shapes appear on the screen, each numbered 1-4. Three of the shapes are identical to each other. The child must press the number of the different one.

Home Babysitter, Commodore (cartridge), $29.95. Documentation: A two-page, no-nonsense set of "instructions for parents."

Commodore's own contribution to this market is a plug-in cartridge with three separate, attractive programs, each instantly accessible by pressing one of the function keys. Except for the overly ambitious title (don't expect this program to feed your child dinner or call the fire department in an emergency), it's a well-designed piece of software that could keep a youngster occupied for quite some time, with minimal supervision.

The first screen is a menu that displays a picture of the F1, F3, and F5 keys, each next to a nonverbal sample of one of the three programs. If no key is pressed immediately, the program repeatedly cycles through a short graphic/sound routine that illustrates each program. Pressing one of the top three function keys begins the corresponding program. You return to the menu by pressing the RESTORE key.

A minor problem with the menu is that the user must wait a few seconds for the computer to cycle through the entire illustration routine before the function keys work. A child might find it difficult to tell when these keys will work and when they won't.

Pressing F1 from the menu leads to Alphabet Blocks. The screen clears to white. Pressing any letter from A to Z results in a large picture of a child's block showing that letter in the lower-right portion of the screen, accompanied by a brief musical tone. At the same time, a smaller version of the letter block appears elsewhere on the screen. Pressing a different letter replaces the large block and adds a small block to the picture. After all the letters have been pressed, the 26 small blocks appear in alphabetical order starting at the upper-left corner of the screen. If a nonletter key is pressed at any point, the screen clears, and each letter block is automatically displayed in order to the tune of the "alphabet song." Once the child has learned the alphabet, he or she could manually play the alphabet song and display the blocks by pressing all the letter keys in order.

The F3 key activates Learning To Count. First, another menu appears, which allows the user to choose a maximum number (5, 10, 15, or 20) for the counting exercises by pressing one of the four function keys. Next, a rectangular outline is displayed on the screen, with two "score" columns to the right of it. For each counting problem, a random number of objects appear within the rectangle at the rate of about three per second, each accompanied by a short beep. Then the question "HOW MANY?" is displayed below, along with a large, blinking cursor.

The child must press a number (which shows up on the screen as a large-sized digit), and hit RETURN. If the child's input is correct, a large, smiling face appears, and a small smiling face is permanently added to one of the score columns. If wrong, a large frowning face appears, and the child is given another chance. After two wrong responses, the program counts the objects for

the child and puts a frowning face in the score column. For variety, different types of objects are displayed with each counting problem, including several standard and custom graphics characters.

The third program, accessed by pressing F5 from the menu, is Face Maker. More creative than educational, this one provides a nice change of pace from the intellectual rigors of the other two programs. Essentially, it is a sort of cartoon version of a police detective's kit for producing composite drawings of a suspect. The outline of a face appears on the screen. By repeatedly pressing various function keys, you can display about six different versions of each of the following: hair, eyes, ears, nose, mouth, and chin. Literally tens of thousands of different shape combinations are possible, and the color of each part can be independently set by pressing one of the color control keys.

This program also has two cute touches. One of the "eyes" is an animated wink, and one of the "mouths" sticks out its tongue; appropriately accompanied by the sound of a "Bronx cheer."

Alphabet. American Peripherals (cassette) $9.95. Documentation: None.

This is a single-function program that runs on an unexpanded VIC. For each problem, a sequence of six letters appears on the screen in alphabetical order, except that in place of one is a "?". Pressing the correct letter results in an upward-sliding tone, a smiling face, and a message such as "DOING FINE!" or "RIGHT ON!" An incorrect answer results in a downward-sliding tone, a frowning face, and a list of the entire alphabet, which remains on the screen for four seconds until the problem returns. After about eight correct answers, a brightly colored rocket ship blasts off and slowly rises until it disappears from the screen.

My Body. American Peripherals (cassette), $11.95. Documentation: None.

This single program can run on an unexpanded VIC. After asking for the user's name, it displays the following instructions: "I'm a robot. My name is Rob. When I show myself, look very carefully and I will play a game with you. We will play ten times, and each time a part of me will be missing. Type in the name of the missing part."

The screen display shows the robot and a list of words for each body part. If the correct word is typed in, the robot is redrawn, but with another part missing. If wrong, the message "LOOK AGAIN" appears. This program uses no color, no sound, and no random variations. If the user replays after all ten words have been entered correctly, the missing parts are presented again in the same order.

Recommendations

Very young children will undoubtedly need some help getting started with any of these programs. However, I've seen three-year-olds who can readily insert a tape, type "LOAD", and (with two hands) press the PLAY button. The 16K version of *Early Games*, which takes five minutes to load, requires both the RAM expansion cartridge and the tape. Then there is the possibility that the child might tire of waiting. The shorter tape programs are simple to load and run, but with the 5K side of the *Early Games* tape there is the problem of cuing up the programs in the middle of the tape. Once any of these programs is loaded, the child would probably need guidance in using the menu (if any) and understanding the response procedure the first time or two. Of course, an older child may not have any problems with this.

Because of its impressive use of graphics, sound, and color, I'd judge that *Home Babysitter* is the most delightful of these for children. Its well-balanced offering of letters, numbers, and visual creativity are nicely designed for the preschooler. A very close second in enjoyment is *Early Games*, especially the 16K version which allows the child to shift instantly from one game to another. In fact, the variety of *Early Games* might sustain more long-term interest than *Home Babysitter*. Both of these products, however, seem to be effective exercises for a youngster's early letter and number skills – a useful complement to the passive learning mode of *Sesame Street*.

By contrast, the American Peripherals tapes seem to offer less, although their lower price might tempt you to give them a try. Of the two, *Alphabet* is clearly more animated, with color/graphic/sound feedback, prompts, and randomly varying reinforcement messages. Any child would surely enjoy the rising rocket ship. However, once the novelty wears off, there is no variety to sustain interest.

My Body, in the version that I tested, has few redeeming features. Its basic concept is unique, but it could use a lot of "extras." Since it's written in BASIC, perhaps you could use your own programming skills to add color, sound, and other enhancements.

80 COLUMN PRINTER SALE—$149.00*

COM-STAR T/F

Tractor Friction Printer

only $219

(Parallel or Serial)

• Lowest price quality tractor friction printer in the U.S.A. • Fast 80 characters per second • 40, 46, 66, 80, 96, or 132 characters per line spacing • Prints labels, letters, graphs, and tables • List your programs • Print out data from modem services

*STX-80 COLUMN PRINTER - $149.00

Prints full 80 columns. Super silent operation, 60 CPS, prints Hi-resolution graphics and block graphics, expanded character set, exceptionally clear characters, fantastic print quality, uses inexpensive thermal roll paper!

DELUXE COMSTAR T/F PRINTER—$219.00

The Comstar T/F is an excellent addition to any micro-computer system. (Interfaces are available for Apple, VIC-20, Commodore-64, Pet, Atari 400 and 800, and Hewlett Packard). At only $219 the Comstar gives you print quality and features found only on printers costing twice as much. Compare these features.

• **BI-DIRECTIONAL PRINTING** with a LOGIC SEEKING CARRIAGE CONTROL for higher through-put in actual text printing. 80 characters per second.

• **PRINTING VERSATILITY:** standard 96 ASCII character set plus block graphics and international scripts. An EPROM character generator includes up to 224 characters.

• **INTERFACE FLEXIBILITY:** Centronics is standard. Options include EIA RS232C, 20mA Current Loop.

• **LONG LIFE PRINT HEAD:** 100 million character life expectancy.

• **THREE SELECTABLE LINE SPACINGS:** 6, 8 or 12 lines per inch.

• **THREE SELECTABLE CHARACTER PITCHES:** • 10, 12 or 16.5 characters per inch. 132 columns maximum. Double-width font also is standard for each character pitch.

• **PROGRAMMABLE LINE FEED:** programmable length from 1/144 to 255/144 inches.

• **VERTICAL FORMAT CONTROL:** programmable form length up to 127 lines, useful for short or over-sized preprinted forms.

• **FRICTION AND TRACTOR FEED:** will accept single sheet paper.

• **224 TOTAL CHARACTERS**

• **USES STANDARD SIZE PAPER**

If you want more try —

Premium Quality COMSTAR T/F SUPER-10X PRINTER—$299.00
More Features Than RX-80

For $299 you get all of the features of the Comstar T/F plus 10" carriage 120 cps, 9 x 9 dot matrix with double strike capability for 18 x 18 dot matrix. High resolution bit image (120 x 144 dot matrix), underlining, backspacing, left and right margin settings, true lower descenders, with super and subscripts, and prints standard, Italic, Block Graphics, special characters, plus 2K of user definable characters. For the ultimate in price performance the Comstar T/F Super 10" leads the pack!

Double Immediate Replacement Warranty

We have doubled the normal 90 day warranty to 180 days. Therefore if your printer fails within "180 days" from the date of purchase you simply send your printer to us via United Parcel Service, prepaid. We will IMMEDIATELY send you a replacement printer at no charge via United Parcel Service, prepaid. This warranty, once again, proves that WE LOVE OUR CUSTOMERS!

15 DAY FREE TRIAL

OTHER OPTIONS

Extra Ribbons	$ 5.95
Roll Paper Holder	32.95
Roll Paper	4.95
5000 Labels	19.95
1100 Sheets Fan Fold Paper	13.95

Add $17.50 shipping, handling and insurance. Illinois residents please add 6% tax. Add $40.00 for CANADA, PUERTO RICO, HAWAII, ALASKA orders. WE DO NOT EXPORT TO OTHER COUNTRIES. Enclose cashiers check, money order or personal check. Allow 14 days for delivery, 2 to 7 days for phone orders, 1 day express mail available!! Canada orders must be in U.S. dollars.

PROTECTO
ENTERPRIZES (WE LOVE OUR CUSTOMERS)
BOX 550, BARRINGTON, ILLINOIS 60010
Phone 312/382-5244 to order

SUPER-10" ABCDEFGHIJKLMNOPQRSTUVWXYZ
ABCDEFGHIJKLMNOPQRSTUVWXYZ 1234567890

Four Tronix Games For VIC And 64

Dan Gutman

Have you wiped out any Killerwatts lately? If not, you may want to pick up a copy of *Juice* for your Commodore 64, one of four new games from Tronix, which last year blessed us with *Kid Grid*, the fill-in-the-blanks game for the Atari 400/800. *Juice* (designed by *Kid Grid*'s designer Arti Haroutunian) is the most exciting game in the new batch. It's a good takeoff, but not a rip-off, of the hit arcade game *Q*Bert*. The other new Tronix games are *Deadly Skies*, *Scorpion*, and *Gold Fever*, all for the VIC-20 only.

Juice

In *Juice*, as in *Q*Bert*, you are a helpless but resourceful character who relentlessly hops around a grid trying to change the appearance of all the blocks by landing on them. "Juice" refers to electricity; if you hit all the blocks without getting killed by various nasties, you have completed a circuit board and you move on to the next level.

While *Q*Bert* hops on a pyramid, Edison, *Juice*'s character, leaps about a slanted rectangle. The configuration is slightly different for each of the six levels. If you get into trouble – and you will – you can jump off the grid onto a line which will cause you to materialize elsewhere.

This game is a knockout. The characters are human-looking, not some videogame stick figures. The graphics are top-notch, and the play action is fun for five minutes as well as five months. It is available on disk for the Commodore 64 and for Atari computers with 32K.

The other three new games from Tronix are very different from one another, seemingly intended for specialized audiences. Still they do have one thing in common – simple directions. Just pop that cartridge in and play.

Deadly Skies

Here's your shoot-em-up. Underneath a protective cloud cover are a missile base, tanks, and missile emplacements set up to destroy your copter. UFOs, smart bombs, and asteroids harrass you in the air. You've got to bomb through the cloud cover, then bomb the ground targets which scroll by from left to right. Unfortunately, your copter cannot scroll off the screen with the targets, nor can you wrap around and come out the other side.

The sound effects are quite good, reminiscent of the shooting in *Centipede*. Although you can barely make out the shapes of your enemies, the overall graphics are adequate.

You will be challenged. Around level four there is so much debris in the sky, you feel like you're attacking Mount St. Helens. It is a good test of peripheral vision, though it is bothersome that the enemy missiles suddenly appear a quarter of the way up the screen, which happens to be your ideal bombing altitude.

Scorpion

Scorpion is another fast-action shooting game, but it requires some thinking, not just twitching. You've got eight different things to shoot at (Venus's-flytraps,

In Juice *for the Commodore 64, the player controls a figure who jumps around a power grid, pursued by "Killerwatts" (artist's rendition of actual screen).*

worms, pod babies, etc.), each with a different point value. More importantly, you've got to feed your clan by stunning frogs and bringing them home.

This is the fastest-moving of all the Tronix games, and you should have fun fighting your way through the maze, wreaking havoc, shooting everything in sight. The program makes pretty good use of the VIC's graphics capabilities, and the screen scrolls in four directions.

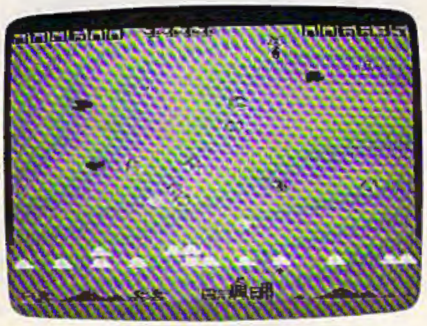
There is a scanner on the side of the screen that reports your location, but if you glance at it you're probably a goner. It might be an improvement to eliminate the gauges and make the playfield larger.

There are 32 levels and three difficulty settings, so you will be occupied for the next few months. Designer Jimmy Huey has also given us left- and right-handed play, and a freeze option.

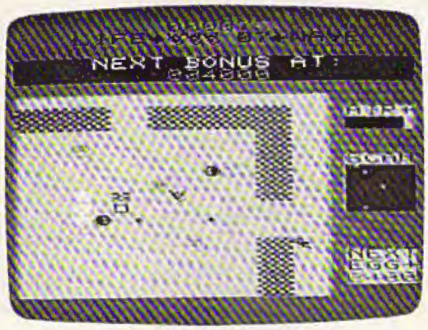
The instructions neglect to say where the scorpion "house" is, but I'll tell you – it's that black square in the center of the screen.

Gold Fever

In *Gold Fever*, you are inside an abandoned mine filled with glittering piles of gold. But



The helicopter (right-center) bombs ground targets in Deadly Skies for the VIC-20.



In Scorpion for the VIC-20, enemy creatures abound in a four-way scrolling maze.

Your miner (upper-right) must avoid hazards while gathering riches in Gold Fever for the VIC-20.

some of the screen objects are difficult to recognize. The gold could just as easily pass for small sleeping rodents. Boxcars are chasing you, and so is an evil Claim Jumper, who looks more like a fire hydrant with a tabletop.

Anyway, after you pick up all the gold in one mine, you move to another. There are nine difficulty levels, distinguished mainly by the amount of gold and speed of action, and only two alternating scenes.

Of the four new games from Tronix, the standouts are *Juice*, *Deadly Skies*, and *Scorpion*.

Tronix Publishing, Inc.
701 W. Manchester Blvd.
Inglewood, CA 90301

Juice
Commodore 64 disk $34.95

Deadly Skies
Scorpion
Gold Fever
VIC-20 cartridges $39.95

Rat Hotel For VIC-20

Clark L. and Kathryn H. Kidd

Like mold on cheese, *Rat Hotel* tends to grow on you after a while. Our copy of the new hide-and-seek VIC-20 cartridge from Creative Software didn't come with instructions, and without the benefit of a press release we might have had trouble learning how to play. But once you know the rules, *Rat Hotel* is easy enough to entertain even younger players.

The object is to maneuver Ermine the Rat down six floors of the Hotel Paradisio to consume Le Grand Cheeseball in the basement. Ermine's enemy is Waldo the Maintenance Man. Waldo, armed with a gun, chases Ermine around the hotel.

Rat Hotel is a game of several challenges. One, of course, is Waldo the Maintenance Man. There are also rattraps scattered throughout the hotel, which Ermine must avoid. The third obstacle is time: players have three minutes to guide Ermine from the hotel attic to the cheeseball. If and when the rat consumes Le Grand Cheeseball, the game starts again at a more difficult skill level.

Along the way to the basement, cheese is hidden in various corners. When Ermine consumes this cheese, points are added to your score. The rat must eat all the cheese on each level to move from one floor to another. Occasionally the cheese changes color from yellow to black. When the cheese is black, it paralyzes the rat for a few crucial seconds. During those seconds, Waldo may sneak up and shoot the rat. Never fear, however: a player is allowed three rats per game.

Playing *Rat Hotel*

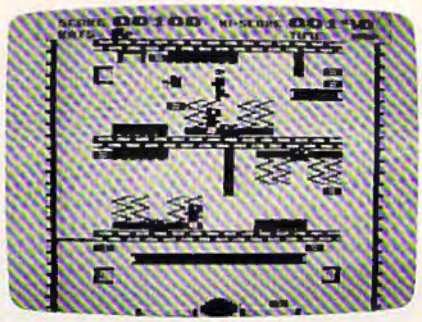
For those who might not learn how to play *Rat Hotel* without printed rules, here they are. *Rat Hotel* is a cartridge game requiring a VIC and a joystick. Plug in the cartridge, turn on the VIC, and jiggle the joystick to center the title screen on your TV. Choose your skill level by pushing the joystick up. Press the fire button to start play.

Once play begins, the top of the screen records your current score as well as the high score of games in the series. Little rats on the upper left side indicate the number of rats left in a game. On the upper right is a black bar showing the amount of time a player has left. Rather than giving the time in minutes and seconds, the black bar shrinks to show the remaining time. This is a clever feature; it's easier to gauge the length of the bar out of the corner of your eye than it is to take your eyes off the game to read a numeric display.

There are five skill levels, and the only significant difference is the speed of the game. Level 5 isn't unreasonably fast unless you've just finished playing a game on Level 1.

Ermine must avoid Waldo by staying ahead of him or by finding places to hide. When the rat has eaten all the cheese on a floor, move him to the extreme right or left of the screen and then push him as far down as possible to await the elevator. As you press the joystick toward the edge of the screen, a beeping sound will call an elevator to take Ermine to the next floor.

While you can move the rat only one floor at a time, Waldo can skip floors. Sometimes you



Waldo the Maintenance Man chases Ermine the Rat in Rat Hotel.

think you're well ahead of him, but he'll catch up to you. More rats "bite the cheese" while waiting for the elevator than in any other *Rat Hotel* situation.

Good Features

The idea of having an elevator that makes you wait is ingenious. It adds a little variety to the game. Another good feature of *Rat Hotel* is the sound effects, which are excellent. When the game begins, a Bach tune is played. After a while, though, the song does get a little tedious, especially if you're not a good player and have to keep starting over. There are also little tunes played when you win or lose.

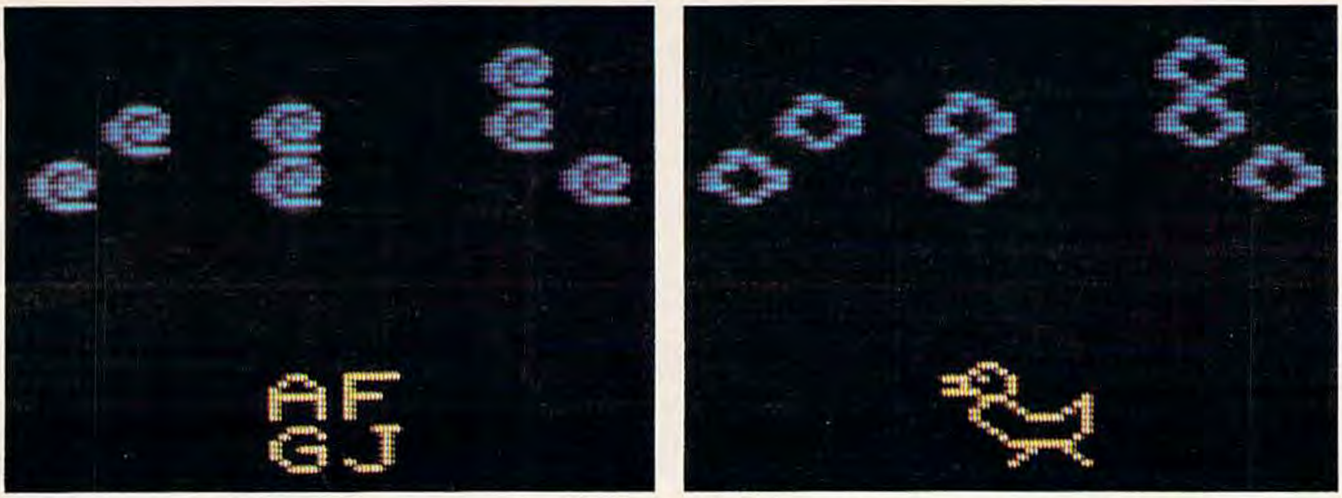
When Waldo shoots the rat there's a popping sound, and when the rat gets caught in a trap there's a definite crunch. There's a beeping noise that calls the elevator. Different sounds are made when the rat eats good or bad cheese. A lot of effort was put into the sound effects, and it makes the game more enjoyable.

The graphics of *Rat Hotel* are above average, perhaps even excellent. Graphics are not the strong feature of the VIC-20, but the graphics in *Rat Hotel* are good enough to add to the enjoyment of the game.

For all its innovation, however, *Rat Hotel* is a game of few

surprises. The rat always starts the game in the same location, cheese and traps are hidden in the same spots every time, and Waldo plods the same basic route each time the game is played. Even the poisonous black cheese shows up only once in most games, and that's at a specific spot in the basement. Thus, all games are basically the same, and after a few dozen rounds the player begins to feel like a rat in a maze. Some randomness introduced into the program would have made *Rat Hotel* an even better game.

Rat Hotel
Creative Software
230 Caribbean Drive
Sunnyvale, CA 94086
$39.95



A close-up screen photo showing how ordinary characters (left) are transformed into custom characters (right) for the VIC-20 version of "Chicken Little," a game program elsewhere in this issue. Note the technique of combining one or more characters (A, F, G, J) to form a single, large custom character.

Introduction To Custom Characters For VIC And 64

Tom R. Halfhill, Editor

What are "custom characters"? Why might you want them? Are they hard to program? How do they work? This introduction to the concept of custom characters answers all these questions and more. Two other articles in this issue then show exactly how to program custom characters on either a VIC-20 or Commodore 64.

Perhaps you've admired the screen graphics of a favorite arcade-style game, or the Old English letters of a Gothic text adventure. These kinds of shapes and special characters are not built into the computer itself. Maybe you've wondered how these effects are achieved and if they are difficult to program.

The secret is a technique called *custom characters,* also known as *redefined characters* or *programmable characters.* The terms are almost self-explanatory — with programming, you can design your own shapes and special characters to display on the TV screen. They can be almost any shapes you want: spaceships, aliens, animals, human figures, Old English letters, anything. In effect, you are *customizing* or *redefining* the characters already built into the computer.

For instance, if you redefine the letter A to look like an alien creature, every time you PRINT A on the screen you'll get the alien instead of the letter. Animation is as easy as erasing the character — by PRINTing over it with a blank space — and then PRINTing it in the next position. When this process is repeated rapidly, the alien seems to move across the screen.

Custom characters are especially useful to game programmers, but also are fun to experiment with for anyone interested in programming.

Character Sets

First, let's clarify exactly what is a *character set.* Briefly, it is the complete set or collection of characters that a particular computer can display on its video screen. *Characters* include letters of the alphabet (both upper- and lowercase), numbers, punctuation marks, symbols, and — on the Commodore 64 and VIC-20 — the 64 special graphics characters that are pictured on the front of the keys. In all, a VIC or 64 has a standard character set of 256 characters. This is the total set of characters which the computer is capable of displaying.

The character set is built into the computer, permanently stored in Read Only Memory (ROM). ROMs are memory chips that retain important information even when power is turned off between sessions. The character set is stored in ROM as a list of numbers. The numbers describe to the computer how each character is formed from a pattern of tiny dots.

You may be able to see these dots if you look very closely at your computer screen. (The dots might be too small to discern on some ordinary TV sets, but they are much more visible on a monitor.) All the characters in the character set are made up of these dots. The dots for each character are part of an 8-by-8 grid, for a total of 64 dots per grid. This method of forming characters is familiar to anyone who has seen the large time/temperature clocks on banks, or the scoreboards in sports stadiums. A computer displays characters the same way, except instead of light bulbs, the dots are very small pinpoints of glowing phosphor on the TV picture tube. (Figure 1 shows the dot pattern for the letter A on a Commodore 64; the dot pattern is slightly different for the VIC-20, but the principle is the same.)

Figure 1: Dot Pattern For Character A On Commodore 64

The character set is always kept in ROM, ready for the computer to use. Let's say you display a character on the screen — for instance, the uppercase letter A. The computer refers to the character set in ROM to see how it should display the A on the screen, much as you would refer to a dictionary to see how to spell a word. Once it looks up the dot pattern for an A, the computer displays the character. The whole process takes only a few microseconds, and happens every time a character is displayed, either by typing on the keyboard or using a PRINT statement in BASIC.

When the computer's ROM chips are preprogrammed for you at the factory, these dot patterns for each character are permanently burned into the chips so the computer will always display the same character set. Short of replacing the ROM chips themselves, there is nothing you can do to change this preprogramming. Normally, this would limit you to the built-in character set. Indeed, on some computers there is no alternative.

Fooling The Computer

However, on the VIC and 64 — and on many other home computers — there is a way to modify the character set to suit your own needs. The technique requires fooling the computer.

Here's how it's done. The first obstacle to overcome is the preprogrammed ROM chips. It is not possible to erase or change information in ROM. But remember, there are two types of memory chips in computers: ROM and RAM.

RAM (Random Access Memory) is temporary memory that *can* be erased and changed. Programs loaded from disk or tape, or which you write yourself, are stored in RAM while they run. They can be changed at any time from the keyboard, or even erased altogether by typing NEW or switching off the computer. RAM is the computer's workspace.

So, the first step toward custom characters is to copy the list of numbers representing the character set from ROM into RAM.

This is a relatively simple programming task. You find out exactly where in ROM the character set is stored by looking at a *memory map*, a list of memory addresses inside the computer. (Memory maps are often found in reference or owner's manuals or magazine articles.) Once you know the beginning memory address of the ROM character set, you can write a short routine which reads the list of numbers in ROM and then copies it into RAM. In BASIC, this is done with PEEKs and POKEs within a FOR/NEXT loop. One or two program lines are all it takes.

Now there's a copied image of the ROM character set in RAM. Again using POKEs, you can freely change the list of numbers to customize the characters any way you want (we'll cover this in detail in a moment).

OK so far, but there's one catch. The computer doesn't know you've relocated the character set. It still expects to find the character set where it always has, in ROM. It will continue to refer to ROM and will ignore your customized set in RAM.

That's why you have to "fool" the computer. The VIC and 64 contain a memory location, called a *pointer*, which points to the character set in ROM. Luckily, the pointer itself is in RAM. With a single POKE statement, you can change the number in this location to point to your custom character set in RAM, thereby fooling the computer into referring there for its information instead of ROM. The computer goes through its usual process of looking up the dot pattern for

We're Changing Perspectives on Software

INTERNATIONAL TRI MICRO

The profusion of software companies and products offers a disjointed, sometimes confusing choice of paths in exploring your personal computer. TRI MICRO wades through the confusion, offering you a singular unified approach through its Personal Library Series, each collection designed to help you discover the varied dimensions of your system.

The starter collection, the Desktop 5-in-1, provides you with a program set which includes text editing, file management, home accounting, electronic painting, and a video game for only $99.95. The Trilogy Series offers a more advanced productivity tool kit with integrated word processing, data base, and spread sheet capabilities for only $149.95.

Other collections introduce you to the worlds of computer entertainment and education. Individual programs are available on request.

Tri Micro . . . Software Innovators

each character and displaying it on the screen, except it looks up *your* modified pattern instead of the pattern preprogrammed at the factory.

Clever, eh?

Character Patterns

Basically, if you've made it this far, you've got the picture. But there are still a few details to clean up.

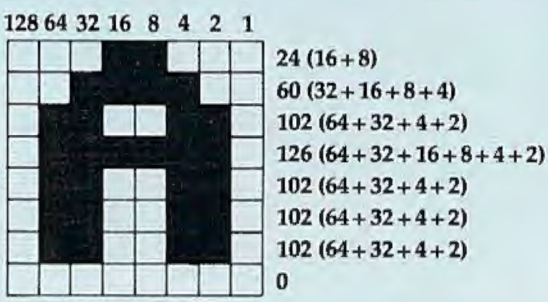
For example, exactly how are characters customized?

Recall that the character set is defined by a list of numbers which describes the dot patterns for each character, and that each character is formed by dots within an 8-by-8 grid. By changing these numbers, you change the shape of the dot pattern, and therefore the shape of the character.

It helps at this point to know something about the binary number system, so you might want to read "The Inner World Of Computers, Part 1: Binary Numbers" elsewhere in this issue. But even if you don't understand binary, look at Figure 2.

The eight numbers running vertically along the right side of Figure 2 are the numbers which define the dot pattern for an A on a Commodore 64 (again, the pattern for A on the VIC is slightly different). These are the same eight numbers which the computer refers to when it looks up A in the character set. They are also the numbers you must change to customize the character. These numbers are decimal versions of the binary dot patterns.

Figure 2: Dot Pattern For A On Commodore 64



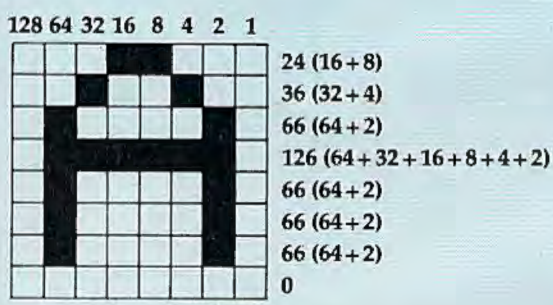
Along the top of Figure 2, running horizontally *from right to left*, are eight more numbers: 1, 2, 4, 8, 16, 32, 64, and 128. Notice that each of the numbers in this series is twice as much as its predecessor. That's how binary works. (If you want to get technical, each number represents a *bit* in a *byte*.)

Now, this is important: to understand how the numbers in the *vertical column* were determined, simply add up the numbers in the *horizontal row* which correspond to colored dots in the 8-by-8 grid. For example, the top row of the grid has two colored dots which form the peak of the A. (These are the same dots which will be lit up when the letter is displayed on the TV screen.) These two dots fall beneath the 8 and 16 of the top row of numbers. Because 8+16=24, the number in the right-hand column for that row is 24.

Likewise, the next number in the right-hand column is 60, because the colored dots in the second row of the grid fall beneath the 4, 8, 16, and 32, which add up to 60. And so on down to the very last row, which has no colored dots. This is represented by a 0 in the right-hand column. When the A is displayed on the screen, no dots will be lit up on this row of the grid. (All patterns for letters and numbers allow a blank line for the last row, and for the extreme right and left-hand columns, in order to keep the characters from running into each other on the screen.)

Figure 3 shows the dot pattern for an A on the VIC-20. Notice how the eight numbers in the vertical column are different from the eight numbers for an A on the Commodore 64. A comparison between the patterns of Figures 2 and 3 shows why: each computer forms its A in a slightly different way. Study these figures until you're sure you know how to add up the dot patterns to arrive at the eight numbers along the right. This is the key to customizing characters.

Figure 3: Dot Pattern For A On VIC-20



Customizing Characters

Once you understand how character patterns work, it's easy to customize them at will.

First, take some graph paper and mark off an 8-by-8 grid, or draw your own grid on a blank sheet. Along the top, write down the horizontal row of numbers as seen in Figures 2 and 3: 1, 2, 4, 8, 16, 32, 64, and 128. *Be sure to list them from right to left.*

Second, design your custom character by coloring in dots on the grid. Figure 4 shows a sample design for a *Space Invaders*-type creature.

Third, add up the colored dots in each row, starting from the top. Write down each sum in a vertical column along the right, as seen in the figures.

Figure 4: Dot Pattern For A Customized Character (*Space Invaders*-Type Alien)

You have now designed your own custom character. You can design as many of these as you'll need — up to the limit of 256 characters in the character set (although on the unexpanded VIC, memory limits make it practical to customize only 64 characters).

The only remaining step is to take the new series of eight numbers for each custom character and substitute them for the numbers in the standard character set. Remember, that's why you relocated the character set from ROM to RAM. Now that the list of numbers spelling out the patterns for the standard character set is in RAM, it can be changed to use your own numbers with POKE statements.

Specific Details

Up to now, this article has had to be fairly general in its explanations. The basic technique for customizing characters is the same for the VIC-20, Commodore 64, Atari, or any computer on which the character set can be relocated and redefined. But the specific details vary for each computer: the character set's memory address in ROM, how to safely copy it to RAM, the memory address of the character set pointer, the order of characters within the character set, and so on.

For these details, as well as example programs and utilities, turn to the article elsewhere in this issue which is appropriate for your computer: "How To Make Custom Characters On The 64" or "How To Make Custom Characters On The VIC."

Besides these articles, from time to time COMPUTE!'s Gazette will offer additional articles and columns on custom characters – one of the most powerful features available on your VIC-20 or Commodore 64.

It's time for your computer to grow up.

Meet PractiCalc.™ The world's most versatile spreadsheet at only $40.*

Games are fun when it's time to play. But at heart, your Commodore 64™ or VIC-20™ is a full-grown computer straining to gallop at the touch of your fingers.

Commodore 64 and VIC-20. Not just for games...

But the problem is, most software's been designed for toys. Not for real computers. Until now.

Now comes PractiCalc. And there's not much in the world of grown-up computing it won't do. It'll keep addresses (in alphabetical or numerical order), phone numbers (likewise), make budgets and menus, project profits, keep track of expenses, inventories, investments, what have you. For your home, your school, your business.

Budgets, inventories, projections, you name it...

Spreadsheet analysis...

PractiCalc gives you everything you'd expect from a spreadsheet, like adding and subtracting, multiplying and dividing, calculating square roots, logarithms, exponential numbers and even trig functions. And PractiCalc gives you a lot more.

Want to make a chart? Hit a key and the high or low resolution graphics of PractiCalc Plus or PractiCalc 64 will turn numbers into graphs. It'll even print out everything that shows on the screen.

High or low resolution graphics...

Want a lot of facts and figures at your fingertips? PractiCalc stores more than 2,000 cells in up to 100 columns and 250 rows. It'll sort them, search them, or shuffle them for you.

High-speed and wild card search...

You can add entries, delete them, or move them around —numerically or alphabetically, and instantly, of course. And you'll find PractiCalc unusually friendly, flexible, and forgiving.

All of which is a far cry from just playing with a joystick. Still, PractiCalc has one thing in common with a computer game. The price. Only $40.

Alpha or numeric sort...

PRACTICALC™ by **COMPUTER SOFTWARE** ASSOCIATES, INC.

Exclusively distributed by **Micro Software International, Inc.**
The Silk Mill, 44 Oak Street, Newton Upper Falls, MA 02164 • (617) 527-7510

*$40 is suggested retail price for tape version of PractiCalc-20 ($45 for disk version; PractiCalc Plus or PractiCalc 64 = $50 for tape version, $55 for disk).

How To Make Custom Characters On The 64

Gary Davis

Before reading this article, be sure to see "Introduction to Custom Characters For VIC And 64" in this issue, especially if you're unfamiliar with the concepts of redefined characters. The following article includes "Chred 64," a character-editing utility that makes the task of customizing characters easy and fun.

The Commodore 64 allows you to change any character in the character set to suit your own needs. In order to understand how this is done, it is first necessary to understand how the 64 (and most other computers) store the character set.

If you look closely at the letters the computer puts on the screen, you will notice that each character is made up of little dots in an 8×8 grid (see the figure).

Since there are 64 possible dots, or *pixels,* that can be either on or off, we need 64 "switches" for each character. This is done by using eight memory locations for each character. Since one memory location,or *byte,* is divided into eight *bits,* using eight bytes gives us the 64 switches we need for each character.

The bytes for each character are stored consecutively, with the first byte for each character representing the top row of dots in the character, the second byte the second row of dots, and so on. For a pixel to be on, the bit at its location must be *set;* for a pixel to be off, the bit must be *clear.* This is not as complicated as it sounds. The figure shows how the bit patterns of sets and clears are converted into the numbers that represent the character. When you make a series of bytes for every character and store them in a computer, you have what is known as a *character generator.*

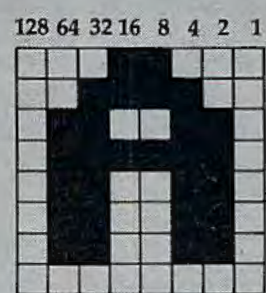
Relocating The Character Set

The character generator in most computers, including the Commodore 64, is stored in Read Only Memory (ROM). This way the computer is ready to display characters on the screen as soon as it is turned on.

Unfortunately, when the character generator is in ROM, you can't change the characters to suit your needs. When you can't change the existing character set, the simplest way to customize a new character set is to move it to Random Access Memory (RAM), and then tell the computer to use your character set rather than the one it has in ROM.

Telling the Commodore 64 where the new character set is located is relatively simple to do. Within the video controller chip (sometimes known as the VIC-II chip) is a special memory

Pixel Pattern For Letter A



16+8=24
32+16+8+4=60
64+32+4+2=102
64+32+16+8+4+2=126
64+32+4+2=102
64+32+4+2=102
64+32+4+2=102
=0

The character codes for A are stored in memory as: 24,60,102,126,102,102,102,0

For each row add the number at the top of each column in which a pixel is turned on. The resulting numbers are then stored consecutively from top to bottom.

location that allows you to set a new character pointer (the location of the first byte of your character set).

Now let's try an experiment. Type: POKE 53272,19 and press RETURN. Your screen will be

filled with strange characters, but don't worry. You have told the 64 to use a RAM character generator, but you haven't supplied one yet. To return your screen to normal type: POKE 53272,21 and press RETURN. You won't be able to read what you are typing until you press RETURN, but the computer understands. If this doesn't work, you can always restore the screen by pressing the RESTORE and RUN/STOP keys at the same time.

When you are designing a new character set, it is nice to have the normal one loaded into RAM to start with. Then you can make changes to it. Program 1 copies the 64's character set from ROM to RAM.

Before you type in this program, *you must enter:*

```
POKE 8192,0
POKE 44,32
NEW
```

This saves a place in RAM memory for your new character set and protects it from being overwritten by a BASIC program.

Now, type in the program and RUN it. After about 45 seconds the computer will come back and say READY. Now type POKE 53272,19 and press RETURN. Nothing appears to happen, but the characters you are now seeing on your screen are coming from your RAM character generator, not from ROM as usual.

To test this, type POKE 2056,255. The top of all the letter A's on the screen should now be a solid line. Try POKEing different numbers into memory locations between 2048 and 6143 and watch the results on the characters.

Using A Character Editor

By sketching an 8×8 grid as seen in the figure, it is possible to map out the entire character set on graph paper and convert your new characters to numbers to POKE into memory.

This method, however, can be both time-consuming and frustrating. A far better way is to create your new characters on the screen and let the computer do all the calculations. With this thought in mind, I wrote a character editor called "Chred 64." With this utility (Program 2), you can redefine any of the text or graphics symbols and save them on tape or disk. This can then be loaded and used with any program.

In order to reserve memory for the alternate character set, it is necessary to set the start of BASIC pointer to 8192. This will leave you with 32K of RAM free for your BASIC program. To do this, *you must* type in the following:

```
POKE 8192, 0
POKE 44, 32
NEW
```

"Chred 64," a utility program for creating custom characters on the Commodore 64.

Now the memory from 2048 to 8191 is free to hold your new character set. You may type in or LOAD Chred 64. After typing Chred 64 for the first time, be sure to SAVE it on tape or disk before you RUN it. If you have made a typing error, it is possible that the computer will "crash" and you'll have to type it all over again if you haven't SAVEd a copy.

When you run Chred 64, the program first copies the resident character set from ROM to RAM and resets the character base to point to the RAM character set. The program then expands the current character being edited to eight times its normal size.

To edit the current character being displayed, you may use the cursor control keys, the asterisk, and the space bar.

To turn on a pixel, position the cursor and press the asterisk. To turn off a pixel, press the space bar. To clear the entire character, press CLR.

To edit a different character, press F1. You will be asked to supply a row and column. This refers to the block of characters displayed on the lower-right corner of the screen. Just type a row number followed by the column number or letter. The character you selected will now be displayed, ready for you to edit.

More Editing Features

An interesting feature of the 64 is that, unlike the Commodore PET, the reverse-field (inverse video) characters are stored as part of the character set. This allows 256 redefined characters. To edit a character not being displayed, press F3. This will select and display the next block of 64 characters. Rest assured that you may mix characters from any of the blocks; only 64 characters are shown at a time for the purpose of editing.

Sometimes you may wish to edit more than

one character at a time to make a larger shape. This can be easily accomplished by pressing F5. Instead of a single character, you will be able to edit a block of four characters. To go back to single character mode, just press F5 again.

After you have redefined several characters, the text on the screen may become unreadable as your new characters replace the existing ones. To restore the character set to normal, *without* destroying your new character set, press F7. To return to your new character set, press F7 again.

When you are done working with a character set, you can restore the font to the normal character set by pressing R. You will be asked "Are you sure?" Now is your last chance to save your character set. If you are really done, press Y; otherwise, press N.

After you have gone to the effort of creating a new character set, you will probably want to save it on disk or tape for use in other programs. To save your character set, press S. Follow the directions given on the screen. After the character set is saved, you will be returned to the editor. (When typing Chred 64, omit line 225 for use with tape.)

Sometimes you may wish to alter a character set that you have already created and saved. To load another character set, press L and follow the directions given on the screen. Be careful — the new character set is loaded on top of the current character set, so be sure to save it if you want to use it later.

OK, you've developed your new character set. To use it with another program, you will have to type POKE 8192, 0:POKE 44, 32: NEW, just as you do when you load Chred 64. To load in the character set, place the cassette containing your new character set in the recorder, or the disk in the drive. For tape, type LOAD "filename",1,1 where "filename" is the name you gave when you saved the character set. For disk, type LOAD"filename",8,1. To use the new character set, POKE 53272, 19. To return to the normal character set, POKE 53272, 21.

I hope you have as much fun using this program as I had writing it. I would like to express my thanks to Don Moses for the use of his 64 in developing this program, and to the Central Ohio PET Users Group for providing detailed maps of the video chip registers.

If you do not want to type in this program yourself, please send a blank disk or cassette tape along with a stamped, self-addressed mailer and $3 to the address below. I will send you a copy of Chred 64 and a character set developed with it.

Gary Davis
1147 Carbone Drive
Columbus, OH 43224

See program listings on page 196.

EUREKA!

That's what we said when our new "invention" solved all our VIC-20™ and Commodore-64™ programming problems

We had a problem. So we invented PC-DocuMate™ to solve it. The problem was how to quickly master the VIC-20 and CBM-64 keyboards and easily start programming in BASIC on our new personal computers. First we went through the manuals.

INCONVENIENT MANUALS

The user's guide was a nuisance and the programmer's reference manual was just plain inconvenient to use. We found the control key combinations confusing and the introduction to BASIC to be too "basic" for our needs. We needed a simple solution to our documentation problems.

So we decided to surround the keyboard of each PC with the information we wanted. We decided to print whatever we needed on sturdy **plastic templates** which would fit the keyboard of either the VIC-20 or Commodore 64.

SIMPLE SOLUTION

This was the simple solution to our problem. Now we could have the essential information right at our fingertips.

On the left side and top of the templates we put **BASIC** functions, commands, and statements. On the lower left we used **key symbols** to remind us of how to use SHIFT, RUN/STOP, CTRL and the "Commodore" key. Over on the bottom right side we put some additional keys to help remember about CLR/HOME and RESTORE. But we were still a little confused.

STILL CONFUSED

We found we were confused about music programming, color graphics, and sprites. On both the VIC-20 and the CBM-64 templates we carefully organized and summarized the essential reference data for **music** programming and put it across the top—showing notes and the scale. All those values you must POKE and where to POKE them are listed.

Then to clarify **color graphics** we laid out screen memory maps showing character and color addresses in a screen matrix. (We got this idea from the manuals.)

For the VIC-20 we added a complete memory address map for documenting where everything is in an expanded or unexpanded VIC.

For the Commodore 64 we came up with a really clever summary table for showing almost everything you ever need to know for **sprite** graphics.

GETTING EASIER

Now we had organized the most essential information for our VIC and 64 in the most logical way. BASIC, music, color graphics, and sprites all seemed a lot easier. Our initial problem was solved by PC-DocuMate™

But we have a confession to make.

WE CHEATED

We had solved this kind of problem before. In fact, many times before. You see, we at SMA developed the original PC-DocuMate for the IBM PC. We've made templates for IBM BASIC and DOS, for WORDSTAR™, VISICALC™ and other best-selling software packages for the IBM PC.

So we knew we could invent another PC-DocuMate™ to solve our problems with the VIC-20 and Commodore 64. Now our solution can be yours and you can join the thousands of satisfied users of our template products.

Take advantage of our experience and success with PC-DocuMate templates. Get one for your personal computer.

SOME SPECIFICS

Our templates for the VIC and 64 are made from the same high quality **non-glare** plastic as the more expensive IBM PC versions.

The templates are an attractive **gray** color and are imprinted with a special black ink which bonds permanently to the plastic. They are precision **die-cut** to fit your keyboard.

Unlike some other products we've seen in this category, PC-DocuMate templates are professionally and expertly designed. And they are fully guaranteed.

OUR GUARANTEE

We guarantee your satisfaction. **You must be satisfied** with your PC-DocuMate for your VIC-20 or CBM-64. Try it for 10 days and if for any reason you are not satisfied return it to us (undamaged) for a full refund. **No risk.**

SOLVE YOUR PROGRAMMING PROBLEMS WITH PC-DocuMate™

Order your PC-DocuMate today (by phone or mail) and solve your VIC-20 or CBM-64 programming problems. Send only **$12.95** and specify which computer you have. We pay for shipping and handling. Use the coupon below or call **919-787-7703** for faster service.

YES! Please RUSH me ______ VIC-20 templates and/or ______ CBM-64 templates at $12.95 each. I have enclosed $__________ by:

Check ___ Money order ___ MC/VISA ___

Name

Address

City State Zip

Card # Exp.

Signature

Foreign orders (except Canada) add $5.00 US

Mail to: **Systems Management Associates**
3700 Computer Drive, Dept. J-1
P.O. Box 20025
Raleigh, North Carolina 27619

Canadians: Please send $18.95 CDN for each template to:

Systems Management Associates
55A Westmore Dr., Dept. J-1
Rexdale, ONTARIO M9V3Y6

Dealer inquiries invited.

How To Make Custom Characters On The VIC

Gregg Keizer, Assistant Book Editor

Before reading this article, be sure to see "Introduction To Custom Characters For VIC And 64" in this issue, especially if you're unfamiliar with the concepts of redefined characters.

The standard characters provided with the VIC are certainly useful, afford plenty of variety, and can be combined to create new shapes and figures. Many games on the VIC, for instance, often use only the standard character set to display the screen and show objects or user-controlled figures.

But there will be times when you need to draw a new shape or figure that the standard character set just can't produce. You'll often find this true as you design your own games. Or perhaps you simply want to experiment, to see what you can do with the VIC.

Creating custom characters takes up more memory, which can be crucial when you're using the VIC, and it can take time to design and add them to a program. But when you're looking for just the right figure, and it shows on the screen during a game, you'll agree that it was worth the effort.

Fooling The VIC

You've already looked through the article "Introduction to Custom Characters," and perhaps read "The Inner World of Computers, Part 1: Binary Numbers" elsewhere in this issue, so you know what custom characters are and how to design them using graph paper. Now that you have the figures in mind, you can actually begin to place those custom characters in the VIC.

Remember that the character set of most computers, including the VIC, is located in Read Only Memory, or ROM, and is permanently stored there. The VIC's character memory begins at location 32768, which stores the number value of the top row of the @ character. The number value of the second row of that character is at location 32769, the third row at location 32770, and so on.

In order to change the character set and insert your own custom characters, you first need to change the place where the VIC looks to find its character set. You can do this by changing the *pointer*, which fortunately is in a Random Access Memory (RAM) location. By changing this memory location, you are in effect instructing the computer to look elsewhere for its character set.

The VIC looks to location 36869 for its pointer. Although the pointer's value is usually 240 or 242, it can be changed by POKEing a new value into that location. Entering POKE 36869,255, for example, fools the computer into looking to a new location in RAM, 7168, for character data, instead of the ROM location 32768. You can begin your custom character set in a RAM location ranging from 4096 to 7168, but the best place to start is at 7168.

Right now, however, there is nothing there for the VIC to look at. You need to copy the character set data to your new RAM location. As explained in "Introduction to Custom Characters," this isn't that hard.

What you need to do is free enough space from BASIC to fit in your recopied character set, as well as protect it from the BASIC's operations. Then you have to tell the VIC to read the numbers in the character set stored in ROM and copy those numbers starting at your new RAM location, 7168.

A short routine such as the one below does all the work for you in only three program lines.

```
10 PRINT"{CLR}":POKE 36869,255
20 POKE 52,28:POKE 56,28:CLR
30 FOR I=7168 TO 7679:POKE I,PEEK(25600+I
   ):NEXT
```

Line	Function
10	Tells the VIC to go to location 7168 to get the data for its character set, rather than looking to ROM location 32768.
20	Frees 512 bytes of memory from BASIC by changing the pointers to the top of available RAM memory. A value of 28 takes 512 bytes, just enough for the 64 characters you would normally copy to RAM, and also protects this area from BASIC.
30	Copies the first 64 characters from their ROM locations to your new RAM location. This is done by PEEKing at the values from 32768 to 33280 (PEEK 25600+I), and then POKEing those values into the new locations running from 7168 to 7679. (We're moving only 64 characters out of a total character set of 256 in order to conserve memory.)

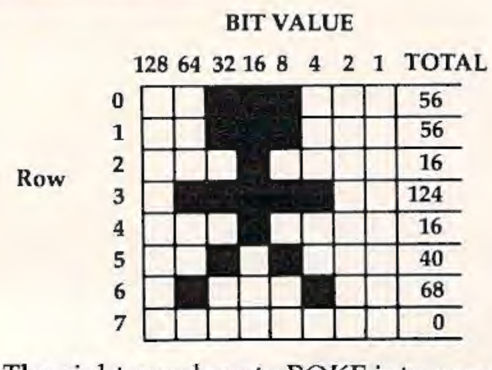
Now that you have part of the character set moved to RAM, the VIC told to look there from now on for its character data, and the area protected from BASIC, you can begin to place your own characters in this set.

Placing Your Custom Characters

Your custom characters have already been designed, either using graph paper or perhaps with the "VIC-20 Character Developer" utility program (see sidebar). You've added up the dot values and should have eight numbers for each new character. These represent the eight bytes of memory each character requires to be displayed on the screen.

What you now need to do is POKE these new numbers where the old values are, replacing one of the standard characters with one of your custom characters. For example, a custom character such as that in the figure would have the following values:

Custom Characters



The eight numbers to POKE into a memory location are 56,56,16,124,16,40,68, and 0 to create this character.

The most convenient way to replace old characters is with DATA and READ statements. By placing all the new numbers into DATA statements, and then having the VIC READ them, your programming task will be simplified. The computer always READs the DATA in the order it's listed, so be sure the numbers are in the right order, and that there are eight numbers for each character. The DATA statements can be anywhere in the program, as long as they are in the same order as the READ commands.

Glance through the Screen Code table in *Personal Computing on the VIC,* the guide that came with your computer. You've copied the first 64 characters, from the @ to the ?, into RAM, but you'll lose some of these when you replace them with your custom characters. Decide which standard characters you won't need, and look up the screen codes for those. If you are designing a game that uses some of the letters in a display, for example, make sure those won't be lost when you develop your custom characters.

Because you changed the pointer, your character set now begins at memory location 7168. Each character takes up eight bytes of memory, so by multiplying the screen code number by 8 and adding it to 7168, you can find the location of the top row of any character. For example, the letter A, with a screen code value of 1, begins at location 7176.

A format you can use to replace a standard character with your own custom character is:

FOR C(your new character)=X TO X+7:READ D:POKE C,D:NEXT

where X is the memory location of a character you want to replace. To replace the A character with the custom character from the figure, for instance, you would write:

FOR C=7176 TO 7183:READ D:POKE C,D:NEXT

and include somewhere in the program the DATA statement:

DATA 56,56,16,124,16,40,68,0

Add these two lines to the program used to copy characters into RAM, and you'll see the custom figure every time you press the A key. Notice, however, that you have lost the A character. There is now no way to print that on the screen. In other words, make sure that the characters you replace are ones you won't want to use.

If you have several new characters and they are replacing standard characters right after each other on the Screen Code table, you can place more than one in a READ statement, simplifying your programming. Replacing the first five standard characters, for example, would look like this:

VIC-20 Character Developer

E. A. Cottrell

The utility program described here takes some of the tedium out of customizing characters for the VIC. It automatically converts the binary dot pattern numbers into decimal numbers for you.

Creating custom characters on the VIC-20 is not difficult. Eight bytes of information are in ROM for each character. To create custom characters, you must change the location to which the VIC looks for the characters to a location in RAM, and then POKE the desired information into the memory addresses normally used by the VIC. I won't go into detail here; see the related articles in this issue. You can also find more information in the *VIC-20 Programmer's Reference Guide.*

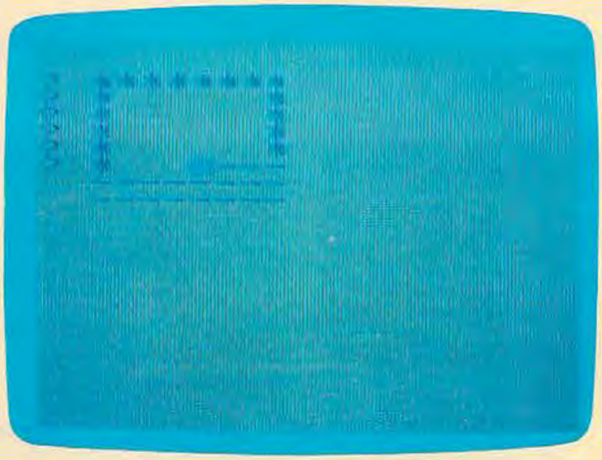
Relieving The Tedium

The most tedious part of this process for me is converting the binary representation of the characters into decimal numbers which can be POKEd into memory. The short utility program with this article eliminates that problem and speeds up the whole character development process. It also provides the hexadecimal equivalent for machine language programmers.

This program uses the full screen. Therefore, care must be taken when typing, especially with the PRINT statements. It is very important that all semicolons and commas be entered as listed.

When the program is first run, the screen will display eight rows of eight dashes in the top left corner, with a question mark in front of the first row. To design a custom character, enter asterisks (*) in place of the dashes (—), then press RETURN. The question mark then moves to the front of the second row. Repeat the process for all eight rows, creating your character with the asterisks as you go.

When RETURN is pressed after the eighth row is entered, the decimal and hexadecimal equivalents of each row are displayed. Below this, the custom character is shown, then the prompt "PRINT-OUT (Y/N)?". Pressing the Y key at this time will dump the screen to the printer. If you do not have a printer, lines 360-380 and lines 500-560 may be omitted.



Designing a hollow box with the "VIC-20 Character Developer."

Pressing the N key in response to the print-out prompt leads to another prompt, "ARE YOU DONE (Y/N)?". Pressing N here produces the prompt, "NEW CHARACTER (Y/N)?". A Y response clears the screen and gives you a clean grid with which to create your next character. Pressing N leaves the character you are working with in the grid and allows you to change only the parts desired. Remember that RETURN must be pressed for all eight rows whether all are changed or not.

The print-out can be very useful when designing characters which will face different directions. Simply turn the copy to the angle desired and enter the new character.

This program is more than a useful tool. Kids have great fun creating all sorts of things. If you answer Y to the DONE prompt, the @ key will display the new character. To restart the program, enter RUN 30 to keep from waiting for the transfer of the character set from ROM to RAM.

See program listing on page 195.

Sample Screen Dump To Printer

```
                DEC    HEX

? --****--      60     3C
? -*----*-      66     42
? *-*--*-*      165    A5
? *------*      129    81
? *-*--*-*      165    A5
? *--**--*      153    99
? -*----*-      66     42
? --****--      60     3C

     @

  PRINT-OUT  (Y/N)?
```

FOR C=7168 TO 7207:READ D:POKE C,D:NEXT

You would then need five DATA statements, one for each new character created.

8K Expanded VIC

If you have an expanded VIC, with 8K or more of RAM, you'll need to enter additional commands before you run any program which copies a character set and creates custom characters.

"Creating Graphics on the Expanded VIC," by Ed Harris, in the February 1983 issue of COMPUTE! Magazine, includes a short machine language program that allows you to create custom characters on your expanded VIC.

If you don't want to use this utility, you can enter a few additional POKEs *before* you load and run any program creating custom characters. Enter each individual POKE, then press RETURN:

```
POKE 43,1:POKE 44,32
POKE 8192,0:NEW
POKE 36869,240:POKE 36866,150
POKE 648,30
```

The first line of POKEs sets the pointer to the start of the BASIC program, much like the POKE 52,28 did in the unexpanded VIC. The second POKE, the first memory location of BASIC, must be set to 0, or you won't be able to run your programs. The third line of POKEs relocates the screen, while the last POKE makes it possible for the operating system to see the screen.

As you enter the last two lines of POKE statements, the screen will change drastically. Don't worry — you haven't done anything wrong. You do, however, need to be careful as you enter these lines, for you can't really see what you've typed on the screen, due to the jumbled display.

Once these are entered, you can LOAD and RUN your program to copy characters and create custom figures. Line 20 in the program, used to copy characters to RAM, must be eliminated, however, if you use the expanded VIC. If you leave it in, the pointers in BASIC will change again, and you won't see the correct screen display.

Custom Hints

You now have the ability to design and enter your own custom characters on the VIC. Refer to the accompanying sidebar, "VIC-20 Character Developer," for a simple utility which will make it easier to design custom characters. This utility will also calculate the byte values of each row of a character for the DATA statements you'll be placing in your programs.

As you type in the lines for your custom characters, remember several things.

Each character needs eight numbers in the DATA statement to define it, even if some of the values are 0. If you don't have all eight numbers, the VIC will READ from the next DATA statement and your figure will not appear as you expected.

List the DATA statements in the same sequence as the READ commands used to replace the characters. The first READ statement looks to the first DATA statement, the second READ command looks to the second group of data numbers, and so on.

When you're finished, RUN the program to make sure the custom characters replace the right standard characters and appear as you wanted. Check the DATA statements, the byte values of the new characters, and the FOR C=X TO Y statements if you have problems. More than likely, any errors can be found in these places.

Users Groups

COMPUTE!'s Gazette is compiling a nationwide guide of VIC-20 and Commodore 64 users groups which will be published periodically. Please send us your group's name, address, the name of a contact person, and a phone number (optional). Also include any other relevant information: the group's membership, when it was founded, whether it operates a phone-in bulletin board system for modem users, etc. Address to:

Kathy Yakal, Editorial Assistant
COMPUTE!'s Gazette
P.O. Box 5406
Greensboro, NC 27403

VIC/64 Program Lifesaver

"UNNEW" Rescues Lost Programs

Vern Buis

If you have ever lost a BASIC program by accidentally typing NEW, then read on. This short machine language routine for both the Commodore 64 and VIC-20 (any memory size) provides an easy means of recovering BASIC programs that have been "erased" — and it loads and executes in only ten seconds.

Sooner or later — practically every programmer does it — thinking a program has been saved, you type NEW to clear out the memory, and a split-second after pressing RETURN, you wind up screaming.

But on the VIC-20 and Commodore 64, typing NEW does not really erase the program from memory. NEW just makes the computer (and the programmer) *think* the program is gone. As long as you don't start typing another program or switch off the machine, the program is still there. To get it back, all you have to do is fool the computer into remembering where in its memory the program begins and ends.

That's what "VIC/64 Program Lifesaver" does. By loading and running this short machine language utility immediately after committing the grievous error, you can save your lost program, save your hours of work, and even save your sanity.

Entering The Lifesaver

The Lifesaver is listed as a BASIC loader, a BASIC program that creates a machine language program. The same listing works for either the VIC-20 or Commodore 64. Be sure to read the following special instructions before typing the program. The procedure is somewhat different from most and requires that certain steps be followed exactly.

First, if you are using tape instead of disk, enter line 60 as follows:

60 CLR:SAVE"UNNEW",1,1

After typing the listing, *do not RUN it*. Instead, save it on disk or tape with a filename such as "LIFESAVER/BASIC" or "UNNEW/BASIC". Do not use the filename "UNNEW" This filename must be reserved.

Now enter RUN. The BASIC loader creates the machine language program and automatically saves it on disk or tape under the filename "UNNEW". This is what you'll actually use to rescue lost programs; the BASIC loader can be set aside as a backup in case you need to create another copy.

Using The Lifesaver

OK, let's say you've just typed NEW and wiped out hours of valuable labor. (To test the Lifesaver, you can load a BASIC program and erase it with NEW.) Recovering it is easy.

To load the Lifesaver from tape, enter:

LOAD"UNNEW",1,1

To load the Lifesaver from disk, enter:

LOAD"UNNEW",8,1

Either way, it loads pretty fast, because the program is short. Now, to activate the Lifesaver, enter:

SYS 525 [RETURN]
CLR [RETURN]

(Incidentally, CLR means to type the keyword CLR, not to press the CLR/HOME key.)

That's all there is to it. When you enter LIST, the BASIC program you thought was forever lost at sea is back, safe and sound.

The Lifesaver itself also remains in memory, but probably not for long. It's tucked away in memory which is unprotected (locations used by the input buffer and BASIC interpreter), so you'll have to load it again each time you want to use it. But unless you're either very unlucky or (shall we say) prone to inadvertent actions, the Lifesaver isn't something you should be needing often.

Why It Works

Instead of erasing the program in memory when you type NEW, the VIC and 64 simply reset two key pointers in such a way that the operating system doesn't "see" that the program is still there. These pointers keep track of where in memory a BASIC program begins and ends. NEW moves the top-of-program pointer down to the bottom of BASIC memory, and the first two bytes of BASIC memory are set to zero. These first two bytes serve as a pointer to the address for the second line of BASIC code. When they are set to zero, the operating system believes that no program is in memory.

The Lifesaver works by skipping the first two bytes of BASIC memory (the address pointer) and the next two bytes (BASIC line number). It scans upward for a zero byte — the end-of-line indicator. Upon finding the zero byte, the routine POKEs its address, plus one, into the second-line-of-BASIC address pointer. One of the erased pointers is thereby restored.

Next, the Lifesaver scans byte-by-byte through the BASIC memory area until it finds three consecutive zero bytes. This is the end-of-program indicator. Once it locates these zeros, the routine POKEs the address of the third zero, plus one, into the top-of-BASIC/start-of-variables pointer at locations 45-46. This completely restores the erased program.

For those who might want to relocate the Lifesaver to a safer memory area — to preserve it for frequent use or to combine it with other utility routines — the machine language program is written to be fully relocatable. It uses no absolute JMP or JSR instructions. The area used here was chosen to make it load easily into a 64 or VIC with any memory configuration and to minimize the danger of it loading atop a BASIC program.

See program listing on page 203.

Understanding Sound On The 64

Part 2

Gregg Peele, Editorial Programmer

Ever wished you could create just that right sound for a game effect? Or that right tone for a song? The conclusion of this two-part article and the accompanying utility program may be just what you need to create interesting new sounds on your 64.

Last month we explored some of the basics of producing sound on the Commodore 64. We discussed ADSR (attack, decay, sustain, and release) and used these parameters along with volume, pitch, and waveform to produce various sounds. This month, we will look even further into the capabilities of the 64's built-in "synthesizer on a chip," the Sound Interface Device (SID). We'll discuss filters, ring modulation and synchronization, and present a utility, "Soundmaker 64," which will make it easier to use these techniques within your own programs.

Changed Your Filters Lately?

The Commodore 64 SID chip has three filters — but unlike the filters in your car, they should never need replacing. However, they do share some similarities with car filters. Just as an oil filter allows oil to pass while blocking out other unwanted particles, the SID chip filters let parts of sounds pass — selectively *filtering* out the remainder of the sound. Synthesizer filters provide an important means of manipulating sounds to produce various effects.

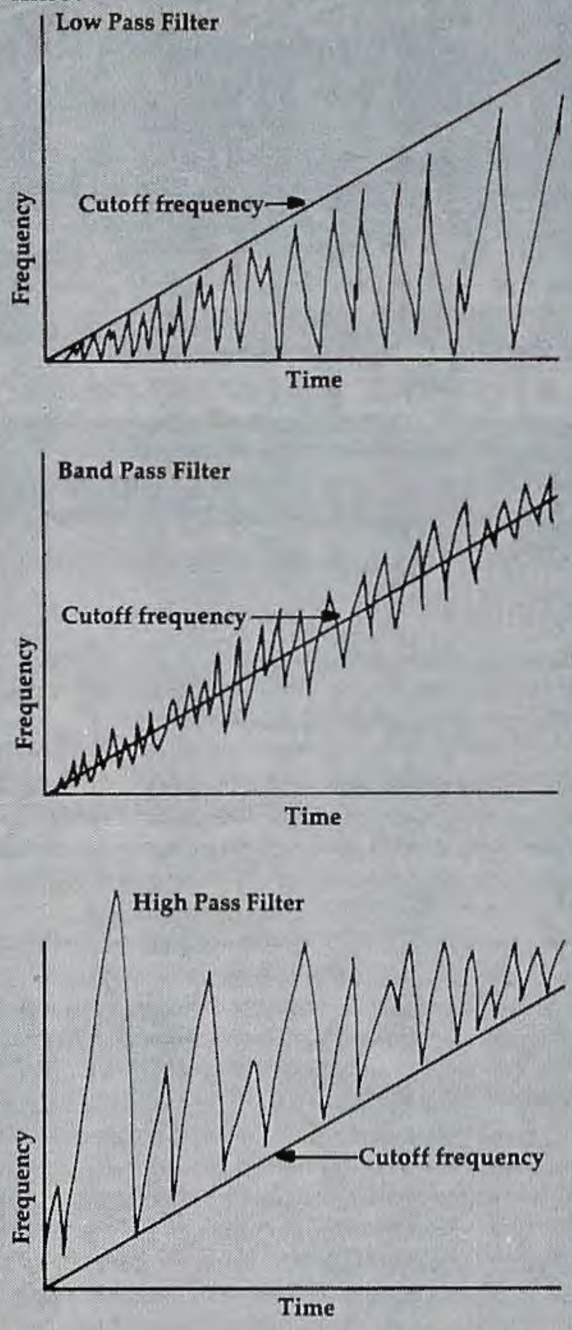
The three filters are called *high pass, low pass,* and *band pass.* The high-pass filter is designed to remove the lower frequencies, letting the higher frequencies pass. The low-pass filter has the opposite effect — it removes the high frequencies while allowing low frequencies to pass. The band pass filter allows a band or group of frequencies to pass through while frequencies above and below the band are suppressed.

The filter you choose is activated by turning on bits 4 (low pass), 5 (band pass), or 6 (high pass) in SID register 24 (read the accompanying short article, "Switching Bits," for details on turning bits on or off). These filters can be used in combinations for additional effects. For instance, adding the low and high pass filters together creates the inverse effect of the band pass filter; only the higher and lower frequencies pass, suppressing the middle frequencies.

The amount of sound that is removed by a filter is determined by the *cutoff frequency.* The filter cuts off the sound beginning at this frequency. The cutoff frequency for filtering is controlled by the lower three bits in SID register 21 and all eight bits in register 22. Some of the most interesting effects possible on the 64 are created by incrementing or decrementing these series of bits while a sound is being played. Want the sound of an alien ship as it lands? Use your normal alien ship sound, add a filter, and gradually increment or decrement these eight bits as your ship descends. A certain combination of waveforms and a changing filter can create just the right sound effect for a descending alien ship.

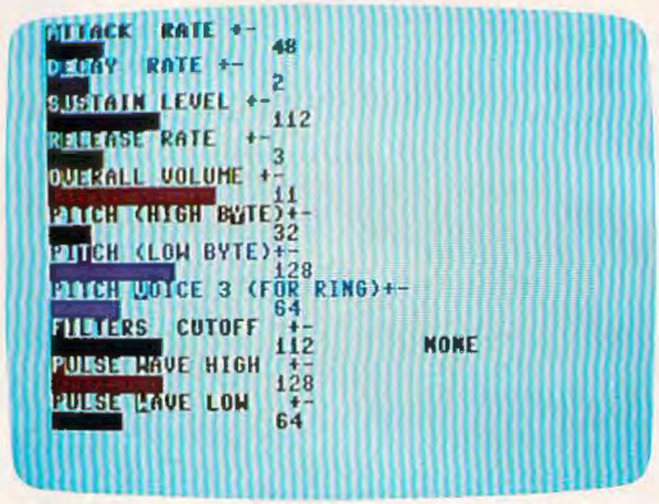

READ THE LABEL
THE FILING ASSISTANT
DATA BASE MANAGEMENT for the COMMODORE 64
This data management package allows you to create record formats, sort information, and produce reports. It includes the following features:
Design records up to 256 characters, up to 25 fields, (up to 35 characters per field)
Designate up to 10 two-level sorts
"Step through" records while displaying selected fields to screen
Add, copy, change, or delete records
"Find" a record by key field, or "locate" by record number
Set function keys to insert a frequently used word, phrase, zip code, etc.
No programming or code necessary. On-screen prompts lead to full operation in one to two hours.
THE FILING ASSISTANT contains a built-in mail label program that allows you to print a 1 to 6 line label. You can include more than one field per line and print up to four labels across the page. Labels to be printed can be selected on one or more fields.
The program also allows you to design your own report formats. Your reports may include:
Automatic record counting
Computation between fields
Totals and/or sub-totals per column
Record selection by one or more fields
TECHNICAL NOTE
This program supports the serial or the user port —you can configure just about any printer. It can be used with more than one 1541 disk drive or with the "C-64 Link-IEEE Interface" which allows you to work with the Commodore 2031, 4040, and 8050 disk drives.
THE FILING ASSISTANT is part of a series. It will integrate with THE WRITER'S ASSISTANT (word processing) and THE SPREADSHEET ASSISTANT (forecasting and financial analysis).
Even if you are a beginner THE FILING ASSISTANT will allow you to manage personal and business records. The ASSISTANT SERIES provides you with the most complete and practical information management package available for the Commodore 64.
RAINBOW Computer Corporation
490 Lancaster Pike
Frazer, Pennsylvania 19355

Filter (Noise Waveform)
Cutoff Frequency is incremented through time.



Additive And Subtractive Synthesis

Filtering is an example of *subtractive synthesis.* Subtractive synthesis is a method of manipulating sounds by subtracting parts of a single sound — pushing other parts which normally may not be heard into the forefront. *Additive synthesis,* however, brings two sounds together to form a totally new sound. Both *ring modulation* and *synchronization* are examples of additive sound synthesis.



"Soundmaker 64" lets you experiment with all the parameters of the sound chip and add the results to other programs.

Ring Modulation

Ring modulation is a form of additive sound synthesis that dramatically changes the timbre or tone quality of two tones. Tones that have been fed through a ring modulator do not retain their original pitches or timbres. Instead, the sums and remainders of the two frequencies are retained. For instance, if the first sound is a tone that vibrates at 100 vibrations per second (vps), and the second tone vibrates at 200 vps, then the ring-modulated tone will be a combination of the sum (300 vps) and the difference (100 vps).

Usually the ring-modulated tone sounds very different than the two original tones. Since most tones are complex phenomena consisting of many less obvious inner frequencies (harmonics), the ring-modulated tone may be very complex in tonal character.

To achieve ring modulation on the 64, you have to set bit 2 of the waveform byte when using the triangle waveform (POKE register 4 with 21). Voice 3 must be set to some frequency. No other parameters of Voice 3 have any effect on ring modulation.

Synchronization on the 64 also adds two tones together to produce a new and different sound. If bit 1 of the waveform byte is set (POKE register 4 with 19), then setting Voice 3 to a definite pitch (POKE registers 14 and 15 for the pitch of Voice 3) and manipulating the pitch of Voice 1 (registers 0 and 1) cause the tone quality of the resulting pitch to change.

Synchronization happens when the two waveforms are linked to make the waveform of Voice 1 dependent on whether it is *in sync* with the frequency produced by Voice 3. Since the two waveforms are not usually in sync, the waveform is distorted, producing different and sometimes interesting waveforms. In sync mode, the pitch of

Switching Bits

Registers are simply memory locations that have special functions. In the SID chip, there are 28 registers, most of which are dedicated to sound production. In Soundmaker 64, the registers we POKE do not begin at the sound chip (54272), but at 49152. The machine language routine then "mirrors" these registers by copying them to the sound chip registers.

To understand how registers can be manipulated, a brief discussion of bytes is necessary. Bytes are memory locations that can store values. On the Commodore 64 and most other small personal computers, bytes consist of eight smaller units called bits (binary digits). A bit can be either on or off, usually represented by 1 or 0. Thus, any byte's value can be represented by eight numbers, where the numbers are limited to 1 and 0. Example: **10110111**

Since only two possibilities exist for each bit (1 or 0), this numbering system is referred to as binary or base two.

Our usual number system is known as decimal or base ten. This system uses ten different symbols to designate numbers (0-9). The position of the number from the rightmost side determines the *power of 10* by which the number is to be multiplied. For instance, here's how the decimal number 2156 breaks down:

Decimal:	2	1	5	6
Position:	3	2	1	0

Power of 10: $2*10^3 + 1*10^2 + 5*10^1 + 6*10^0 = 2156$
Value: (2000) + (100) + (50) + (6) = 2156

The value of binary numbers is determined the same way, except the position of the digit (the bit) determines the *power of 2* by which the number is multiplied. Here's how the binary number 10110111 breaks down when translated to decimal:

Binary:	1	0	1	1	0	1	1	1
Position:	7	6	5	4	3	2	1	0

Power of 2: $1*2^7 + 0*2^6 + 1*2^5 + 1*2^4 + 0*2^3 + 1*2^2 + 1*2^1 + 1*2^0 = 183$
Value: (128) + (0) + (32) + (16) + (0) + (4) + (2) + (1) = 183

Now we can attack our original problem – changing bits within registers on the 64. Bits are labeled 0 to 7 *from right to left*. To set bit three of a register to 1 (in other words, turn it on), you can POKE the register with an 8 (2 to the third power). This POKE erases what was in that register and replaces it with an 8 (00001000).

You can change certain bits within a register without affecting the other bits with the keywords OR and AND. The OR operator is used to compare two bytes, bit for bit, to alter certain bits without altering the others. Here's an example:

Original byte — 00010001 = 17
OR with 8 — 00001000 = 8
Resulting byte — 00011001 = 25

The OR statement looks at each bit of the original byte and compares it to the corresponding bit of the OR byte. If one OR the other of the bits is a 1, then the resulting byte will contain a 1 in that particular bit. If neither of the bits is a 1, the resulting byte will contain a 0 in that bit. Thus, here's how you would set bit 3 to a 1 without changing the other bits, in BASIC:

POKE byte,(PEEK(byte)OR 8)

The AND keyword also looks at each bit of one byte and compares it to the corresponding bit of the second byte, but in a different way. Here's an example of AND:

Original byte — 00001111 = 15
AND with 254 — 11111110 = 254
Resulting byte — 00001110 = 14

The AND operation looks at the two bytes and sets a 1 in the corresponding bit of the resulting byte *only* if both the first bit AND the second bit are 1's. Thus, ANDing 15 with 254 clears the 0 bit. In BASIC this is written: **POKE(byte),(PEEK(byte)AND254)**

the tone you hear depends on the pitch of Voice 3, not Voice 1 as would normally be the case.

Paddling With The SID

The SID chip also contains two registers (25-26) connected to the two joystick ports. These registers will contain a number from 0 to 255, depending on the resistance of a potentiometer attached to the ports (255 at maximum resistance). Since game paddles are really potentiometers (variable resistors), these ports can be used to register paddle movement and can easily be used to change values in other registers within the chip while sounds are being produced.

This simple routine can be added to a sound program to control the pitch of Voice 1 with a paddle plugged into port one while a tone is being played:

```
10 POKE 54272+1,PEEK(54272+25):GOTO10
```

This line connects the paddle value to the high-byte frequency value of Voice 1. It's much easier to study the effects of changing sound values if you can hear the sound playing as you experiment. That is the basis of "Soundmaker 64."

Soundmaker 64

Soundmaker 64 allows you to create your own sounds and manipulate them by changing various parameters. Attack, decay, sustain, and release are included as well as pitch, filters, ring modulation, and synchronization. The pulse waveform may be manipulated to change the pulse width of the sound — altering the timbre of the resulting sound considerably.

To use Soundmaker 64, type in the program and save it on disk or tape. When you are sure you have a saved copy, run the program. After a brief delay while the program loads a small machine language routine into memory, the word "Attack" appears at the upper-right corner of your screen. Using the + and - keys, you can increase or decrease the attack value for your sound. The current value POKEd is represented by both a bar graph and a number. The number varies in units of sixteen or one depending on which parameter you are working with. These values are meant to serve as a reference point only, since they may differ from the actual value by one unit. The increments were selected to make the changes in parameters very easy to hear and the program easy to use.

Once you have decided on the attack value, simply hit RETURN and the next parameter appears. Keep in mind that "Sustain" and "Volume" must be a reasonably high number for the sound to be audible. When you have picked all the parameters ("Pulse wave low" is the last one on the screen), then you can play the sound with the function keys. F1 plays the sound with the sawtooth waveform, F3 with the triangle waveform, F5 with the noise waveform, and F7 with the pulse waveform.

Ring Modulation And Sync

The up-arrow key (beside the asterisk) plays your sound as it is ring modulated with Voice 3, and the left-arrow key (beside the 1) plays the synchronized sound resulting from the pitches of Voice 1 and Voice 3. (Ring modulation and synchronization are limited to Voice 1.)

Once you have heard Voice 1, simply hit the 2 key and you will again be prompted for the parameters. As with Voice 1, you play Voice 2 with the function keys. To hear Voices 1 and 2 simultaneously, hit the space bar. To select the parameters for Voice 3, press the 3 key. The space bar then plays all voices previously defined. If you have selected ring modulation or synchronization for Voice 1, you may not be able to use Voice 3 as a separate sound.

Changing Sounds

To alter any parameter at any time after entering it originally, merely press the key which is in reverse field on the parameter name and press the + or - key to raise or lower the value. When done, hit RETURN.

You can even change parameters as the sound is playing. To do this, hit one of the function keys or one of the arrow keys to start the note and, without releasing it, hit the reverse field character of the parameter you wish to change. Then change the sound with the + and - keys.

To use the filters as the sound is being played, you must first start the sound that you want, then, without releasing the key, hit either H (for high pass), B (for band pass), or L (for low pass). Next, hit F for filter, and use the + and - keys to increment or decrement the cutoff frequency. As before, hit RETURN to end the note.

To save the sound or sounds that you have created, press Q while the note is playing. The screen clears and a program appears on the screen. Type NEW and press RETURN over the lines as they are listed on the screen. Then you can play this sound, or save it on tape or disk and use it later as a routine in your own programs. To use it as a routine, you'll need a delay loop such as this to set the duration:

```
70 FOR T=1TO 2000:NEXT T
```

Then, to turn off the sound, use this line:

```
80 FORT=49152+4TO49152+18STEP7:POKET,
(PEEK(T)AND254):NEXT:SYS53017
```

To turn on the sound in your own program, you can either GOSUB the whole routine, or use this line (with your own line number):

```
FORT=49152+4TO49152+18STEP7:POKET,
(PEEK(T)OR1):NEXT:SYS53017
```

A Bit About The Program

Soundmaker 64 uses a tiny machine language (ML) routine which copies the contents of 24 bytes starting at 49152 to the sound registers beginning at 54272. The ML routine copies the registers in the order they should be POKEd to properly create a sound.

This is done because sound registers are *write only* registers. That is, when values are POKEd into the SID registers, they cannot be PEEKed later. Instead, you must store the values in variables or other memory locations. The ML routine stores these values in a safe area of memory and allows us to copy them at any time to the SID registers. The ability to "remember" the values which have been POKEd into the SID chip makes Soundmaker 64 possible.

See program listing on page 198.

DAN CARMICHAEL, ASSISTANT EDITOR

Animating With Custom Characters

In this month's column we'll show you how to spice up your programs with animation. The basic information also applies to the Commodore 64.

Elsewhere in this issue we've shown how to create custom characters for your programs. Now we'll learn how to animate those custom characters and add a little action to the screen.

Almost everyone remembers their first experience with animation. You might have drawn a stickman on successive pages of a scratch pad, and when the pages were rapidly flipped, the stickman appeared to run across the pages. This is the same principle used in computer animation. By creating various custom characters (like a stickman) in different positions, and then POKEing them to the screen, you can simulate animation.

Before proceeding, turn off your VIC, remove any memory expansion, then turn it on again. Enter, SAVE to tape or disk, and then RUN Program 1. This program creates a few custom characters (stored in the DATA statements). Line 10 reserves a special place in memory for them (512 bytes at the top of BASIC memory), then POKEs the characters into memory. The POKE (36869) in line 50 directs the operating system to look at our special character set in RAM memory instead of the standard character ROM.

Program 1

```
10 POKE52,28:POKE56,28:CLR               :rem 18
20 FORA=7313TO7679:POKEA,0:NEXT         :rem 148
30 FORA=7168TO7312:READB:POKEA,B:NEXT
                                         :rem 55
50 POKE36869,255:PRINT"{CLR}{5 DOWN} @ A
   {SPACE}B C D E F G H I{DOWN}{3 SPACES}
   J K L M N O P"                        :rem 56
60 GETA$:IFA$=""THEN60                  :rem 239
7168 DATA 56,84,146,146,146,146,84,56
                                         :rem 62
7176 DATA 56,68,138,146,146,162,68,56
                                         :rem 64
7184 DATA 56,68,130,130,254,130,68,56
                                         :rem 43
7192 DATA 56,68,162,146,146,138,68,56
                                         :rem 62
7200 DATA 12,44,90,28,24,164,66,1       :rem 83
7208 DATA 12,12,24,60,24,88,40,8        :rem 35
7216 DATA 12,12,8,28,12,14,16,32        :rem 26
7224 DATA 12,12,8,30,44,10,20,32        :rem 14
7232 DATA 0,0,0,0,0,0,0,0              :rem 156
7240 DATA 0,0,0,0,0,0,0,0              :rem 155
7248 DATA 24,24,60,60,255,255,126,66
                                        :rem 248
7256 DATA 8,88,26,188,111,253,126,74
                                          :rem 4
7264 DATA 138,17,144,56,109,229,62,72
                                         :rem 49
7272 DATA 130,17,128,40,76,165,26,64
                                        :rem 246
7280 DATA 0,16,128,40,4,164,24,0        :rem 26
7288 DATA 0,0,0,40,0,36,0,0             :rem 20
7296 DATA 0,0,0,8,0,0,0,0              :rem 174
7304 DATA 159,255,77,111,192,254,16,59
                                        :rem 100
7312 DATA 0                              :rem 23
```

This program displays the custom characters with a PRINT statement. To create the animation effects, we could use the PRINT statement, but it would be easier and more efficient to use POKE.

The next few programs use the custom characters created in Program 1, so don't turn off the computer. Press RUN/STOP—RESTORE, then enter NEW. This will clear BASIC memory of the program that was there, but will leave our custom character set intact, still protected.

COMPUTE!'s First Book of VIC Games

Authors: COMPUTE! Magazine Contributors
Price: $12.95
On Sale: Now

The VIC-20 is a versatile computer. Its programmable color, graphics, and sound can add a lot to game programs.

Over the past few years, COMPUTE! Magazine has published a wide variety of games for the VIC. But some back issues are hard to find or unavailable.

That's why the editors of COMPUTE! have chosen the best games, revised them, and added previously unpublished games, putting them together into *COMPUTE!'s First Book of VIC Games*. Each game has been tested and debugged and is ready to type into a VIC-20.

First Book of VIC Games contains fast-action arcade games that require quick reflexes, as well as strategy games that test logical skills. For $12.95 you get two dozen games, including:

Time Bomb — You hear the time bomb ticking, and you have to find your way through the maze to defuse it. But the maze is larger than the screen, and you can't always see where the dead ends are.

Closeout — The local department store is having a sale. You try to snatch as many sale items as possible, while avoiding the horde of bargain hunters who are trying to thwart you.

Cryptic Numbers — A good logic game. The computer picks a pattern of numbers that you have to guess. After each turn you learn how close you were.

Air Defense — Missiles are falling out of the sky onto your city. Aim carefully; you get only one shot at each missile.

Thunderbird — Your goal is to break out of the playing field by using the thunderbird that controls the satellite.

Sky Diver — Put on your parachute, jump out of the plane, and try to land on one of the targets. Watch out for wind currents that may blow you off course.

Deflector — A ball is bouncing around the screen. You can aim it toward targets by strategically placing deflectors that change its direction.

Hidden Maze — Lost in a maze, you're trying to get out. But it is dark, and you can see only a few spaces in front of you.

Outpost — Your small fortress is under siege. You have two types of lasers and some torpedoes. Unfortunately, the energy supply is dwindling and the computer is on the blink. The supply ship may (or may not) show up in time to make repairs.

Table of Contents

First Book of VIC Games is more than just a book of program listings. Each program is annotated and explained; you can modify the games if you like or use the many programming techniques in your own games. Also included is a useful program you can use to draw mazes for games you write. Three chapters show you how to develop a game program. Another tells you how to take advantage of the VIC's sound, graphics, and color capabilities. The index lists references where you can learn more about programming. And *First Book of VIC Games* is spiral bound to lie flat while you are typing in programs.

To order directly, call
TOLL FREE
800-334-0868
(919-275-9809 in NC)

ORDER NOW

Available at computer dealers and bookstores nationwide. Or send a check or money order to COMPUTE! Books, P. O. Box 5406, Greensboro, NC 27403.

Add $2 shipping and handling. Outside the U.S. add $5 for air mail, $2 for surface mail. All orders prepaid, U. S. funds only.

Rolling And Spinning Wheels

If you turn the computer off then back on, you will have to rerun Program 1. Now enter, then RUN, Program 2.

Program 2

```
10 PRINT"{CLR}":C=PEEK(646):POKE38630,C:P
   OKE38632,C:POKE36869,255:SP=50
20 FORA=0TO3:POKE7910,A:POKE7912,3-A:FORT
   =1TOSP:NEXTT:NEXTA:GOTO20
```

As you can see, all this program does is POKE custom characters to the center of the screen. The effect of the spinning wheels is created by POKE-ing the screen with various wheels (characters) with the spoke in different positions. You can vary the speed of the spin by changing the variable SP in line 10. A lower value will increase the speed of the spin, a higher value will slow it.

Study line 20 of Program 2. It contains a programming technique that might be of interest to you. While the "FOR A . . . " loop is set to count forward (to animate the left wheel running clockwise), the second POKE (7912) in line 20 in effect counts backwards. This is how the right wheel is made to run counterclockwise. By subtracting the variable A from the constant 3, the FOR A loop seems to run backwards. It's like having two FOR/NEXT loops in one — one counting forwards, the other counting backwards.

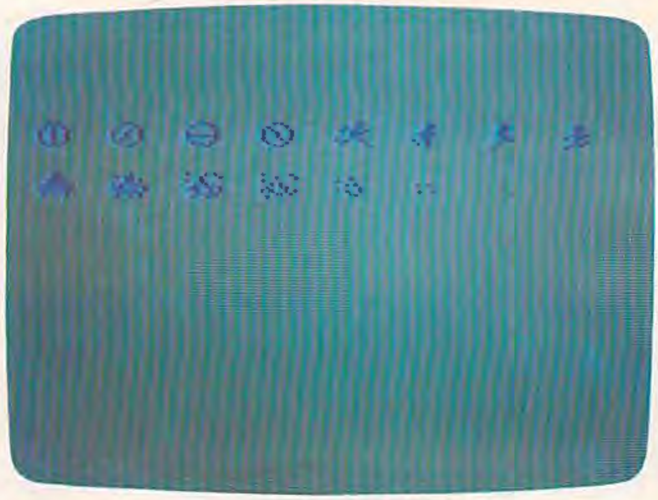
Again, press RUN/STOP—RESTORE, then enter NEW. Now enter Program 3 and RUN. This program uses the same wheel characters, but instead of spinning in one place, the various wheels are POKEd to successive positions, producing the effect of the wheel rolling across the screen. Again, the variable SP in line 10 controls the speed. If you change the speed and make the program run too fast, the rolling effect is almost lost.

Program 3

```
10 PRINT"{CLR}":POKE36869,255:C=PEEK(646)
   :FORA=38400TO38905:POKEA,C:NEXT:N=0:SP
   =90
20 FORA=7680TO8185:POKEA,N:POKEA-1,32:N=N
   +1:FORT=1TOSP:NEXTT:IFN=4THENN=0
30 NEXTA
```

Notice **POKE A-1,32** in line 20. This POKE puts a blank space (screen POKE character 32) on the screen just before the current position of the rolling wheel. This causes the wheel to erase the space just behind itself as it rolls across the screen. Remove this POKE from the program, RUN it, and see what happens.

Speed is important when animating custom characters. If they move too fast, the eye cannot see all the subtle changes in the characters, and the effect is lost. On the other hand, if the program runs too slow, the characters appear to jump, and the effect of smooth animation is lost.



A close-up of the custom characters used in the example programs.

The Running Stickman

Press RUN/STOP—RESTORE, then enter NEW. Now enter Program 4 and RUN.

Program 4

```
10 PRINT"{CLR}":POKE36879,27:FORA=38400TO
   38905:POKEA,0:NEXT:POKE36869,255:N=4:S
   P=100
20 FORA=4TO7:POKE7910,A:FORT=1TOSP:NEXTT:
   NEXTA
30 GETA$:IFA$=""THEN20
35 SP=150
40 POKE7910,32:FORA=7680TO8185:POKEA,N:PO
   KEA-1,32:FORT=1TOSP:NEXTT:N=N+1:IFN=8T
   HENN=4
50 NEXTA
```

As you can see, Program 4 also POKEs custom characters (various poses of a stickman) to the center of the screen. The animated running effect here is produced with four different custom character stickmen. If you watch the animated action closely, it even looks as if the stickman is alternating legs while running.

Now press any key on the keyboard and watch the stickman run across the screen. Again, speed is an important factor here. Change the variable SP in line 10 to 0, then RUN. As you can see, the speed is so fast that not only is the animation effect lost, but it's hard to even recognize the character as it moves.

Program 4 shows the main drawback of animating with POKEd custom characters: jumpy action. This effect is caused by the characters moving one character block, or eight pixels, at one time. For truly smooth animation, high-resolution graphics is better because it allows you to move objects one pixel at a time. However, we will not touch upon hi-res animation here because it is an advanced programming technique that requires machine language.

Exploding UFOs And Other Effects

POKEd animation isn't limited to making objects move. You can also create interesting effects with stationary objects. Again, press RUN/STOP—RESTORE, then enter NEW. Now enter and RUN Program 5. This program uses custom characters to make a flying saucer explode and disintegrate. Change the variable SP in line 10 to 250, then RUN it again. Notice how tiny bits (pixels) of the saucer seem to fly outward, then disappear.

Program 5

```
1Ø PRINT"{CLR}":POKE36879,27:POKE36869,25
   5:POKE3863Ø,Ø:SP=12Ø
15 POKE791Ø,1Ø:FORT=1TO5ØØ:NEXTT
2Ø FORA=11TO16:POKE791Ø,A:FORT=1TOSP:NEXT
   T:NEXTA:FORT=1TO5ØØ:NEXTT:GOTO15
```

Once you've conquered the art of animating with custom characters, you can add the final touch: sound. Adding the sounds of footsteps as the stickman runs across the screen, or an exploding sound as the flying saucer disintegrates, will provide that little extra touch that makes your graphics animations fun and enjoyable.

Merging Programs On The 64

John A. Winnie

For intermediate programmers, "Merger" allows you to build up large programs by working on smaller portions separately and then linking them together later. This approach is used by many professionals.

If you do much BASIC programming, sooner or later you'll need to merge two short programs to form a larger one. Or perhaps you'll need to append onto a program a series of DATA statements – DATA for sprites, redefined characters, sound and music, or whatever. Here is a quick and easy way to add those DATA statements – or any other BASIC statements, for that matter – onto the end of your programs.

Of course, various techniques for merging programs have been around for some time. When all that is needed is a simple append, however, the method presented here does the job nicely. The program below, "Merger," is designed to merge with any programs which are appended to it, and it allows you to keep on appending indefinitely.

Using Merger

After typing and saving Merger, load it in the usual way. Next, RUN Merger, and then load in your main program. Now, as Merger instructs, POKE locations 43 and 44 with 1 and 8, respectively. Your main program is now appended to Merger and ready for any DATA statements you may want to add later.

Remember, Merger allows you to append programs only, not to insert them. So to prepare for using Merger later, begin your programs with a line number greater than five. For the same reason, all DATA statements to be added should begin with a line number higher than those already present in the program. When you have finished, just erase Merger by deleting lines 1 through 5.

How Merger Works

First, clear out your Commodore 64 by typing NEW and pressing RETURN. Then enter the following simple program:

```
10 REM
```

Press RETURN, and the one-line program is now entered into memory beginning at address 2048 and running on upward. To see just how the program is stored, enter:

```
FOR I=2048 TO 2056:PRINT PEEK (I) :NEXT I
```

If all this has been done correctly, you now should see a list of memory contents which looks like this:

```
0,7,8,10,0,143,0,0,0
```

The 0 in address 2048 is invariable: all BASIC programs begin with zero. They also always end with a zero; in fact, they always end with exactly three zeros – which is just what we see here in memory locations 2054 through 2056. From this point on in memory, BASIC will store any variables and other information that it may need to execute the program.

In general, when a BASIC line is stored, it will end with a single zero, not three zeros. When a new line is appended to the program, its code begins immediately after that single zero. So in the example above, if the line

```
20 REM
```

were now added to our sample program, the (link

Commodore 64 and VIC-20

Graphics

DRAW with your joystick!

"ELF" from Picture Library

Now, you can create **high-resolution pictures** on-screen with your joystick as a "pen." Design critters, objects, pie-charts — whatever your imagination wishes! **SAVE your creations to tape or disk, and PRINT them on a VIC printer.** Educational and fun!

Draw narrow or wide lines, curvy or straight; set points; add captions; create background patterns; change picture, background, and border colors; reverse colors for a negative; even connect dots with straight lines automatically! **You control every dot on the screen.**

A large "Picture Library" is included to get you started, plus a 20-page instruction manual. Joystick required. VIC printer and disk drive optional. **Now on disk or cassette!**

"FLY" from Picture Library

For the Commodore 64:
'64 Panorama $29.95

For the VIC-20:
VIC-PICS $29.95

(Full features need 8K mem exp; reduced version included for unexpanded VIC.)

Available for other printers—Call!

PRINT (the unprintable)... with Smart ASCII Plus-$59.95

Now, print the unprintable **Commodore graphics** on your **dot-addressable parallel printer*** with Smart ASCII Plus. This **powerful, low-cost software interface** converts your **user port** into a fast, intelligent port for "Centronics" protocol printers, and we even supply the cable!

Six flexible print modes: GRAPHICS, TRANSLATE, DaisyTRANSLATE, CBM ASCII, True ASCII, PIPELINE. GRAPHICS mode creates actual VIC/64 keyboard graphics. TRANSLATE converts normally unprintable control-codes into text: (CLR), (RVS), (BLU), etc., with an extended mode for Daisywheel printers. Convenient set-up menu and simplified operation. Compatible with most application programs: WordPro 3+, EasyScript, Quick Brown Fox (for the VIC), Writer's Assistant, etc.

Complete with connecting cable for printer and instruction manual. On cassette. Copy to your disk for quick loading. (Upgrades available for original Smart ASCII owners.)

*Requires dot-addressable printer such as: Epson FX-80 or MX-80/100 with Graftrax; Okidata Microline 84; C. Itoh Prowriter 1 & 2; Star Micronics Gemini-10 or 15. **Other printers—Call!**

VIC 20 and Commodore 64 are trademarks of Commodore Electronics, Ltd.

(816) 333-7200

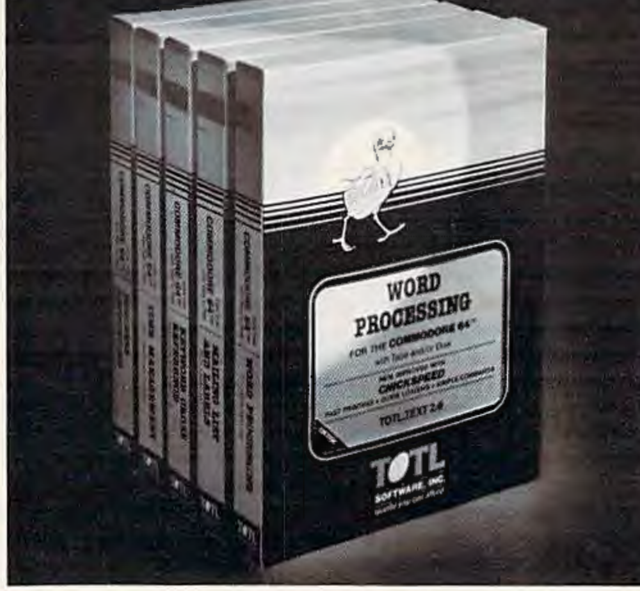
Send for a free brochure.

MAIL ORDER: Add $1.50 shipping and handling ($3.50 for C.O.D.); VISA/Mastercard accepted (card# and exp. date). MO residents add 5.625% sales tax. Foreign orders payable U.S.$, U.S. Bank ONLY; add $5 shp/hndlg.

MIDWEST MICRO inc.

311 WEST 72nd ST. • KANSAS CITY • MO • 64114

There's a New Chick in Town



...with a wing load of Quality Software for the Commodore 64™ and VIC 20™

Once you've compared our programs, their features and prices, you'll agree there's no competition in sight. You'll also discover another important reason to go TOTL...Customer Assistance After You Buy...Something nearly unknown in the low cost software field.

some features

Menu driven, Easy to use ■ Available on tape or disk ■ Compatible With Most Column Expansion Hardware ■ Built-in ASCII Translation for Non Commodore Hardware ■ Color Variables for Easy Modification ■ Machine language speed for Word Processing and Mailing List & Label software.

and there are 5 unique programs to choose from

1. **WORD PROCESSING** has the speed and versatility to produce documents, forms and letters in a straightforward approach that is easily and quickly learned.
2. **MAILING LIST AND LABEL** lets you organize your mailing lists, collection catalogs, menus, recipes and anything that demands listing or sorting.
3. **TOTL TIME MANAGER** helps you plan schedules and analyze events and activities by persons, project catagory and date. Ideal for project planning in the home or business.
4. **RESEARCH ASSISTANT** turns your computer into an advanced, automated indexing and cross reference system. A must for the student, educator or the research professional.
5. **SMALL BUSINESS ACCOUNTING** is a set of straightforward accounting programs. Frees the salesman, entrepreneur or service professional from time consuming record keeping tasks.

TOTL SOFTWARE, INC. quality you can afford

Ask your dealer about TOTL Software or send in the coupon for further details and ordering information.

1555 Third Avenue, Walnut Creek, CA 94596

PLEASE SEND ME MORE INFORMATION ON TOTL SOFTWARE

Name: ______________________

Address: ______________________

______________________ Zip ________

of the) new line would now come in at address 2055 – the address of the middle zero in the triplet; a new triplet of zeros would appear later in memory, signaling the end of line 20 and the new end of the program. (Try this later to see for yourself.) So, to merge programs, we simply have to make sure that we load in the new section at the address of the middle zero (2055, in our example) within the three zeros which signal the end of our original program. What we need to do is raise the floor of BASIC to this new address, load the section to be merged, and then lower the floor to its original value (here, 2049).

Tinkering With BASIC

Raising the floor of BASIC is easy. The new address is simply POKEd into addresses 43 and 44 in low-byte, high-byte order. (HI = INT (ADDRESS#/256) :LO = ADDRESS#-256*HI.) Finding this new address is another matter, but fortunately, this turns out to be easy as well.

As I mentioned above, BASIC needs to know where it is safe to begin to store its variables. In other words, BASIC needs to know the first address to come after the three zeros which end the program. Hence, the computer stores this address in a pair of memory locations in the usual low-byte, high-byte form. In the 64, these locations are addresses 45 and 46.

To see this, enter PRINT PEEK(45),PEEK(46), and out should come the pair 9,8. Since the address 2057 is the first address to follow our sample program, and 2057 = 256*8 + 9, we have the expected result.

Now that we have the address of the first location after the end of the program, the rest is easy. The new program is simply loaded into memory two places before this location. In our example, we load at location 2055 (2057-2). And that's all there is to it.

The basic idea behind Merger should now be clear. Everything of interest is packed into line 4. First, for any program which begins with these lines, the new floor for BASIC is computed using the contents of locations 45 and 46, as described above. Next, the floor of BASIC is raised to the new location. As a result, any new program now loaded will start right at the tail end of the previous program – just where we want it.

```
1 PRINT "{CLR}{10 DOWN}";TAB(15)"MERGE?"
  :GETA$:IFA$=""THENFORI=0TO500:NEXT:GOT
  O1
2 IFA$<>"Y"THEN5
3 PRINT"{CLR}{10 DOWN}";TAB(5)"LOAD YOUR
   ADDITION.{2 SPACES}THEN POKE 43,1 AND
   44,8."
4 E=256*PEEK(46)+PEEK(45)-2:H=INT(E/256)
  :L=E-256*H:POKE43,L:POKE44,H:END
5 REM
```

The Automatic Proofreader

• IMPORTANT: We added a POKE to the "Automatic Proofreader" (October) to protect it from being erased when you LOAD another program from tape. The POKE does protect the Proofreader, and the Proofreader itself is not affected. However, a quirk in the VIC-20's operating system means that programs typed in with the Proofreader and SAVEd to tape cannot be LOADed properly later. If you LOAD a program SAVEd while the Proofreader was in memory, you see ?LOAD ERROR. This applies only to VIC tape SAVEs (disk SAVEs work OK, and the quirk was fixed in the 64). The solution is this special LOAD procedure:

1. Turn the power off, then on.
2. LOAD the program from tape (disregard the ?LOAD ERROR).
3. Enter: POKE 45,PEEK(174):POKE 46,PEEK (175):CLR
4. ReSAVE the program to tape.

The program will LOAD just fine in the future. This month, the Proofreader has been updated to prevent this problem. It also has been improved in other ways. Please observe these new procedures:

1. After first entering the Proofreader, SAVE it before typing RUN. The new Proofreader erases its BASIC loader from memory when first activated.

2. The new Proofreader checks itself for typing errors in the DATA statements when first RUN.

3. The new version now works on both the VIC and 64. Since the POKE to protect the Proofreader has been removed for the VIC's sake, when using tape you must reLOAD the Proofreader and RUN it whenever you type in a new program.

4. SAVE to tape wipes out the Proofreader, so press RUN/STOP-RESTORE before SAVE.

We strongly recommend that you type in the new version of the Proofreader and discard the old one. We apologize for any inconvenience this may have caused you.

"The Automatic Proofreader" will help you type in program listings from COMPUTE!'s Gazette without typing mistakes. It is a short error-checking program that hides itself in memory. When activated, it lets you know immediately after typing a line from a program listing if you have made a mistake. Please read these instructions carefully before typing any programs in COMPUTE!'s Gazette.

Preparing The Proofreader

1. Using the listing below, type in the Proofreader. Watch out for typing an l instead of a 1, or an O instead of a 0, extra commas, etc.
2. SAVE it on tape or disk at least twice *before running it for the first time*. If you mistype the Proofreader, it may cause a system crash when you first run it. By SAVEing a copy beforehand, you can reLOAD it and hunt for your error. Also, you'll want a backup copy of the Proofreader because you'll use it again and again – every time you enter a program from COMPUTE!'s Gazette.
3. RUN the Proofreader. It will be POKEd into a relatively safe area of memory, the cassette buffer.
4. Type RUN to activate the Proofreader. If you ever need to reactivate it, just enter the command SYS 886 and press RETURN.

Using The Proofreader

All VIC and 64 listings in COMPUTE!'s Gazette now have a *checksum number* appended to the end of each line, for example ":rem 123". *Don't enter this statement when typing in a program.* It is just for your information. The rem makes the number harmless if someone does type it in. It will, however, use up memory if you enter it, and it will confuse the Proofreader, even if you entered the rest of the line correctly.

When you type a line from a program listing and press RETURN, the Proofreader displays a number at the top of your screen. *This checksum number must match the checksum number in the printed listing.* If it doesn't, it means you typed the line differently than the way it is listed. Immediately recheck your typing. Remember, don't type the rem statement with the checksum number; it is published only so you can check it against the number which appears on your screen.

The Proofreader is not picky with spaces. It will not notice extra spaces or missing ones. This is for your convenience, since spacing is generally not important. But occasionally proper spacing *is* important, so be extra careful with spaces, since the Proofreader will catch practically everything else that can go wrong.

There's another thing to watch out for: if you enter the line by using abbreviations for commands, the checksum will not match up. But there is a way to make the Proofreader check it. After entering the line, LIST it. This eliminates the abbreviations. Then move the cursor up to the line and press RETURN. It should now match the checksum. You can check whole groups of lines this way.

When you're done with the Proofreader, disable it by pressing RUN/STOP-RESTORE (hold down the RUN/STOP key and press RESTORE). If you need it again, enter SYS 886. It will then be ready once again to act as your personal typing aid. However, sometimes the Proofreader can be wiped out of memory. In this case, you'll have to reLOAD the Proofreader from tape or disk.

Automatic Proofreader For VIC And 64

```
100 PRINT"{CLR}PLEASE WAIT...":FORI=886TO
    1018:READA:CK=CK+A:POKEI,A:NEXT
110 IF CK<>17539 THEN PRINT"{DOWN}YOU MAD
    E AN ERROR":PRINT"IN DATA STATEMENTS.
    ":END
120 SYS886:PRINT"{CLR}{2 DOWN}PROOFREADER
     ACTIVATED.":NEW
886 DATA 173,036,003,201,150,208
892 DATA 001,096,141,151,003,173
898 DATA 037,003,141,152,003,169
904 DATA 150,141,036,003,169,003
910 DATA 141,037,003,169,000,133
916 DATA 254,096,032,087,241,133
922 DATA 251,134,252,132,253,008
928 DATA 201,013,240,017,201,032
934 DATA 240,005,024,101,254,133
940 DATA 254,165,251,166,252,164
946 DATA 253,040,096,169,013,032
952 DATA 210,255,165,214,141,251
958 DATA 003,206,251,003,169,000
964 DATA 133,216,169,019,032,210
970 DATA 255,169,018,032,210,255
976 DATA 169,058,032,210,255,166
982 DATA 254,169,000,133,254,172
988 DATA 151,003,192,087,208,006
994 DATA 032,205,189,076,235,003
1000 DATA 032,205,221,169,032,032
1006 DATA 210,255,032,210,255,173
1012 DATA 251,003,133,214,076,173
1018 DATA 003
```

THE BEGINNER'S CORNER

C. REGENA

DATA, READ And RESTORE Statements

By now you have typed in programs from COMPUTE!'s Gazette and probably have them running for your own use. If you have had any trouble, the most likely place for errors (other than simple typing errors that are easily recognized) is in DATA statements. This month we'll discuss some data examples that will help you understand how the data is used and why typing the DATA statements exactly is so important. If you do get some errors involving data, you can follow the computer's RESTORE and READ process to pinpoint trouble spots.

DATA statements are used in combination with READ statements. Generally, fewer program statements can be used in a DATA-READ procedure than using "assignment" (LET or A=4 type) statements. A DATA statement starts with the line number, then the word DATA (which may be abbreviated D shift-A), then numbers or words separated by commas. A DATA statement may *not* be combined with any other statements for the line number. A DATA statement may be placed anywhere in the program. If the computer comes to a DATA statement, the statement is ignored until a READ statement is encountered.

When the computer comes to the first READ statement, it then looks for the first DATA statement and assigns the value in the DATA statement to the variable in the READ statement. Numbers or strings may be used, and all items are separated by commas. You may READ one or any number of items. When the computer meets another READ statement, the computer will read the very next data item which hasn't been used, whether it is in the same DATA statement previously used or not. The computer keeps track of where it is in a series of data items. Let's look at an example.

```
10 READ A,B,C,D,E,F
20 DATA 3,4,7,2,7,8
30 PRINT A+B,C-D,E*F
```

These three lines are equivalent to:

```
10 A=3
20 B=4
30 C=7
40 D=2
50 E=7
60 F=8
70 PRINT A+B,C-D,E*F
```

When the computer executes line 10, the instructions are to read and assign values to the variables whose names are A, B, C, D, E, and F. The computer finds the values in the DATA statement, line 20. The computer assigns the first data item to A, the second data item to B, and so forth according to the READ statement. You can try typing the DATA statement as line 5 or line 35 instead of line 20, and you'll see it doesn't matter where the DATA statement is located. If you have several READ statements and several DATA statements, however, then your DATA statements must be in the proper sequence matching the READ statements, but it doesn't matter where they are placed in the program.

It also doesn't matter if you type a few extra numbers in the DATA statement; the extra items will just be ignored until another READ statement needs them. However, you must have enough items to satisfy the READ statement. If you have too few items, you will get an OUT OF DATA error.

Matching Variables To Data

Another stipulation is that you need to match strings with string variable names. If you use a string variable in the READ statement, the item in the DATA statement will become the value for the string variable (and it may be a number or alphabetic characters or symbols). If you want to READ a numeric variable, you must have a cor-

COMPUTER MAIL ORDER

VIC-20
$99

commodore COMPUTER
HARDWARE

COMMODORE 64
$219

1541
SINGLE DISK DRIVE
$249

1525 80 Column Printer	$219.00
1600 VIC Modem	$59.00
1610 VIC Term 40	$49.00
1650 AD/AA Modem	$89.00
1701 14" Color Monitor	$249.00
1311 Joysticks (each)	$5.99
1312 Paddles	$11.99
1110 VIC 8K	$42.00
1111 VIC 16K	$69.00
1011 RS-232 Interface	$42.00
1211 Super Expander	$53.00

1530
DATASETTE
$69.00

1520
COLOR
PRINTER/PLOTTER
$169

PRINTERS

Epson (MX80 FT, MX100, RX80, FX80, FX100)	CALL
Okidata (82, 83, 84, 92, 93)	CALL
Star Gemini 10X	$299.00
Smith Corona TP-1	$469.00
C.Itoh Prowriter 8510P	$379.00
C.Itoh Gorilla	$209.00

SOFTWARE

commodore

1906 Super Alien	$14.00
1910 Radar Rat Race	$14.00
1917 VooDoo Castle	$19.00
1922 Cosmic Cruncher	$14.00
1923 Gorf	$17.00
1924 Omega Race	$17.00
110 VIC Reference Guide	$15.00
CBM 64 Reference Guide	$18.00

EASY BUSINESS SERIES
C-64 Disks

EasyCalc 64	$59.00
EasyFinance 64	$19.00
EasyMail 64	$19.00
EasyScript 64	$39.00
Word/Name Machine	$16.00

PROGRAMMER SERIES
C-64 Disks

Assembler	$16.00
Logo	$39.00
Pilot	$39.00
Pet Emulator	$16.00
Screen Editor	$16.00
Video/Music Support	$15.00

ART & MUSIC SERIES
C-64 Disks

Music Machine	$16.00
Music Composer	$16.00
Meta Music I	$39.00

FLOPPY DISKS

Maxell	$32.00
Verbatum	$26.00
Elephant	$18.95

CARDCO.

Light Pen	$32.00
Cassette Interface	$29.00
Parallel Printer Interface	$69.00
3-Slot VIC Expansion Interface	$32.00
6-Slot VIC Expansion Interface	$79.00

ARTWORX
C-64 Cassettes

Bridge 3.0	$14.95

C-64/VIC 20 Cassettes

Teacher's Pet	$11.95

BRODERBUND
VIC 20 Cassettes

Martian Raid	$16.00
Shark Trap	$16.00
Multisound Synthesizer	$16.00

COMMERCIAL DATA
VIC 20 Cassettes

Motor Mouse	$23.00
Centipode	$23.00
Road Toad	$23.00

C-64 Cassettes

Road Toad	$23.00

CREATIVE SOFTWARE
VIC 20 Cassettes

Home Finance	$16.00
Home Inventory	$16.00

VIC 20 Cartridges

Astro Blitz	$29.00
Black Hole	$29.00
Trashman	$29.00
Choplifter	$29.00

EPYX
VIC 20 Cassettes

Sword of Fargoal	$24.00
Rescue at Rigel	$24.00

QUICK BROWN FOX
VIC 20 Cartridges

Word Processor	$49.00

UMI
VIC 20 Cartridges

Amok	$30.00
Meteor Run	$40.00
Alien Blitz	$30.00

VIC 20 Cassettes

Viterm A	$16.00
The Alien	$16.00

HES
VIC 20 Cassettes

Torg	$14.00
Skier	$14.95
Tank Wars	$14.95
Pak Bomber	$12.95
Dam Bomber	$12.95
Maze of Mikor	$14.95
Laser Blitz	$14.95
Pinball	$12.95
Tank Trap	$14.95

VIC 20 Cartridges

VIC Forth	$47.95
HES Mon	$29.00
HES Writer	$29.00
Aggressor	$31.95
Synthesound	$47.95
Shamus	$29.00
Protector	$32.00
Turtle Graphics	$31.95

C-64 Cartridges

HES Mon	$29.00
HES Writer	$35.00

NUFEKOP
VIC 20 Cassettes

Alien Panic	$10.00
Race Fun-Drag Race	$16.00
The Catup	$10.00
Exterminator	$19.00

C-64 Cassettes

3-D Man	$16.00

ROMOX

Princess and Frog (20)	$24.00
Anteater (20/64)	$26.00
Typo (20/64)	$26.00
Whiz Kid (20/64)	$26.00

RAINBOW
C-64 Disks

Personal Finance	$48.00
Writer's Assistant	$95.00

MICROSPEC
VIC 20 Cassettes

Spelling Bee-Grades 2, 3, 4, 5, or 6	$8.00
Math Drill	$8.00
Portfolio Manager	$16.00
Data Manager	$16.00

VIC 20 Disks

General Ledger	$69.00
Mailing List Manager	$35.00
Inventory Package	$69.00
Payroll	$69.00
Data Base	$49.00

C-64 Cassettes

Black Box	$12.00
Color Sketch	$20.00
Match Maker	$16.00

C-64 Disks

Mailing List Manager	$45.00
Inventory Package	$79.00
General Ledger	$79.00
Data Base	$69.00
Black Box	$16.00
Color Sketch	$22.00
Match Maker	$20.00

TRONIX
VIC 20 Cassettes

Galactic Blitz	$19.95
Swarm	$22.95
Sidewinder	$22.95

VICTORY
C-64/VIC 20 Cassettes

Adventure Pak I(3 games)	$12.00
Adventure Pak II(3 games)	$12.00
Annihilation	$16.00
Grave Robber	$11.00
Kongo Kong	$16.00
Trek	$11.00

PROFESSIONAL SOFTWARE

Word Pro 64	$65.00

WEST 1-800-648-3311

EAST 1-800-233-8950

VISA

In NV call (702)588-5654. Dept 1106a Order Status # 588-5654
P.O. Box 6689, Stateline, NV 89449

In PA call (717)327-9575 Dept 1106a Order Status # 327-9576
477 E. Third St Williamsport, PA 17701

MasterCard

No risk, no deposit on C.O.D. orders. Pre-paid orders receive free shipping within the UPS continental United States with no waiting period for certified checks or money orders. Add 3% (minimum $3.00) shipping and handling on all C.O.D. and credit card orders. Larger shipments may require additional charges. NV and PA residents add sales tax. All items subject to availability and price change. We stock manufacturer's and 3rd party software for most all computers on the market. Call today for our catalog responding *number* in the DATA statement. If the computer finds an alphabetic character or symbol, you'll get a BAD DATA error.

Here's an example using string and numeric expressions:

```
10 FOR I=1 TO 4
20 READ NAME$,AGE,C$
30 PRINT
40 PRINT NAME$;" IS ";AGE;"YEARS OLD."
50 PRINT "FAVORITE COLOR: ";C$
60 NEXT I
70 DATA RICH,12,BLACK,BOB,7,BLUE,RANDY,3
80 DATA GREEN,GRANT,3,RED
```

This example illustrates a process repeated several times. Three variables are read each time, NAME$, AGE, and C$. The first time through the program loop, the data items are read and assigned as follows. NAME$="RICH", AGE=12, and C$="BLACK". Each time through the loop three more items are read. Note that it doesn't matter if all three items are not in the same DATA statement. Each DATA statement may consist of one item or several items separated by commas, although it saves memory to put as many items as possible in each DATA statement. The computer accepts up to four VIC or two Commodore 64 screen lines for each program line. Be sure you do not put a comma at the end of the list of items.

A common use of DATA and READ statements is to read variables into an array (subscripted variables). An example is:

```
10 FOR I=0 TO 10
20 READ A(I)
30 NEXT I
40 DATA 7,4,3,5,6,2,7,3,8,5,3
```

The first time through the loop, I is zero and A(0) will be given the value of 7, the first data item. The second time through the loop I is incremented to 1, and A(1) is assigned 4. The process continues.

Saving Memory And Time

If you like to write music on your computer, you may find using DATA statements is a way to play many notes without writing too many program lines. In the following examples I have not made the DATA lines as long as they could be; you can put as many items in one line as there is room. In these examples, line 10 turns on the volume. Lines 20-40 set up different variables for creating sounds on the Commodore 64. Lines 50-110 are the procedure to READ the note or tone number and the length from the DATA statements, then play the note. The length is determined in line 90.

Commodore 64 Version

```
10 POKE 54296,15
20 POKE 54277,4
30 V1=54273
40 V2=54272
50 FOR C=1 TO 14
60 READ N1,N2,L
70 POKE V1,N1:POKE V2,N2
80 POKE 54276,17
90 FOR D=1 TO 100*L:NEXT D
100 POKE 54276,16
110 NEXT C
120 POKE 54296,0
130 DATA 34,75,1,34,75,1,51,97,1,51,97
140 DATA 1,57,172,1,57,172,1,51,97,2
150 DATA 45,198,1,45,198,1,43,52,1
160 DATA 43,52,1,38,126,1,38,126,1
170 DATA 34,75,4
180 END
```

VIC-20 Version

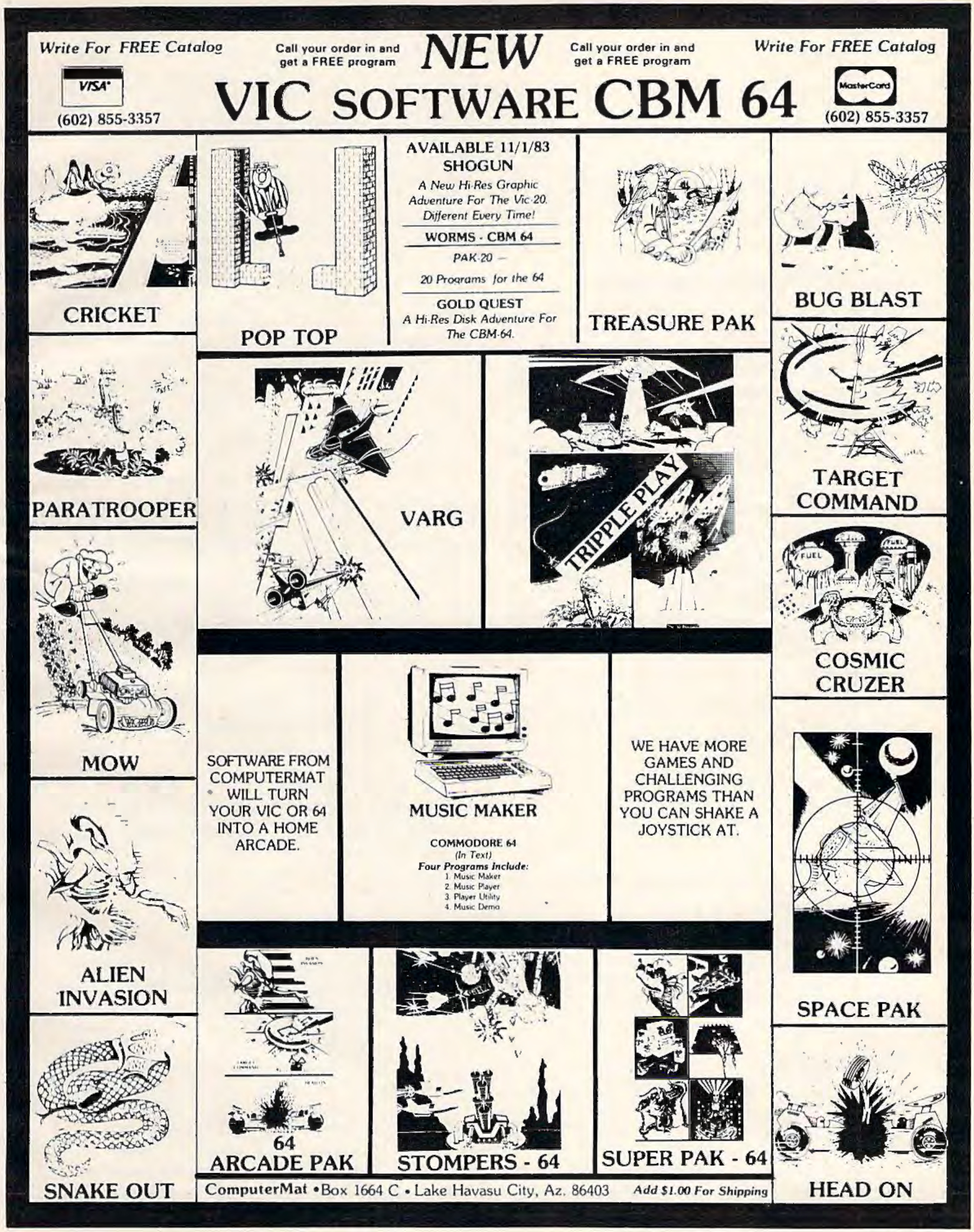
```
10 POKE 36878,15
50 FOR C=1 TO 14
60 READ N,L
70 POKE 36876,N
90 FOR D=1 TO 100*L:NEXT D
100 POKE 36876,0
110 NEXT C
130 DATA 195,1,195,1,215,1,215,1
140 DATA 219,1,219,1,215,2,209,1
150 DATA 209,1,207,1,207,1,201,1
160 DATA 201,1,195,4
180 END
```

You may also like to try graphics using DATA and READ statements. To POKE graphics onto the screen, refer to the user's guide that came with your computer. Look at the screen and color memory maps, page 144 for the VIC-20 and pages 138-39 for the Commodore 64. These maps give you the location numbers. I usually sketch a picture on graph paper corresponding to these maps. The character numbers that you POKE or place on the screen are called screen codes or screen display codes and are on pages 141-42 in the VIC-20 manual and pages 132-34 in the 64 manual. The following sample programs clear the screen, then display 16 graphic characters to draw a picture.

Line 2 in the Commodore 64 version changes the screen to white. Line 5 clears the screen. Lines 10-50 POKE the graphics on the screen, and lines 60-90 contain the data for the graphics. Line 20 READs S, the screen location, and C, the screen character code number, then line 30 places that graphic character on the screen. Line 40 assigns yellow to that character's location.

VIC Face

```
5 PRINT"{CLR}"
10 FOR I=1 TO 16
20 READ S,C
30 POKE S,C
40 POKE S+30720,7
50 NEXT I
60 DATA 7865,85,7866,67,7867,67,7868,73
70 DATA 7887,66,7888,81,7889,81,7890,93
80 DATA 7909,66,7910,74,7911,75,7912,93
90 DATA 7931,74,7932,64,7933,64,7934,75
100 END
```


Write For FREE Catalog
Call your order in and get a FREE program
NEW
Call your order in and get a FREE program
Write For FREE Catalog
VISA
(602) 855-3357
VIC SOFTWARE CBM 64
MasterCard
(602) 855-3357
CRICKET
POP TOP
AVAILABLE 11/1/83
SHOGUN
A New Hi-Res Graphic Adventure For The Vic-20. Different Every Time!
WORMS - CBM 64
PAK-20 —
20 Programs for the 64
GOLD QUEST
A Hi-Res Disk Adventure For The CBM-64.
TREASURE PAK
BUG BLAST
PARATROOPER
VARG
TRIPPLE PLAY
TARGET COMMAND
COSMIC CRUZER
MOW
SOFTWARE FROM COMPUTERMAT WILL TURN YOUR VIC OR 64 INTO A HOME ARCADE.
MUSIC MAKER
COMMODORE 64
(In Text)
Four Programs Include:
1. Music Maker
2. Music Player
3. Player Utility
4. Music Demo
WE HAVE MORE GAMES AND CHALLENGING PROGRAMS THAN YOU CAN SHAKE A JOYSTICK AT.
ALIEN INVASION
SPACE PAK
64
ARCADE PAK
STOMPERS - 64
SUPER PAK - 64
SNAKE OUT
ComputerMat • Box 1664 C • Lake Havasu City, Az. 86403
Add $1.00 For Shipping
HEAD ON

64 Face

```
2 POKE 53281,1
5 PRINT "{CLR}"
10 FOR I=1 TO 16
20 READ S,C
30 POKE S,C
40 POKE S+54272,7
50 NEXT I
60 DATA 1402,85,1403,67,1404,67,1405,73
70 DATA 1442,66,1443,81,1444,81,1445,93
80 DATA 1482,66,1483,74,1484,75,1485,93
90 DATA 1522,74,1523,64,1524,64,1525,75
100 END
```

The RESTORE Statement

The RESTORE statement tells the computer to RESTORE the data and start with the very first data item on the next READ statement. You may want to use the RESTORE statement if you want to do a procedure more than once but hate typing numbers in DATA statements. If you program a song, you can play the song more than one time by putting a RESTORE statement before the first statement that reads the first note, and put the whole procedure in a FOR-NEXT loop for the number of times you want the song to play, or a GOTO loop that plays continuously.

Let's look at an example. Suppose you have just drawn a face with one of the above programs. Now you want another face in another section of the screen. The screen characters used will be the same, and the relative positions of the characters will be the same. If we would like to use the same DATA statements we already have in the program, we can RESTORE the data for the next procedure. Try it by adding these lines to the "VIC Face" or "64 Face" program above.

```
92 RESTORE
93 FOR I=1 TO 16
94 READ S,C
95 POKE S+104,C
96 POKE S+104+30720,3
97 NEXT I
```

For the Commodore 64, change line 96 to:

```
96 POKE S+104+54272,3
```

With a little practice and practical experience you'll be able to see patterns in your work and learn to use DATA and READ statements along with RESTORE statements in the most efficient way.

Debugging

I mentioned that I think DATA statements are the most likely place to have an error when you are typing in a program from a published listing. Here are some suggestions to help you find the error.

When the program stops with an error message, you can PRINT the values of any variable and the computer will tell you the current value of that variable. As soon as you edit the program (make changes or add or delete a line), the variables will be zero, so PRINT the values first.

Let's say you are trying to RUN the VIC Face program above (the first version without the RESTORE lines). Now suppose the program prints part of the face, then stops with an error message. First, you can look to see how much of the face appeared before the error. If about half the face got printed, then the first half of the DATA statement items are OK. You may also type PRINT I and press RETURN, and the current value of I will be printed. You may also PRINT S,C to see what the values of S and C are. S should be a four-digit number starting with 7, and C should be a two-digit character number. If you know the value for I and the face looks all right as drawn so far, you can count the number of pairs of numbers that you know are OK to try to pinpoint the trouble area. You may also list certain line numbers. For example, type LIST 60-90, then press RETURN to see the DATA statements.

When you list the DATA statements, you can compare the screen listing to the published listing. Make sure you distinguish between the number zero and the letter O. Make sure you have the right number of commas in the right order.

If you get an OUT OF DATA error, the computer has read all the data items but needs more. List the DATA statements and make sure you have the right number of commas. You should also check any program statements, such as FOR statements, to make sure the numbers are typed correctly. The program may be trying to make too many loops. In the Face programs, you could LIST 10 to make sure line 10 ends with a 16 and not a 17.

If you get a BAD DATA error, it means the computer was trying to read a value for a numeric variable but the DATA statement contained a string. Again, check for commas in the right places. You may also try PRINTing some previous variables to see the latest correct value.

Bake A Cake

This program illustrates the use of DATA, READ, and RESTORE statements in a recipe file. First, you may go through a list of ingredients and press Y (yes) if you have the ingredient or N (no) if you do not have the ingredient. When the inventory list is complete, the computer program will tell you which cakes can be made with the ingredients you have. You may then choose a cake recipe, and that recipe will be printed on the screen. You then have the option to convert the recipe. If you would like to convert the recipe, enter a number or decimal fraction – such as 3 to triple the recipe or .5 to halve the recipe. The converted recipe will be printed.

For convenience in programming (especially with the limited memory of the VIC-20), the amounts in the recipes are given in decimals. For example, 2/3 cup sugar is written as .67 c. sugar. In the Commodore 64 version you may want to change the decimals to fractions.

The DATA statements in lines 101-107 contain the measure, then the ingredient, for 25 ingredients. Line 3 READs A$, the measure, and B$(N,0), the name of the ingredient, then assigns I$(N) equal to the measure plus a space plus the name of the ingredient. By the way, that's a zero in the subscript for B$. B$ is used in the inventory list, and I$ is used in printing the recipe. Z is the number of ingredients minus one (because the subscripts start with the number zero).

As you are typing the DATA statements, you may notice two or more commas together with nothing between them (,,,). Be sure you get the right number of commas as you are typing. This indicates a null string, or a string variable equal to "". You may also use "" in your DATA statements (two double quote marks with no space between).

The recipes are in the DATA statements in lines 109-123. The first item is the name of the cake. The next items are the amounts of the ingredients in the following order: cups shortening, cups flour, cups sugar, cups brown sugar, tsp. baking powder, tsp. salt, tsp. soda, cups cherry juice, number of cherries, cups bananas (mashed), cups sauerkraut, cups milk, cups buttermilk, number of eggs, number of egg whites, tsp. red food coloring, ounces of chocolate, tbsp. cocoa, tsp. vanilla, tsp. cinnamon, tsp. nutmeg, tsp. vinegar, cups salad oil, cups water, and cups of oatmeal. The DATA will contain a null string if the cake does not contain that ingredient.

An example is the first part of line 109, the data for banana cake:

```
109 DATA BANANA,.67,2.5,1.67,,1,1,1,,,1,,
    ,.67,2,,,,,,,,,,,,CHERRY ...
```

The name of the cake is BANANA. The recipe is .67 cup shortening, 2.5 cups flour, 1.67 cups sugar, 1 tsp. baking powder, 1 tsp. salt, 1 tsp. soda, 1 cup mashed bananas, .67 cup buttermilk, and 2 eggs.

Program Explanation

How does the program work? To print a recipe, the data is RESTOREd first, then the first 50 ingredients and measures are read and ignored (line 73). The key you pressed to choose a recipe is E$, and the ASCII code will be 65 for A, 66 for B, and so forth. Lines 75-77 figure out which recipe was chosen and read through the title and ingredients to get to the appropriate recipe. Line 79 prints the title of the cake. Line 81 reads the amount from the DATA statement. If the amount is a null or zero, then that ingredient is not printed. However, if there is a value, the value is printed, followed by the corresponding measure and name of ingredient from the I$ array.

Line 83 defines variables M(I) for measure and C$(I) for ingredient for only those ingredients in the recipe. These values are used in printing the converted recipe, lines 97-99.

For the inventory list, the computer keeps track of your Y or N answers in the B$(N,1) array, where N varies from 0 to 24 for the ingredients. Line 33 checks to see if an N is stored as a no answer for flour, sugar, or salt. If any one of these three items has a no, then no cakes can be baked. Y is a variable for the number of Y answers, and if there are not enough ingredients with a Y answer you cannot make a cake (checked in line 35).

Line 47 RESTOREs the data, then ignores the first 50 items. Lines 49-51 check through the recipe for each cake. If there is an amount listed for an ingredient, then the corresponding B$(N,1) value is checked. If it is N for no, then you are lacking one of the ingredients required for the cake. The rest of the ingredients are skipped over and the computer goes to the next recipe. If each of the required ingredients also has a B$ value of Y for yes, then the cake can be made and the name of the cake is printed.

These are real recipes that really work. No baking instructions are given because there isn't room in the VIC-20 version, but usually the person baking knows how to bake the cake and just needs to be reminded of the amount of each ingredient. In case you want to try these recipes, all are for cakes to be baked in two 9-inch layers. Mix the shortening with the sugars, then add the dry ingredients alternately with the liquid, then add the eggs and vanilla. Bake at 350 to 375 degrees.

In the cherry cake, cut up the cherries before adding to the mixture. For the oatmeal cake, prepare the oatmeal separately. While the oatmeal cools, mix everything else, then add the oatmeal to it. The wacky cake can be mixed all at once. Just dump everything into one big bowl and mix it up. For the red velvet cake combine the ingredients as usual, except for the vinegar and soda. Mix the vinegar and soda together and fold into the rest of the batter. The sauerkraut cake is worth trying – just don't tell anyone what it is until after they have eaten it. Rinse the sauerkraut well, then chop it into small pieces before combining it with the rest of the cake batter.

Typing In The Program

The VIC-20 version is very close to full memory. Be sure to leave out all the spaces and be sure to use the abbreviations for all the BASIC words, such as ? for PRINT and D shift-A for DATA. If you need to

edit a line, be sure the edited line also contains the abbreviations.

Be sure you copy the DATA statements exactly for the correct recipes. There are no spaces except in the names of the cakes. If there are commas together, do not put spaces between the commas.

For the Commodore 64 version, please add line 2 POKE 53281,1 to change to a white screen, and change the following lines:

```
17 PRINT"{CLR}{BLU}IN THE FOLLOWING LIST,
    PRESS":PRINT "{2 SPACES}'Y' IF YOU HA
   VE THE INGREDIENT"
19 PRINT "{2 SPACES}'N'IF YOU DO NOT
   {DOWN}":PRINT"{2 SPACES}'S' TO START O
   VER.{2 DOWN}"
63 PRINT "D DEVIL'S FOOD CAKE":PRINT "E G
   OLD LAYER CAKE":PRINT "F OATMEAL CAKE"
99 FOR N=Ø TO I-1:PRINT INT(F*M(N)*1ØØ)/1
   ØØ;TAB(6);C$(N):NEXT:GOTO87
```

These lines adjust the printing for the Commodore 64 screen.

If you prefer to save typing effort and time, you may receive a copy of "Bake A Cake" by sending $3, a blank cassette, and a stamped, self-addressed mailer to C. Regena, P.O. Box 1502, Cedar City, Utah 84720. Be sure to specify which computer version.

See program listing on page 202.

One-Touch Commands For The 64

David Martin

This utility program is an ideal application for the normally unused special function keys to the right of the Commodore 64 keyboard. An entire command can be typed on the screen with a single key press. Programmers in particular will appreciate the repetitive typing this technique can save.

Unlike people, computers excel at performing boring, repetitive tasks. What's more, time-consuming tasks which annoy us can be performed by an uncomplaining computer in a fraction of a second. So it only makes sense to let computers handle the simple little things they do best.

One of these tiny jobs is the routine typing of frequently used commands. During a session with your computer, how many times do you type RUN, LIST, SAVE, or LOAD? Probably many more times than you think. If you're a hunt-and-peck person new to typewriter-style keyboards, this can be a major annoyance. Even if you're a fast touch-typist, you probably stumble over such often-used commands as POKE 53281,1:PRINT {BLK} (which sets up an easier-to-read white screen background with contrasting black characters).

The utility presented here can liberate you from all that. It redefines the special function keys (F1 through F8 to the right of the keyboard) so that a single key press enters a whole command. The short while it takes to type in this program can pay for itself many times over.

One-Touch Commands

Be sure to type the program carefully. As always, save it twice on tape or disk before running it for the first time. The program is in the familiar form of a BASIC loader – a BASIC program which includes a machine language program encoded in DATA statements. A mistyped number can "crash" the computer when the program is first run, forcing you to switch off/on to clear the machine. Saving the program beforehand can keep you from losing all your work.

Actually, this BASIC loader contains two machine language programs. Neither program consumes any memory normally used by BASIC (see Programmer's Notes below). After activating the utility, it erases the BASIC loader from memory and allows you to load your own programs. The utility keeps working "in the background," so to speak, until you turn off the computer or reset it by pressing RUN/STOP-RESTORE.

The utility is very easy to use. First, enter and run the BASIC loader. You'll see a screen prompt which asks:

F1?

Now, type in whatever command you'd like to have available at a stroke of the F1 key. Then press RETURN. For instance, if you answer the prompt by typing LIST and pressing RETURN, then hitting F1 after the utility is activated will print the command LIST on the screen.

There's a way to save even more key strokes, too. If you answer the prompt by typing the command followed by a reverse arrow – using the reverse-arrow key in the upper-left corner of the keyboard – then the utility will even press RETURN for you, when activated. Otherwise, it will be up to you to press RETURN when using each command. In other words, answering the prompt like this:

F1? LIST [*Press RETURN*]

means that when the utility is working, it will type the command LIST on the screen for you, but you'll still have to press RETURN yourself to actually execute the command. But if you answer the prompt like this:

F1? LIST ← [*Press RETURN*]

it means the utility, when working, will type LIST *and* press RETURN for you when you hit the F1 key. The reverse-arrow makes the command *self-executing*. Pressing the function key will execute the command instantly. Depending on the command, this may or may not be desirable. For instance, you probably wouldn't want the command NEW to execute instantly because it would be too easy to accidentally wipe out a BASIC program. (In fact, you probably wouldn't want to program a function key with NEW at all.)

You can also answer the prompt with more than one command. An example might be:

F1? LOAD ← RUN ← [*Press RETURN*]

which means F1 will automatically LOAD and RUN the next program from tape.

After answering the F1? prompt, the utility asks for F2, F3, and so on through F8. After F8, the utility immediately activates itself and erases the BASIC loader from memory (you did SAVE it, didn't you?).

The function keys are now programmed. They will remain so until you shut off the computer or trigger a "warm start" by pressing RUN/STOP-RESTORE.

Programmer's Notes

The one-touch command utility consists of two machine language programs tucked away in different parts of the Commodore 64's memory. The first part is in the cassette buffer, starting at memory location 828 ($033C hexadecimal). This program asks for the key definitions. Each time RETURN is pressed, it stores the ASCII values of the characters into high memory.

After entering F8, control jumps to the second program, stored in high memory at location 49152 ($C000 hex). This is a 4K block of unused memory in the 64. The first two POKEs in the first line of the BASIC loader fool BASIC into thinking that memory ends at 53248. To restore normal vectors, you can enter POKE 56,160:POKE 55,0.

The first machine language program also sets up an interrupt. Every sixtieth of a second, the computer checks the second program to see if a function key has been pressed. If so, the key's definition is printed on the screen. If a reverse-arrow was defined after the command, the program forces a RETURN to execute the command also.

See program listing on page 202.

VIC/64 Disk Defaulter

Eric Brandon, Programming Assistant

This month's "Power BASIC" — a continuing series of useful utilities and routines — saves typing for people who regularly use a disk drive instead of a cassette recorder. The machine language routines are in the form of easy-to-use BASIC loaders.

When Commodore designed the operating system used in the VIC-20 and Commodore 64, the designers assumed that most people would be using a cassette recorder for storage instead of the more expensive disk drive. That's why, when you type LOAD or SAVE, the computer responds by prompting "Press Play On Tape" or "Press Record & Play On Tape." It *defaults* to the tape recorder.

If you're using a disk drive, you have to type the device number —,8— after each command (as in LOAD"filename",8). This can become bothersome after a while.

"Disk Defaulter" is a short utility, written in machine language, that modifies the computer's operating system to recognize the disk drive as the default device instead of the cassette recorder. As long as the utility is activated, you no longer have to append ,8 to the LOAD, SAVE, and VERIFY commands.

To use Disk Defaulter, enter Program 1 for the VIC-20 or Program 2 for the Commodore 64. When you type RUN, this BASIC loader will POKE the machine language into some free memory space and activate the utility. To turn it off (for instance, if you want to use cassette), press RUN/STOP-RESTORE. To turn it back on, type SYS 679.

To load machine language programs, you still must type LOAD"filename ",8,1. Also, pressing SHIFT-RUN/STOP will not access the disk drive because it results in a "Missing Filename Error." But otherwise, all LOAD, SAVE, and VERIFY commands will refer to disk.

The only program we have found that will interfere with Disk Defaulter is the PAL Assembler for the Commodore 64.

Program 1: VIC Version

```
10 I=679
20 READ A:IF A=256 THEN 1000
30 POKE I,A:I=I+1:GOTO 20
679 DATA 169,188,141,48,3,169,2
686 DATA 141,49,3,169,195,141,50
693 DATA 3,169,2,141,51,3,96
700 DATA 162,8,134,186,76,73,245
707 DATA 162,8,134,186,76,133,246,256
1000 PRINT"DISK DEFAULTER ACTIVATED
1010 PRINT"USE RUN/STOP RESTORE TO DEACTI
     VATE
1020 PRINT"TYPE SYS 679 TO REACTIVATE
1030 SYS 679
```

Program 2: 64 Version

```
10 I=679
20 READ A:IF A=256 THEN1000
30 POKE I,A:I=I+1:GOTO20
679 DATA 169,188,141,48,3,169,2
686 DATA 141,49,3,169,195,141,50
693 DATA 3,169,2,141,51,3,96
700 DATA 162,8,134,186,76,165,244
707 DATA 162,8,134,186,76,237,245,256
1000 PRINT"DISK DEFAULTER ACTIVATED
1010 PRINT"USE RUN/STOP RESTORE TO DEACTI
     VATE
1020 PRINT"TYPE SYS 679 TO REACTIVATE
1030 SYS 679
```

EPYX
TOP 10 GAME SALE
* BUY TWO — GET ONE FREE *

SPECIAL FREE GAME
Buy Two Epyx Games From Protecto And Get A Free Game

The Thinking Man's Paradise

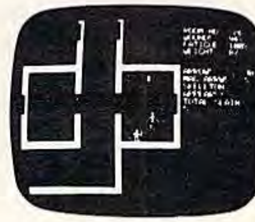
(Disk / Cassette)

- **Award Winning Games**
- **Fantastic Graphics**
- **Skill (not luck) Needed**
- **Already in top 10 charts**

COMMODORE - 64 / VIC-20

Buy Any Two Epyx Games From Protecto And Send The Proof Of Purchase Seals To Epyx And Epyx Will Send You A Free Game.

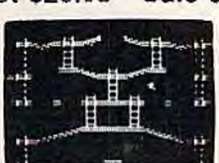
JUMPMAN
If you like Donkey Kong, you'll love Jumpman. Over 30 different screens with 8 speeds and 5 skill levels make this the fastest action game in the country. You must leap girders, climb ropes, and scale ladders to reach and diffuse bombs while avoiding robots, birds, bullets, explosives, crumbling walls, vanishing escape routes, and many other obstacles. (Truly a fantastic game!)
List $39.95 **Sale $27.95**



TEMPLE OF APSHAI (computer game of the year)
This is the standard by which other adventure games are judged. Full color graphics portray the temple and all its contents - magic, monsters, doomed cities and damsels in distress. Do battle in real time with over 20 monsters, expansion modules will keep your adventure alive in the future.
List $39.95 **Sale $29.95**

SWORD OF FARGOAL
Search for the wondrous sword in the depths of an ever changing dungeon. Make yourself invisible, teleport to a new location, drink a healing potion or use enchanted treasures, but watch out for traps and hideous creatures who will try to stop you. (Fantastic dungeon adventure)
List $29.95 **Sale $21.95**

CRUSH CRUMBLE & CHOMP (Computer game of the year nominee)
Choose one of six monsters or create your own, and use your monster to destroy one of four unsuspecting cities. The cities aren't totally defenseless, they call on police, national guard, and even a mad scientist, complete with helicopter, to save humanity from the relentless threat.
List $29.95 **Sale $21.95**



JUMPMAN JUNIOR
The devilish ALIENATORS are back! And they have overrun the Jupiter Command Substation. In this cartridge format sequel to the best-selling Jumpman, players must leap through 12 all new screens featuring electrocution traps, moving walls, hellstones, and dangers much too bizarre to be believed. How many screens can you master? Twelve different screens, 8 speeds.
List $39.95 **Sale $27.95**

We have all Epyx games

PROTECTO
ENTERPRIZES (WE LOVE OUR CUSTOMERS)
BOX 550, BARRINGTON, ILLINOIS 60010
Phone 312/382-5244 to order

RICHARD MANSFIELD, SENIOR EDITOR

The Assembler

People often use the words *machine language* and *assembly language* interchangeably. However, *machine language* is becoming the more common term; it is more accurate — when you program in this language, you're speaking directly to your computer in its native tongue.

Unfortunately, the computer's internal language is almost impossible for humans to work with. These machines communicate only with numbers, and very odd numbers at that. They're binary, consisting of only 1's and 0's, grouped together in eight-digit clusters called *bytes:* 01100111, 11110001, and so on. Humans find it easier to work with words. That's where an *assembler* comes in.

The Primary Tool

This month we're going to build the basic tool for machine language (ML) programming. Type in Program 1 and you'll have your own working assembler. (Be careful with line 244; it's too long to type in normally. You'll need to abbreviate some of the BASIC words to make it all fit on one line. Use gO for GOTO, ? for PRINT, and pO for POKE. Remember to SHIFT the second letter of each abbreviation.)

The assembler works like this: you type in a wordlike, three-letter code, and the assembler looks up the correct number (in the computer's language) and POKEs it into RAM memory to start forming an ML program. In a minute we'll create a simple ML program to show you how ML programming is done. But let's clear up a few possible sources of confusion first.

These wordlike codes are called *mnemonics,* which means they've been designed to be simple to remember. It's easy enough to remember what USA stands for. Likewise, you can quickly pick up the essential ML words. There are 56 of these commands available to you, roughly as many words as there are in BASIC. But, like BASIC, there is a core group of about 20 important ones. They are the only ones you need to use to get almost anything accomplished. What's more, the ML words *are* easy to learn and remember. For example, BRK stands for Break (like BASIC's STOP), JSR is Jump to SubRoutine (GOSUB), and RTS is ReTurn from Subroutine (RETURN). The command which does the same thing as BASIC's GOTO is called JMP, for JuMP.

A Kind Of Swing

ML programming involves a kind of swing between Command and Target. First you give a command, then you give the specific target for that command. Then another command, another target. These paired-event phenomena are called by many names and appear in many disguises in programming as well as in real life. They're called Operator/Operand, Instruction/Argument, Mnemonic/Address, Analyst/Analysand, Shopper/Apples, Thief/Victim. Notice that the first half of the pair is the more general, the second more specific. At a given moment, the apple is the specific thing the shopper's involved with, but the shopper will be buying other things during this visit to the store. Similarly, a thief is always a thief, but a victim is a victim only that once (we hope). Also, the transaction which all these pairs have in common is that the first half of the pair is *doing something to* the second half. Together they form a complete action in the sense that Open/Envelope or Eat/Peach are paired (command/target) actions.

A Robot Thanksgiving

If you think about it, you can see this do-it-to-it rhythm throughout BASIC programming: PEEK (8), PRINT "HELLO", SAVE "PROGRAM", X=15, X=X+1, GOTO 1500, etc. The reason

we're stressing this distinction, this rhythmic swing between actor and acted-upon, is because an ML program is constructed in precisely this way — you make a list of tiny, elementary actions for the computer to later carry out. It's like a robot Thanksgiving dinner: spear/turkey, raise/arm, insert/food, chew/morsel, lower/arm, spear . . . . But list enough of these mini-instructions and you can do amazing things.

One result of all this is that an ML program doesn't look like a BASIC program. BASIC tends to spread these pairs out along a line:

```
100 Y=3: X=X+1: POKE 63222,Y: Y=PEEK
    (1200)
```

ML lists each tiny action-pair on its own line:

```
100 LDY #3
110 INX
120 STY 63222
130 LDY 1200
```

These two programs are doing exactly the same thing, but in different ways. STY and LDY mean STore Y and LoaD Y (it's like a variable in BASIC). INX means INcrement X (raise it by one). The # sign means to think of the number as *literally* the number three, not address three. Without the #, the computer assumes you mean a memory location.

Take a look at the mnemonics here. They're all three-letter words. They are always the first thing on each line. And they usually have their target right next to them (the INX doesn't because the mnemonic itself already contains the specific information required). The other half of the pair, those numbers, are called *addressing modes* in ML. In general, that's because numbers are usually being sent to and from addresses in the computer's memory while an ML program is running. That, plus simple arithmetic, is the essence of what a computer does to accomplish any given task.

We'll get to the addressing modes (there are about ten) in a future column, but you can already recognize two of them: line 100's mode is called *immediate addressing* (the number is immediately after the instruction, not in some memory location elsewhere in the computer) and line 110's mode is called *implied addressing* (because the instruction contains its own target).

Putting The Assembler To Work

Enough theory, let's do something. Let's assemble a small program. If you've typed in Program 1, the first thing to do is to change line 10 so that the assembler will accept ordinary decimal numbers. It's designed to work with either decimal or hexadecimal, but we've not yet touched on hex so we'll stick with the familiar. Change the line to:

```
10 H=0
```

Then RUN the assembler and type in 830 when it asks you where you want to put the ML program. That's a safe place until you next load in a program from cassette. ML can be put into a variety of places in RAM. BASIC, of course, has a computer-determined starting location in memory, but *you* specify the start of an ML program. Now you'll see that address printed on screen. The addresses where the instructions are being stored will function as the "line numbers" for your reference when programming. Unlike BASIC, you can't go back up and change a line. If you make a mistake, start over. (There are easier ways to fix errors, but that, too, is for a future column.)

Now type LDY #0, hit RETURN, and you've written a line of ML which will put a zero into the Y register. (You'll see the numbers forming the ML version of your program appear to the right of the mnemonic/address you've typed.) Then the assembler will furnish you with the next available "line number" address in RAM, 832. The mnemonic/address pair LDY #0 uses up two bytes.

You are ready to type in your next pair: LDA #66. Hit RETURN on this line and you've put the code for the letter B into the A register. Then type in the rest of our ML program, one pair per line:

```
JSR 65490
DEY
BNE 834
RTS
```

That's it. To let the assembler know that you're through with your program, type END instead of a normal mnemonic and it will tell you the start and end addresses of your ML program. Then, having done its job, the assembler quits. The mnemonics and addresses were all POKEd into their proper places after being translated into the machine's language. To see what happens when this RUNs, you can type SYS 830 and see the effect of the small ML loop we wrote. You'll get 256 B's on screen in record time. Not something you've been anxious to do? More useful things are on their way.

In the coming months we'll go into detail about these mnemonics and instructions. We'll also make some modifications to the assembler itself. It's written in BASIC, after all, and there are some valuable things to learn about ML by simply adding to the functions of the assembler program. You'll likely find yourself using the assembler frequently as you get a better feel for ML programming.

If you have any questions or suggestions, write to me c/o COMPUTE!'s Gazette.

See program listing on page 200.

HORIZONS: 64

CHARLES BRANNON
PROGRAM EDITOR

Last month, I promised we'd look at some of the latest hardware and software for the Commodore 64. This isn't meant as a review, but it will give you a first look at these products in less time than it takes to do a full-scale evaluation. You'll generally see the in-depth review in a later issue of either COMPUTE!'s Gazette or COMPUTE! Magazine.

Neutral Zone. This is a nicely done multicolor space game which finely scrolls left and right. There are some attempts at 3-D sprite animation, and the sound effects are OK. The game is written entirely in machine language, so joystick response is quick. Graphically, the game is quite beautiful, with multicolored space scenes, cruising starships, and a gigantic mothership.

One interesting feature is that *Neutral Zone* requires a "dongle" that plugs into joystick port one. Although you can copy the tape or disk, it will not run on a machine without the dongle. This is a good way to address the problem of software piracy. It lets you make backup copies, but protects the manufacturer from unethical copying. Let me know what you think of this, and other, approaches to software piracy. ($34.95 tape/disk, Access Software, 925 East 900 South, Salt Lake City, UT 84105.)

Screen Graphics 64. Tired of all those PEEKs and POKEs? *Screen Graphics 64* adds 24 commands to BASIC to make graphics programming a good bit easier. Most of the commands are obvious: HIRES, MULTI, TIC, DOT, DRAW, BOX, CIRCLE, CHAR, BLOCK, MODE, FILL, PIXEL. These control the high-resolution screen and draw axes, dots, lines, rectangles, circles, and solid rectangles; fill any shape, change dot modes (enabling you to reverse screen areas); add any text from any set to the screen; and return the dot stored at pixel position X,Y.

The coordinate drawing system is based with 0,0 at the lower-left corner of the screen — great for mathematicians, but it will make converting Apple and Atari programs a little more difficult.

There are also some commands to make sprite programming much easier. You can imbed sprite shapes in your programs, copy them, animate them, and remove sprites. Even multicolor sprites are easy to use.

There are many products on the market that add graphics commands to BASIC. Based on a similar VIC-20 product, *GRAPH-VICs,* this one is well implemented and easy to learn. ($24.95 tape/$27.95 disk, Abacus Software, P.O. Box 7211, Grand Rapids, MI 49510.)

Crossfire. This Apple translation by Sierra On-Line plays very well on the 64. The game is apparently done with high-resolution graphics. Even though most of the sprites go unused, the motion is smooth and fast. The sound and multipart music is especially well done. This game is so fast that if you aren't really sharp, your average game can be measured in seconds. The background music is almost hypnotic, but some may find it grating after a few hours of play.

In *Crossfire,* you move through the streets of a city, shooting aliens moving in all the horizontal and vertical corridors. The screen is filled with shooting, and it's hard not to blunder into an alien. The shapes of the creatures grow more complex as you play, and the difficulty skyrockets. Challenging and addicting. It's available at your local dealer for about $29.95 retail/disk (Sierra On-Line sells wholesale only).

The Zork Trilogy. If you can't enter sentences like this on your favorite word-based adventure game, it's time for *Zork:* "Open the brown bag, remove the bottle of water, open it and drink the water."

Zork puts the power of mainframe-based text adventures on your 64. It uses some of the most sophisticated techniques known. For example, to compact the verbose descriptions, words are stored using only five bits. This requires a special shift code to bank select different five-bit character sets, but it makes for very detailed prose. You control a "mind clone" through an underground kingdom on your way to fame and fortune. You battle trolls, cross gorges, find jewels, and solve puzzles. And watch out for the Grue!

Prices the Competition Can't Touch

Commodore 64

Avalon Hill Game Company
180-701 B-1 Nuclear Bomber (C) $12
180-702 Midway Campaign (C) $12
180-703 North Atlantic Convoy Raider (C) $12
180-704 Nukewar (C) $12
180-706 Planet Miners (C) $12
180-712 Computer Stocks & Bonds (C) $15
180-719 Andromeda Conquest (C) $14
181-721 Computer Football Strategy (C) $12
181-732 Telengard (C) $16

Broderbund
David's Midnight Magic $23
Choplifter (CT) $34
Serpentine (CT) $27
Sea Fox (CT) $27

Datamost
Roundabout $20
Bilestoad $20
Mating Zone $20

EPYX/Automated Simulations
14E-036 Jump Man (D) $27

Human Engineered Software (HES)
HEE-307 6502 Professional Dev. System (C) $23
HEE-400 Retro Ball (Crt) $27
HEE-401 Hesmon (Crt) $27
HEE-402 Turtle Graphics II (Crt) $45
HEE-404 Heswriter 64 (Crt) $35
HEE-412 Gridrunner (Crt) $27

Infocom
63E-001 Zork (D) $ 27
63E-002 Zork II (D) $ 27
63E-003 Deadline (D) $ 34
63E-004 Starcross (D) $ 27
63E-005 Zork III (D) $ 27

Sierra On-Line
54E-048 Frogger (D) $23
Jaw Breaker $20

Sirius Software
70E-036 Blade of Blackpoole (D) $27
70E-037 Type Attack (Crt) $27
70E-043 Repton (D) $27
70E-046 Critical Mass (D) $27
70E-424 Snake Byte (Crt) $23
70E-445 Spider City (Crt) $27
70E-447 Squish 'em (Crt) $23
70E-448 Final Orbit (Crt) $23

Spinnaker
SKE-001 Snooper Troops #1 (D) $30
SKE-004 Facemaker (D) $ 23
SKE-006 Kindercomp (D) $ 20
SKE-008 Hey Diddle Diddle (D) $ 20
SKE-009 In Search of the Most Amazing Thing $ 27
Fraction Fever (CT) $ 20
Alphabet Zoo (CT) $ 20
Delta Drawing (CT) $ 23

Snyapse Software
SSE-011 Ft. Apocalypse (D) $ 23
SSE-016 Drelbs (D) $ 23
SSE-019 Survivor (D) $ 23
SSE-020 Pharoah's Curse (D) $ 23
SSE-311 Ft. Apocalypse (C) $ 23
SSE-316 Drelbs (C) $ 23
SSE-319 Survivor (C) $23
SSE-320 Pharoah's Curse (C) $ 23

United Microwave Industries
92E-302 Renaissance (C) $ 20
92E-331 Motor Mania (C) $ 20

Commodore 64 .... $219
VIC 20 .... $ 99
1525 Printer $229
1530 Datasette .... $ 64
1541 Disk Drive $249
1600 Modem .... $ 89
1701 Commodore Monitor .. $289
VIC 1311 Joystick .... $ 8
VIC 1312 Game Paddles ... $ 16

VIC 1210 3K Memory Expander $34
VIC 1110 8K Memory Expander $52
VIC 1111 16K Memory Expander $89
VIC 1011 RS 232 Terminal Interface $43
VIC 1211 Super Expander $59
VIC 1213 Vicmon Machine Language Monitor $45
VT 106A Recreation Pack $45
VT 107A Home Calculation Pack $45
VIC 1600 Vicmodem $89
VM Programmer's Reference Guide $14

Educational

Books
Kids and the Vic $ 18
Programmer's Reference Guide (Vic) $ 14
Programmer's Reference Guide (64) $ 18

Language Arts
Super Hangman (C) $ 14
Simon/Hess (C) $ 13
Concentration (C) $ 13
Home Babysitting $ 23

Math
Sky Math (C) $ 12
Space Division $ 12
Bingo Speed Math (CT) $ 23
Number Crunch (CT) $ 23
Number Chaser $ 17
Number Gulper $ 17

Music
Vic Music Composer (CT) $ 42
HES Synthesound (CT) $ 49

Reading and Language Arts
Finger Spelling (D, C) $ 12
My Spelling Easel (Ages 3-10) $ 26

Programming Techniques
Intro to Basic Prog. I $ 22
Intro to Basic Prog. II $ 22
Programmer's Aid Cart $ 22
Turtle Graphics/Hess (CT) $ 29
Gorteck & the Microchip $ 23

Social Studies/Science
Visible Solar System $ 23
Reaganomics (CT) $ 27

Commodore Software
Avenger $ 23
Superslot $ 23
Super Alien $ 23
Jupiter Lander $ 23
Draw Poker $ 23
Midnight Drive $ 23
Radar Rat Race $ 23
Raid on Fort Knox $ 23
Sargon II Chess $ 29
Cosmic Cruncher $ 23
Gorf $ 29
Omega Race $ 29
Sea Wolf $ 23
Adventureland $ 29
Pirate Cove $ 29
Mission Impossible $ 29
The Count $ 29
Voodoo Castle $ 29
The Sky is Falling $ 23
Mole Attack $ 23
Bingo Speed Math $ 23
Home Babysitter $ 23
Visible Solar System $ 23
Personal Finance $ 29
Quick Brown Fox $ 65

Call Toll Free

1-800-634-6766

Order Line Only

COMPUTER OUTLET

1095 E. Twain (702) 796-0296
Las Vegas, Nevada 89109

Information & Order Inquiries (702) 369-5523

Mon. — Fri. 8 A.M. to 6 P.M. — Sat. 9 A.M. to 6 P.M.
Dealers' Inquiries Invited

Vic 20

Avalon Hill
Nuke War $12

Automated Simulations
Rescue at Rigel (C) $20
Ricochet (C) $15
Monster Maze (CT) $27
Sword of Fargoal $27

Broderbund
Martian Raider $15
Multisound Synthesizer $15
Shark Trap $15
Sky Blazer (CT) $27
Sea Fox (CT) $27
A.E. (CT) $27

Creative Software
Black Hole (CT) $36
Trashman (CT) $36
Astroblitz (CT) $36
City Bomber & Minefield (CT) $20
Apple Panic (CT) $36
Choplifter (CT) $36
Serpentine (CT) $36
Videomania (CT) $36
Terraguard (CT) $36

HES Software
HES Mon (CT) $ 29
HES Writer (CT) $ 29
Synthesound Music Synthesizer (CT) $ 49
Turtle Graphics (CT) $ 29
VIC Forth (CT) $ 45
Victrek (C) $ 15
Predator (CT) $ 27

Sirius
Type Attack $27
Snake Byte $27

Thorn EMI
River Rescue (CT) $ 29
Mutant Herd (CT) $ 29

Tronix
Galactic Blitz (C) $17
Swarm (C) $20
Sidewinder (C) $20
Gold Fever (CT) $27
Deadly Skies (CT) $27

United Microware
Spiders of Mars (CT) $ 34
Meteor Run (CT) $ 34
Amok (C) $ 15 (CT) $ 27
Alien Blitz (C) $ 17 (CT) $ 27
Skymath (C) $ 12
Space Division (C) $ 12
Super Hangman (C) $ 14
The Alien (C) $ 17
3D Maze (C) $ 12
Kosmic Kamikaze (C) $ 17
Sub Chase (C) $ 15
Renaissance $ 20
Cloud Burst (CT) $ 27
Satellites & Meteorites (CT) $ 34
Outworld (CT) $ 34
Wordcraft $ 65

We accept VISA and MasterCard

ORDERING INFORMATION AND TERMS: For Fast Delivery send cashier checks, money orders or direct bank wire transfers. Personal and company checks allow 3 weeks to clear. C.O.D. orders ($3.00 minimum) and 1% of all orders over $300. School purchase orders welcomed. Prices reflect a cash discount only and are subject to change. Please enclose your phone number with any orders. Shipping — Software ($2.50 minimum). Shipping — Hardware (please call). Foreign orders, APO & FPO orders — $10 minimum and 15% of all orders over $100. Nevada residents add 5¾% sales tax. All goods are new and include factory warranty. Due to our low prices, all sales are final. All returns must have a return authorization number. Call 702-369-5523 to obtain one before returning goods for replacement.

This is such a popular and well-done adventure that it has inspired two *Zork* sequels. There is a special *Zork* User Group, and paraphernalia such as cheat books, maps, bumper stickers, T-shirts, and buttons. *Zork* was originally written and sold by Infocom, Inc., a Cambridge, Massachusetts, company. Recently, Commodore itself has started marketing the *Zork* series. Infocom is earning a formidable reputation for its "interactive prose" programs. Other Infocom products for the 64 are *Starcross, Suspended* (a science-fantasy nightmare), *Deadline,* and *Witness* (both detective scenarios).

All of these are text-only adventures. Infocom says it "puts the graphics where the sun don't shine," in your brain, holding that the best graphics are in your imagination. This is arguable, but the fine detail of these games would be wasted on crude graphic displays. Some people like adventures with full-screen pseudo-realistic pictures, but the *Zork* trilogy and its kin really don't need them.

Telecommunicate!

How many of you out there own modems? These devices let your 64 send and receive data over the telephone. Boring? Hardly. With a modem, you can exchange programs with friends. You can dial up any of several Commodore-oriented bulletin boards. And then there's The Source and CompuServe. Having used them, I feel the under-$100 VICmodem is like adding the planet Earth to your computer. It significantly expands your personal sphere of communication. For the majority of people, one-way television and radio, plus an occasional phone call and letter, provide most of our long-distance needs. With a nationwide bulletin board, however, you're casually chatting with dozens of people from all over the country. On CompuServe Citizen's Band radio simulation, you can talk to people nationwide on such diverse subjects as computers, birth control, and peanut butter. I feel that this is substantially changing our "world concepts," when your next-door neighbor is 2000 miles away. (See the special telecomputing issue of COMPUTE!'s Gazette, September 1983.)

If there's sufficient interest, we'll talk about some of the latest developments in telecommunications, and show how you and your friends can make the most of your modems.

Rumors

Like rumors? There are some significant new products on the 64 horizon. For the memory-hungry, yes, there is memory expansion for the Commodore 64. One in particular is worth notice, the Monolith from Richvale Telecommunications. This not-yet-released cartridge is supposed to add 512K of RAM memory (1/2 megabyte, or 524,288 bytes)!

In addition, the Monolith has an on-board 68000 co-processor. This is the daddy microprocessor of the new high-speed, super-powerful 16-bit chips, and is used in Apple's Lisa. Expect big-system performance. The price? I hear it will be around $350.

The Unexploited SID

What are you doing with your 64's fantastic built-in sound synthesizer? Not many programmers seem to be exploiting its advanced features. There should be more experimentation with the filters, ring-modulation, and synchronization. These features are truly essential to really make the SID (Sound Interface Device) sing. Keep reading Gregg Peele's continuing series on 64 sound and music.

In any case, we're always interested in getting the utmost from your computer. If you've done something interesting, send it in!

Next month, we'll look at more products, including a software simulation of 80 columns, and various aids for BASIC and machine language programmers. Keep those cards and letters coming from over the Horizon.

NEWS & PRODUCTS

Editor/Assembler And World Processor for Commodore 64

Elcomp has released an editor/assembler and a word processor for the 64.

The editor/assembler, *Macrofire*, includes an editor with 24 commands and an assembler that can translate 10K of source code in about five seconds. The $89 program, which is available on tape or disk, has full macro capability and includes a function that allows you to assemble files larger than available memory.

Elcomp's word processor, *Blitztext*, is also available on tape or disk for $89. It includes horizontal and vertical scrolling, the ability to handle text files up to four disks long, left and right margin justification, and formatted output to any device. The program can be used with almost any printer.

Other products available from Elcomp include: *Tricks for VICs*, a book of ready-to-run programs and tutorials for the VIC-20, $9.95; *MORE On the 64*, a collection of machine language subroutines for the Commodore 64, $9.95; and *The Great Book of GAMES, Vol. 1*, an explanation of game programming on the Commodore 64, $9.95.

Elcomp Publishing, Inc.
53 Redrock Lane
Pomona, CA 91766
(714) 623-8314

Data Base, Terminal Program, And Adventure Game

Arfon Micro has released a handful of new software products for the VIC-20 and Commodore 64.

MicroBase is a data base and mailing label program for the VIC and 64. It allows up to 12 fields per record, 80 characters per field, and 176 characters per record. Sorting can be done by any field. The program is available for $29.95 on tape and $34.95 on disk.

Microterm 64 is a terminal communications program for the 64 that allows file transfer under ASCII and Commodore protocols. It includes the ability to download incoming files to disk, tape, or printer, and upload and transmit files from tape or disk. *Microterm 64* is available on tape or disk for $39.95 and $44.95, respectively.

Baldor's Castle is a realtime graphic adventure game for the VIC. The castle contains more than 70 rooms on three levels. Nine types of monsters patrol the castle. *Baldor's Castle* is available on cartridge for $49.95. A Commodore 64 version of the game is planned. It will feature more than 250 rooms on 10 levels and the option of designing your own castle.

Arfon Microelectronics, Inc.
111 Rena Drive, Suite C
Lafayette, LA 70503
(318) 988-2478

Computer Math For Kids

Integrity Software has introduced a line of programs designed to teach math concepts to young students.

Climb Time teaches addition and subtraction to preschool and kindergarten children through animated visual display. The program is available for the VIC-20 and Commodore 64 for $12.50.

Math Bash is an addition and subtraction drill game for first- and second-graders. The game, available for $12.50 for the unexpanded VIC, has four skill levels.

Math Bash II, designed for third- and fourth-graders, includes drills on multiplication and division. Two skill levels are available for each drill. The VIC program sells for $12.50.

Chopper for the 64 includes addition, subtraction, multiplication, and division in one package. Correct answers to math questions free the choppers from behind a wall. *Chopper* includes 12 play levels, sound, and sprite graphics. The program, designed for first through fourth grades, sells for $22.50.

Integrity Software
Box 29
Bristol, VT 05443

8-Slot VIC Expander

Compuscope is producing Supermother, which adds 35K of memory to the VIC-20 and provides eight expansion slots.

Supermother includes a system RESET button, a pause

Compuscope's Supermother provides eight expansion slots for the VIC.

button that freezes a program or game in progress, and a replaceable fuse. The board includes a write-protection switch, which, when used with *Blocksave* software (included with the board), allows you to make backup copies of cartridge programs on tape or disk.

The board, which is compatible with most VIC expansion modules, sells for $129.95.

Compuscope
6400 Signal St.
Tillamook, OR 97141

VIC-20 Adventure

MicRo Information Systems has produced an adventure game called *Zorlok* for the VIC-20.

In the game, you become the great-great-grandson of Zorlok the Wizard. You must enter his castle, wipe out a plague of monsters, and regain his treasures.

The tape version of the program requires 8K expansion and sells for $39.95; the disk version requires 16K expansion and sells for $45.95.

MicRo Information Systems
P.O. Box 73
Wayne, NJ 07470
(201) 696-3296

Intelligent Multiple Interface

Interpod, an intelligent 6502-based multiple interface,

The Interpod, an intelligent freestanding interface for the VIC and 64, uses a 6502 to communicate with all Commodore and RS-232 devices.

gives serial and parallel capabilities to the VIC and 64.

The interface lets the computer communicate with all Commodore serial.IEEE and IEEE-488 devices, as well as RS-232 devices. It is a standalone unit that uses no computer memory.

Up to 30 IEEE devices and any RS-232 modem or printer can be accessed by the system when the Interpod is plugged into the serial port. Through the Interpod, the computer can communicate with voltmeters, plotters, and other peripherals.

The interface is available for $180.

Limbic Systems, Inc.
1056 Elwell Court
Palo Alto, CA 94303
(415) 964-8788

COMPUTE!'s Gazette *for Commodore welcomes announcements of new products for VIC-20 and Commodore 64 computers, especially products aimed at beginning to intermediate users. Please send press releases and photos well in advance to: Tony Roberts, Assistant Managing Editor,* COMPUTE!'s Gazette, *P.O. Box 5406, Greensboro, NC 27403.*

New product releases are selected from submissions for reasons of timeliness, available space, and general interest to our readers. We regret that we are unable to select all new product submissions for publication. Readers should be aware that we present here some edited version of material submitted by vendors and are unable to vouch for its accuracy at time of publication.

Bug-Swatter:
Modifications And Corrections

• IMPORTANT: We added a POKE to the "Automatic Proofreader" (October 1983) to protect it from being destroyed when you LOAD another program from tape. The POKE does protect the Proofreader, and the Proofreader itself is not affected. However, a quirk in the VIC-20's operating system means that programs typed in with the Proofreader and SAVEd to tape cannot be LOADed properly later. If you LOAD a program SAVEd while you had the Proofreader in memory, you will get a ?LOAD ERROR message. This applies only to VIC-20 tape SAVEs (disk SAVEs work OK, and the quirk was fixed in the Commodore 64). The solution is to use this special LOAD procedure:

1. Turn the power off, then on again.
2. LOAD the program from tape (disregard the ?LOAD ERROR).
3. Type: POKE 45,PEEK(174):POKE 46,PEEK (175):CLR
4. Press RETURN.
5. ReSAVE the program to tape.

The program will LOAD just fine in the future. This month, the Proofreader has been updated to prevent this problem. The Proofreader also has been improved in other ways. We strongly recommend that you type in the new version of the Proofreader and discard the old one (the new version works on either the VIC or 64 and checks itself for typing errors). We apologize for any inconvenience this may have caused you.

• Many readers ran into recurring errors when attempting to run the "Single Drive File Copy" program that accompanied September's "64 Explorer" column. The errors involved numerous GOTOs and GOSUBs without corresponding target lines.

Here's what happened: just before the working version of the program was listed on our printer, 28 seemingly needless REM statements were deleted to save space. Usually, deleting a REM statement does not affect program operation. But in this case the REMs were being used to separate blocks of code to make the listing easier to read, and they were targets of the GOTOs and GOSUBs. The program will work if all GOTOs and GOSUBs without target lines are renumbered to branch to the next higher line number.

Since this solution would require lots of editing, and some references might be missed, we've decided to list all the deleted line numbers. The program can be fixed by adding these lines, and by changing the GOTO statement in line 1090 from GOTO 5000 to GOTO 4000.

(By the way, this is a good argument for adhering to the principle of never making a REM statement the target of a branch instruction. Many people, when entering listings, routinely omit REMs to save typing and memory.)

Add these lines: 10 REM, 30 REM, 100 REM, 120 REM, 200 REM, 220 REM, 300 REM, 320 REM, 400 REM, 420 REM, 500 REM, 520 REM, 600 REM, 620 REM, 700 REM, 720 REM, 800 REM, 820 REM, 900 REM, 920 REM, 1000 REM, 1020 REM, 2000 REM, 2020 REM, 3000 REM, 3020 REM, 4000 REM, 4020 REM.

• "TeleTerm 64" (September) works as published. But if you are having problems with uppercase letters appearing as lowercase or vice versa, try this modification: delete line 124 and reenter the statement as line 95.

• In the VIC-20 version of "The Viper" (August), omit the colon between THEN and PRINT in line 570.

• The author of "VIC/64 Mailing List" (August) suggests these changes to correct an infrequent bug in the Examine and Change options:

```
47 A$="":INPUT"{CLR}WHICH ITEM";A$:A=VAL(
   A$):IFA$=""ORA<1THEN19
48 READA$:IFA$="END"THEN19
49 IFA$<>"XX"THEN48
50 READA$:IFA<>VAL(A$)THEN48
```

If you are using "VIC/64 Mailing List" with adhesive mailing labels spaced one inch apart, the author suggests these changes for proper spacing:

```
95 N2$=LEFT$(B$,X):N1$=RIGHT$(B$,LEN(B$)-
   X)
101 READA:GOSUB92:GOSUB102:GOTO99
104 PRINT#1,CHR$(10)CHR$(10):RETURN
114 IFZ=4THEN120
116 PRINT#1,CHR$(10)CHR$(10)"ITEM";A:GOSU
    B92
120 GOSUB92:GOSUB102:CLOSE1:GOTO60
```

Also, in case the instructions for modifying the program for tape were unclear, here is the line to change:

```
55 PRINT"{CLR}":SAVE R$,1:END
```

We appreciate receiving both corrections and modifications from readers. Please address them to:

Bug-Swatter
c/o COMPUTE!'s Gazette
P.O. Box 5406
Greensboro, NC 27403

How To Type In *COMPUTE!'s Gazette* Programs

Many of the programs which are listed in *COMPUTE!'s Gazette* contain special control characters (cursor control, color keys, inverse video, etc.). To make it easy to know exactly what to type when entering one of these programs into your computer, we have established the following listing conventions.

Generally, any VIC-20 or Commodore 64 program listings will contain bracketed words which spell out any special characters: {DOWN} would mean to press the cursor down key. {5 SPACES} would mean to press the space bar five times.

To indicate that a key should be *shifted* (hold down the SHIFT key while pressing the other key), the key would be underlined in our listings. For example, S would mean to type the S key while holding the shift key. This would appear on your screen as a "heart" symbol. If you find an underlined key enclosed in braces (e.g., {10 N}), you should type the key as many times as indicated (in our example, you would enter ten shifted N's).

If a key is enclosed in special brackets, ⟦x⟧, you should hold down the *Commodore key* while pressing the key inside the special brackets. (The Commodore key is the key in the lower left corner of the keyboard.) Again, if the key is preceded by a number, you should press the key as many times as necessary.

Rarely, you'll see a solitary letter of the alphabet enclosed in braces. These characters can be entered on the Commodore 64 by holding down the CTRL key while typing the letter in the braces. For example, {A} would indicate that you should press CTRL-A. You should never have to enter such a character on the VIC-20, but if you do, you would have to leave the quote mode (press RETURN and cursor back up to the position where the control character should go), press CTRL-9 (RVS ON), the letter in braces, and then CTRL-0 (RVS OFF).

About the *quote mode*: you know that you can move the cursor around the screen with the CRSR keys. Sometimes a programmer will want to move the cursor under program control. That's why you see all the {LEFT}'s, {HOME}'s, and {BLU}'s in our programs. The only way the computer can tell the difference between direct and programmed cursor control is the quote mode.

Once you press the quote (the double quote, SHIFT-2), you are in the quote mode. If you type something and then try to change it by moving the cursor left, you'll only get a bunch of reverse-video lines. These are the symbols for cursor left. The only editing key that isn't programmable is the DEL key; you can still use DEL to back up and edit the line. Once you type another quote, you are out of quote mode.

You also go into quote mode when you INSerT spaces into a line. In any case, the easiest way to get out of quote mode is to just press RETURN. You'll then be out of quote mode and you can cursor up to the mistyped line and fix it.

Use the following table when entering cursor and color control keys:

When You Read:	Press:	See:	When You Read:	Press:	See:	When You Read:	Press:	See:
{CLEAR}	SHIFT CLR/HOME		{CYN}	CTRL 4		⟦7⟧	C= 7	
{HOME}	CLR/HOME		{PUR}	CTRL 5		⟦8⟧	C= 8	
{UP}	SHIFT ↑CRSR↓		{GRN}	CTRL 6		{F1}	f1	
{DOWN}	↑CRSR↓		{BLU}	CTRL 7		{F2}	f2	
{LEFT}	SHIFT ←CRSR→		{YEL}	CTRL 8		{F3}	f3	
{RIGHT}	←CRSR→		⟦1⟧	C= 1		{F4}	f4	
{RVS}	CTRL 9		⟦2⟧	C= 2		{F5}	f5	
{OFF}	CTRL Ø		⟦3⟧	C= 3		{F6}	f6	
{BLK}	CTRL 1		⟦4⟧	C= 4		{F7}	f7	
{WHT}	CTRL 2		⟦5⟧	C= 5		{F8}	f8	
{RED}	CTRL 3		⟦6⟧	C= 6				

A Beginner's Guide To Typing In Programs

What Is A Program?

A computer cannot perform any task by itself. Like a car without gas, a computer has *potential*, but without a program, it isn't going anywhere. Most of the programs published in *COMPUTE!'s Gazette* for Commodore are written in a computer language called BASIC. BASIC is easy to learn and is built into all VIC-20s and Commodore 64s.

BASIC Programs

Each month, *COMPUTE!'s Gazette* for Commodore publishes programs for both the VIC and 64. To start out, type in only programs written for your machine, e.g., "VIC Version" if you have a VIC-20. Later, when you gain experience with your computer's BASIC, you can try typing in and converting certain programs from another computer to yours.

Computers can be picky. Unlike the English language, which is full of ambiguities, BASIC usually has only one "right way" of stating something. Every letter, character, or number is significant. A common mistake is substituting a letter such as "O" for the numeral "0", a lowercase "l" for the numeral "1", or an uppercase "B" for the numeral "8". Also, you must enter all punctuation such as colons and commas just as they appear in the magazine. Spacing can be important. To be safe, type in the listings *exactly* as they appear.

Brackets And Special Characters

The exception to this typing rule is when you see the curved bracket, such as "{DOWN}". Anything within a set of brackets is a special character or characters that cannot easily be listed on a printer. When you come across such a special statement, refer to "How To Type In *COMPUTE!'s Gazette* Programs."

About DATA Statements

Some programs contain a section or sections of DATA statements. These lines provide information needed by the program. Some DATA statements contain actual programs (called machine language); others contain graphics codes. These lines are especially sensitive to errors.

If a single number in any one DATA statement is mistyped, your machine could "lock up," or "crash." The keyboard and STOP key may seem "dead," and the screen may go blank. Don't panic – no damage is done. To regain control, you have to turn off your computer, then turn it back on. This will erase whatever program was in memory, *so always SAVE a copy of your program before you RUN it*. If your computer crashes, you can LOAD the program and look for your mistake.

Sometimes a mistyped DATA statement will cause an error message when the program is RUN. The error message may refer to the program line that READs the data. *The error is still in the DATA statements, though.*

Get To Know Your Machine

You should familiarize yourself with your computer before attempting to type in a program. Learn the statements you use to store and retrieve programs from tape or disk. You'll want to save a copy of your program, so that you won't have to type it in every time you want to use it. Learn to use your machine's editing functions. How do you change a line if you made a mistake? You can always retype the line, but you at least need to know how to backspace. Do you know how to enter inverse video, lowercase, and control characters? It's all explained in your computer's manuals.

A Quick Review

1) Type in the program a line at a time, in order. Press RETURN at the end of each line. Use backspace or the back arrow to correct mistakes.

2) Check the line you've typed against the line in the magazine. You can check the entire program again if you get an error when you RUN the program.

3) Make sure you've entered statements in brackets as the appropriate control key (see "How To Type *COMPUTE!'s Gazette* Programs" elsewhere in the magazine.)

We regret that we are not able to respond to individual inquiries about programs, products, or services appearing in COMPUTE!'s Gazette *for Commodore due to increasing publication activity. On those infrequent occasions when a published program contains a typo, the correction will appear in the magazine, usually within eight weeks. If you have specific questions about items or programs which you've seen in* COMPUTE!'s Gazette *for Commodore, please send them to Gazette Feedback, P.O. Box 5406, Greensboro, NC 27403.*

Chicken Little

(Article on page 64.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

Program 1: VIC Version, Instructions And Character Set

```
Ø IF PEEK(7344)=PEEK(7344+256ØØ)THEN4
                                     :rem 23
1 POKE52,28:POKE56,28:CLR:FORT=7168TO7679
  :POKET,PEEK(T+256ØØ):NEXT        :rem 62
4 PRINTCHR$(147)                   :rem 175
5 GOSUB1ØØØØ:GOSUB6ØØØ              :rem 38
3Ø PRINT:PRINT"PLEASE WAIT FOR THE
   {3 SPACES}GAME TO LOAD."        :rem 189
4Ø POKE198,5:POKE631,78:POKE632,69:POKE63
   3,87:POKE634,13:POKE635,131:END
                                   :rem 226
6ØØØ PRINTCHR$(147);SPC(4);CHR$(18);"CHIC
     KEN LITTLE";CHR$(146)          :rem 71
7ØØØ PRINT:PRINT"YOU PLAY THE PART OF
     {2 SPACES}CHICKEN LITTLE."     :rem 48
7ØØ1 PRINT                          :rem 85
7Ø1Ø PRINT"YOU CAN MOVE IN THE{3 SPACES}B
     OTTOM QUARTER OF THE SCREEN."
                                   :rem 116
7Ø11 PRINT                          :rem 86
7Ø2Ø PRINT"PIECES OF SKY (BLOWN{2 SPACES}
     RANDOMLY BY THE WIND){2 SPACES}FALL
     {SPACE}DOWN AT YOU."          :rem 221
7Ø3Ø PRINT:PRINT"TO SCORE POINTS, TOSS RO
     CKS AT THE FALLING"           :rem 2ØØ
7Ø31 PRINT"PIECES OF SKY WITH THE SPACEBA
     R."                           :rem 1Ø9
7Ø4Ø PRINT:PRINT"YOU CAN ALSO HIT NON- FA
     LLING PIECES."                 :rem 66
7Ø5Ø PRINT:PRINTCHR$(18);"HIT A KEY TO CO
     NTINUE";CHR$(146)             :rem 153
7Ø55 GETA$:IFA$=""THEN7Ø55         :rem 197
7Ø56 PRINTCHR$(147)                 :rem 77
7Ø6Ø PRINT"KEYS I,J,K AND M MOVE UP,LEFT,
     RIGHT AND DOWNRESPECTIVELY.":PRINT
                                   :rem 172
7Ø8Ø PRINT"YOUR SCORE AND THE{4 SPACES}NU
     MBER OF REMAINING{3 SPACES}CHICKENS
     {SPACE}ARE SHOWN AT THE TOP.":rem 93
7Ø9Ø PRINT:PRINT"YOU GET 5Ø POINTS FOR HI
     TTING FALLING SKY,"           :rem 165
71ØØ PRINT"AND 1Ø POINTS FOR{5 SPACES}OTH
     ERS."                         :rem 148
769Ø PRINT:PRINTCHR$(18);"HIT A KEY TO CO
     NTINUE";CHR$(146)             :rem 163
77ØØ GETA$:IFA$=""THEN77ØØ         :rem 191
77Ø1 PRINTCHR$(147)                 :rem 74
77Ø2 PRINT"EACH TIME YOU CLEAR{3 SPACES}T
     HE SCREEN AND ADVANCE TO THE NEXT LE
     VEL,"                          :rem 18
77Ø3 PRINT"THE PIECES OF SKY FALL FASTER.
     "                             :rem 218
77Ø4 PRINT:PRINTCHR$(18);           :rem 28
77Ø5 PRINT"IF MORE THAN EIGHT{4 SPACES}PI
     ECES OF SKY HIT THE GROUND, THE GAME
      IS{3 SPACES}OVER."             :rem 9
7711 PRINT:PRINT:PRINT:PRINT:PRINT
                                   :rem 121
772Ø FORT=1TO25:PRINTCHR$(145);SPC(4);CHR
     $(18);"GOOD LUCK!":FORP=1TO15Ø:NEXTP
                                   :rem 171
773Ø PRINTCHR$(145);SPC(4);"GOOD LUCK!":F
     ORP=1TO15Ø:NEXTP:NEXTT        :rem 15Ø
774Ø PRINTCHR$(147)                 :rem 77
918Ø RETURN                        :rem 178
1ØØØØ READA:IFA<ØTHENRETURN        :rem 211
1ØØ1Ø FORI=ATOA+7:READJ:POKEI,J:NEXT
                                    :rem 39
1ØØ2Ø GOTO 1ØØØØ                    :rem 29
1ØØ31 DATA 7168,48,44,1ØØ,131,129,195,36,
      24                            :rem 73
1ØØ41 DATA 7176,12,18,233,33,241,14,9,16
                                   :rem 171
1ØØ5Ø DATA 7184,Ø,Ø,28,34,36,24,Ø,Ø
                                   :rem 164
1ØØ6Ø DATA 72ØØ,36,227,1,Ø,Ø,195,65,82
                                    :rem 67
1ØØ7Ø DATA 7216,Ø,Ø,Ø,Ø,Ø,1,3,133 :rem 45
1ØØ8Ø DATA 7224,16,8,4,2,1,Ø,Ø,1   :rem 9
1ØØ9Ø DATA7248,121,1,1,2,252,68,13Ø,1
                                    :rem 61
1Ø1ØØ DATA 7264,1,1,6,4,1Ø,5,2,1   :rem 4
1Ø11Ø DATA 7272,192,48,8,12,9,58,228,8
                                    :rem 92
1Ø12Ø DATA 7296,41,42,6Ø,72,164,135,68,56
                                   :rem 237
1Ø13Ø DATA 73Ø4,252,2,1,1,1,253,3,1
                                   :rem 153
19999 DATA -1                      :rem 141
```

Program 2: VIC Version, Main Program

```
Ø QW=4:POKE7678,4:POKE7675,9      :rem 149
1 TY=3:POKE7677,Ø:DIMC(3Ø)        :rem 249
2 POKE7679,3:TY$=":3"             :rem 15Ø
3 POKE36869,255:PS=PEEK(7675):ZZ=3Ø72Ø
                                  :rem 2Ø2
4 PRINTCHR$(147)                  :rem 175
5 TY=PEEK(7679):TY$=CHR$(58)+CHR$(TY+48)
                                   :rem 58
6 QW=PEEK(7678)                   :rem 1Ø9
7 V=36878:S1=36874:S2=36875:S3=36876:S4=3
  6877                              :rem 2
11 POKE36879,8:X=11:Y=2Ø          :rem 1Ø5
12 PRINTCHR$(147)                 :rem 222
15 DEFFNA(Z)=INT(RND(1)*Z)         :rem 24
16 FORT=1TO3Ø:C(T)=FNA(352)+77Ø1:POKEC(T)
   ,Ø:POKEC(T)+ZZ,6:NEXT          :rem 231
17 X=11:Y=2Ø                      :rem 145
18 IFQW=1THENM=1:GOTO2Ø           :rem 2Ø5
19 FORM=1TOQW                      :rem 87
2Ø A=((Y*22)+X)+768Ø              :rem 236
21 PRINTCHR$(159);CHR$(19);"SCORE:"SC
                                  :rem 163
22 PRINTCHR$(19);CHR$(156);SPC(12);"CHICK
   ENS:";CHR$(157);TY$            :rem 226
3Ø POKEA,1:POKEA+1,6:POKEA+22,7:POKEA+23,
   1Ø:POKEA+ZZ,7:POKEA+ZZ+1,7:POKEA+ZZ+22
   ,7                             :rem 17Ø
31 POKEA+ZZ+23,7                  :rem 166
4Ø IFPEEK(197)<>64THENGOSUB9ØØØ   :rem 1Ø2
5Ø IFF=1THENGOSUB9Ø4Ø              :rem 36
55 IFM<>1THEN65                   :rem 139
```

```
56 IFR1=1THENGOSUB217Ø                  :rem 100
57 IFR=31THEN7ØØØ                       :rem 228
6Ø IFR1=ØTHENGOSUB216Ø                  :rem 93
65 IFQW=1THENGOTO18                     :rem 225
2ØØØ NEXTM:GOTO18                       :rem 42
216Ø R=R+1:IF R=31THEN7ØØØ              :rem 184
2162 R1=1:IFC(R)=ØORPEEK(C(R))=32THEN216Ø
                                        :rem 111
2165 Q=C(R):QQ=6                        :rem 145
2166 IFQ1<>ØTHENQ1=32:QQ=Ø              :rem 38
217Ø POKEQ,Q1:POKEQ+ZZ,QQ:Q=Q+22+FNA(3)-1
     :IFQ=AORQ=A+1ORQ=A+22ORQ=A+23THENGOT
     O31ØØ                              :rem 12Ø
2171 Q1=PEEK(Q)                         :rem 81
218Ø POKEQ,Ø:POKEQ+ZZ,6                 :rem 162
219Ø IFQ>8185THENR1=Ø:POKEQ,32:POKEQ+ZZ,Ø
     :C(R)=Ø:DE=DE+1                    :rem 145
2195 IFQ=BTHENPOKEQ,4:POKEQ+ZZ,2        :rem 54
2196 IFDE=PSTHEN92ØØ                    :rem 196
22ØØ IFQ=BTHENFORT=17ØTO255:POKEV,15:POKE
     S4,T:POKES2,T:POKES1,T:POKES3,T:NEXT
     :POKEQ,32                          :rem 112
22Ø1 IFQ=BTHENPOKEQ+ZZ,Ø                :rem 14
22Ø2 POKEV,Ø:POKES4,Ø:POKES1,Ø:POKES3,Ø:P
     OKES2,Ø                            :rem 2Ø9
221Ø IFQ=BTHENSC=SC+5Ø:C(R)=Ø:R1=Ø:F=Ø
                                        :rem 24Ø
2222 RETURN                             :rem 168
31ØØ POKEA,12:POKEA+1,13:POKEA+22,16:POKE
     A+23,17:POKEA+ZZ,7:POKEA+ZZ+1,7:POKE
     A+ZZ+22,7                          :rem 162
31Ø1 POKEA+ZZ+23,7                      :rem 7
31Ø2 POKEV,15:POKES2,158:FORT=1TO6ØØ:NEXT
     :POKES2,Ø:FORT=1TO6Ø:NEXT          :rem 217
31Ø3 POKEV,15:POKES2,158:FORT=1TO3ØØ:NEXT
     :POKES2,Ø:FORT=1TO6Ø:NEXT          :rem 215
31Ø4 POKEV,15:POKES2,181:FORT=1TO1Ø5Ø:NEX
     T:POKES2,Ø:FORT=1TO6Ø:NEXT         :rem 7
31Ø5 POKEV,Ø:POKES1,Ø:POKES2,Ø          :rem 61
32ØØ TY=TY-1:IFTY=ØTHEN92ØØ             :rem 151
32Ø1 TY$=CHR$(58)+CHR$(TY+48)           :rem 84
32Ø3 R1=Ø                               :rem 184
321Ø IFTY<>ØTHENF=Ø:POKEB,32:POKEA,32:POK
     EA+1,32:POKEA+22,32:POKEA+23,32
                                        :rem 49
322Ø IFTY<>ØTHENPOKEB+ZZ,Ø:POKEA+ZZ,Ø:POK
     EA+1+ZZ,Ø:POKEA+22+ZZ,Ø:POKEA+23+ZZ,
     Ø:GOTO17                           :rem 114
7ØØØ PRINTCHR$(147):PRINT:PRINT:PRINT:PRI
     NT:PRINT:PRINT:PRINT               :rem 179
7ØØ1 POKE36869,24Ø:FORT=1TO6:POKEV,15:POK
     ES2,239                            :rem 142
7ØØ2 PRINTCHR$(5);CHR$(145);SPC(4);CHR$(1
     8);                                :rem 219
7ØØ3 PRINT"NEXT LEVEL":FORP=1TO15Ø:NEXT:P
     OKES2,Ø                            :rem 45
7ØØ5 PRINTCHR$(145);SPC(4);"NEXT LEVEL":F
     ORP=1TO15Ø:NEXTP:NEXTT             :rem 2Ø7
7ØØ6 POKE36869,255:PRINTCHR$(5)         :rem 34
7ØØ7 IFQW=1THENQW=2                     :rem 185
7Ø1Ø R=Ø:POKE7679,TY:QW=QW-1:POKE7678,QW
                                        :rem 228
7Ø11 IFPS=1THEN7Ø2Ø                     :rem 97
7Ø12 PS=PS-1                            :rem 171
7Ø2Ø R=Ø:DE=Ø:GOTO3                     :rem 94
9ØØØ J=PEEK(197):POKEA,32:POKEA+1,32:POKE
     A+22,32:POKEA+23,32                :rem 2Ø6
9ØØ1 POKEA+ZZ,Ø:POKEA+ZZ+1,Ø:POKEA+ZZ+22,
     Ø:POKEA+23+ZZ,Ø                    :rem 159
9ØØ3 IFF=1THEN9Ø1Ø                      :rem 8
9ØØ5 IFJ=32THENF=1:B=A-22:FORT=1TO2Ø:POKE
     V,15:POKES4,18Ø:NEXT:POKES4,Ø
                                        :rem 1Ø6
9Ø1Ø IFJ=44THENX=X+1                    :rem 192
9Ø11 IFJ=2ØTHENX=X-1                    :rem 189
9Ø15 IFX>21THENX=2Ø                     :rem 125
9Ø16 IFX<ØTHENX=Ø                       :rem 23
9Ø2Ø IFJ=36THENY=Y+1                    :rem 196
9Ø21 IFJ=12THENY=Y-1                    :rem 193
9Ø25 IFY>21THENY=21                     :rem 129
9Ø26 IFY<17THENY=17                     :rem 138
9Ø3Ø RETURN                             :rem 172
9Ø4Ø POKEB,2:POKEB+ZZ,4                 :rem 134
9Ø5Ø POKEB,32:POKEB+ZZ,Ø:B=B-22         :rem 66
9Ø7Ø IFB<768ØTHENF=Ø                    :rem 148
9Ø75 IFB=QTHEN2195                      :rem 52
9Ø8Ø IFPEEK(B)<>ØTHEN9Ø9Ø               :rem 195
9Ø85 POKEB,4:FORT=225TO17ØSTEP-1:POKEV,15
     :POKES4,T:POKES2,T:NEXT:SC=SC+1Ø
                                        :rem 158
9Ø9Ø POKES2,Ø:POKES4,Ø:POKEV,Ø          :rem 73
9159 IFPEEK(B)<>32THENPOKEB,32:POKEB+ZZ,Ø
     :F=Ø:GOTO9166                      :rem 75
9165 POKEB,2:POKEB+ZZ,4                 :rem 142
9166 REM                                :rem 186
918Ø RETURN                             :rem 178
92ØØ POKE36869,24Ø:PRINTCHR$(147);CHR$(5)
     ;"THE SKY HAS FALLEN."             :rem 86
921Ø POKE198,Ø                          :rem 249
```

Program 3: 64 Version

```
1ØØ REM CHICKEN                         :rem 1Ø6
11Ø :                                   :rem 2Ø4
12Ø PRINT"{CLR}":POKE5328Ø,Ø:POKE53281,Ø:
    FORI=ØTO27:POKE54272+I,Ø:NEXT       :rem 31
13Ø POKE54277,16:POKE54278,242:POKE54296,
    15                                  :rem 155
136 POKE54284,16:POKE54285,242:POKE54291,
    16:POKE54285,242                    :rem 2Ø6
14Ø POKE214,11:PRINT:PRINTSPC(13)"{6}CH
    ICKEN LITTLE"                       :rem 178
15Ø POKE56,48:CLR:FORI=ØTO62:READA:POKE49
    152+I,A:NEXT:SYS49152               :rem 98
16Ø GOSUB141Ø:GOSUB112Ø                 :rem 91
17Ø POKE9Ø5,9:POKE9Ø7,Ø:POKE9Ø8,4:POKE9Ø9
    ,3                                  :rem 3
18Ø TY=3:QW=4:DIMC(3Ø)                  :rem 2Ø
19Ø POKE53272,28:PS=PEEK(9Ø5):ZZ=54272
                                        :rem 191
2ØØ PRINT"{CLR}"                        :rem 246
21Ø TY=PEEK(9Ø9)                        :rem 149
22Ø QW=PEEK(9Ø8)                        :rem 144
23Ø S1=54272:S2=54279:S4=54286:XL=53248:X
    H=53264:YL=53249                    :rem 199
24Ø POKE5328Ø,Ø:POKE53281,Ø             :rem 235
25Ø DEFFNA(Z)=INT(RND(1)*Z)             :rem 73
26Ø FORT=1TO3Ø:C(T)=FNA(64Ø)+1Ø63:POKEC(T
    ),Ø:POKEC(T)+ZZ,6:NEXT              :rem 19
27Ø X=2Ø:Y=2Ø:POKE53269,1:POKE2Ø4Ø,11:POK
    E53287,7?                           :rem 35
28Ø :                                   :rem 212
29Ø REM MAIN LOOP                       :rem 222
3ØØ IFQW=1THENM=1:GOTO33Ø               :rem 43
31Ø FORM=1TOQW                          :rem 129
33Ø PRINT"{HOME}{CYN}SCORE:"SC;TAB(27)"
    {PUR}CHICKENS:"TY                   :rem 64
335 IFPEEK(5632Ø)<>127THENGOSUB87Ø      :rem 2
35Ø XP=X*8+24:POKEXL,XPAND255:POKEXH,XP/2
    56:POKEYL,Y*8+5Ø                    :rem 7Ø
355 A=INT(X)+INT(Y)*4Ø+1Ø24             :rem 239
```

```
390 IFF=1THENGOSUB98Ø                  :rem 47
400 IFM<>1THEN45Ø                     :rem 227
41Ø IFR1=1THENGOSUB52Ø                 :rem 91
42Ø IFR=31THEN75Ø                     :rem 227
43Ø IFR1=ØTHENGOSUB48Ø                 :rem 97
45Ø IFQW=1THEN3ØØ
46Ø NEXT:GOTO3ØØ                      :rem 223
47Ø :                                 :rem 213
48Ø R=R+1:IFR=31THEN75Ø                :rem 96
49Ø R1=1:IFC(R)=ØORPEEK(C(R))=32THEN48Ø
                                       :rem 2Ø
5ØØ Q=C(R):QQ=6                        :rem 88
51Ø IFQ1<>ØTHENQ1=32:QQ=Ø             :rem 237
52Ø POKEQ,Q1:POKEQ+ZZ,QQ:Q=Q+4Ø+FNA(3)-1:
    IFQ=AORQ=A+1ORQ=A+4ØORQ=A+41THEN64Ø
                                      :rem 226
53Ø Q1=PEEK(Q)                         :rem 3Ø
54Ø POKEQ,Ø:POKEQ+ZZ,6                :rem 112
55Ø IFQ>1864THENR1=Ø:POKEQ,32:C(R)=Ø:DE=D
    E+1                               :rem 1Ø3
56Ø IFQ=BTHENPOKEQ,4:POKEQ+ZZ,2
57Ø IFDE=PSTHEN1Ø9Ø                   :rem 141
58Ø IFQ<>BTHEN61Ø                     :rem 255
585 POKES4+4,129:POKES1+4,17:POKES2+4,33:
    FORT=8TO9Ø                         :rem 52
586 POKES4+1,T:POKES2+1,T*2:POKES1+1,T:NE
    XT:POKEQ,32                       :rem 232
59Ø POKEQ+ZZ,Ø                         :rem 89
6ØØ SC=SC+5Ø:C(R)=Ø:R1=Ø:F=Ø           :rem 51
61Ø POKES4+4,128:POKES1+4,16:POKES2+4,32
                                       :rem 48
62Ø RETURN                            :rem 12Ø
63Ø :                                 :rem 211
64Ø FORI=ØTO15:POKE53287,(I+7)AND15:FORT=
    1TO1ØØ:NEXT:NEXT                   :rem 28
66Ø POKES2+4,17:POKES2+1,21:FORT=1TO6ØØ:N
    EXT:POKES2+4,16:FORT=1TO6Ø:NEXT
                                      :rem 246
67Ø POKES2+4,17:POKES2+1,21:FORT=1TO3ØØ:N
    EXT:POKES2+4,16:FORT=1TO6Ø:NEXT
                                      :rem 244
68Ø POKES2+4,17:POKES2+1,25:FORT=1TO1Ø5Ø:
    NEXT:POKES2+4,16:FORT=1TO6Ø:NEXT
                                       :rem 44
7ØØ TY=TY-1:IFTY=ØTHEN1Ø9Ø            :rem 1Ø4
71Ø R1=Ø                              :rem 136
725 F=Ø:POKEB,32                      :rem 141
74Ø GOTO27Ø                           :rem 1Ø9
744 :                                 :rem 217
75Ø PRINT"{CLR}{7 DOWN}"              :rem 119
76Ø POKE53272,21:FORT=1TO6:POKES2+1,31:PO
    KES2+4,17                         :rem 213
77Ø PRINTSPC(15)"{WHT}{UP}{RVS}NEXT LEVEL
    ":FORP=1TO15Ø:NEXT:POKES2+4,16
                                      :rem 22Ø
78Ø PRINTSPC(15)"{UP}NEXT LEVEL":FORP=1TO
    15Ø:NEXT:NEXT                      :rem 95
79Ø PRINT"{CLR}":POKE53272,28           :rem 6
8ØØ IFQW=1THENQW=2                    :rem 131
81Ø R=Ø:POKE9Ø9,TY:QW=QW-1:POKE9Ø8,QW
                                       :rem 63
82Ø IFPS=1THEN84Ø                       :rem 5
83Ø PS=PS-1                           :rem 124
84Ø R=Ø:DE=Ø:GOTO19Ø                  :rem 152
85Ø :                                 :rem 215
86Ø REM MOVE PLAYER                   :rem 134
87Ø J=PEEK(5632Ø)                     :rem 156
89Ø IFF=1THEN91Ø                      :rem 173
9ØØ IF(JAND16)THEN91Ø                 :rem 198
9Ø5 F=1:B=A-4Ø:POKES4+4,129:POKES4+1,2Ø:F
    ORT=1TO2Ø:NEXT:POKES4+4,128        :rem 69
```

```
91Ø IF(JAND8)=ØTHENX=X+1:IFX>39THENX=39
                                      :rem 175
92Ø IF(JAND4)=ØTHENX=X-1:IFX<ØTHENX=Ø
                                       :rem 52
93Ø IF(JAND2)=ØTHENY=Y+1:IFY>21THENY=21
                                      :rem 157
94Ø IF(JAND1)=ØTHENY=Y-1:IFY<17THENY=17
                                      :rem 167
95Ø RETURN                            :rem 126
96Ø :                                 :rem 217
97Ø REM MOVE ROCK                     :rem 234
98Ø POKEB,2:POKEB+ZZ,4                 :rem 9Ø
99Ø POKEB,32:B=B-4Ø                    :rem 48
1ØØØ IFB<1Ø24THENF=Ø                  :rem 119
1Ø1Ø IFB=QTHEN56Ø                     :rem 235
1Ø2Ø IFPEEK(B)THEN1Ø4Ø                :rem 254
1Ø3Ø POKEB,4:POKES2+4,17:POKES4+4,129
                                      :rem 141
1Ø35 POKEB,4:FORT=33TO1ØSTEP-1:POKES4+1,T
     :POKES2+1,T:NEXT:SC=SC+1Ø        :rem 142
1Ø4Ø POKES2+4,16:POKES4+4,128         :rem 129
1Ø5Ø IFPEEK(B)<>32THENPOKEB,32:F=Ø:RETURN
                                       :rem 36
1Ø6Ø POKEB,2:POKEB+ZZ,4               :rem 128
1Ø7Ø RETURN                           :rem 168
1Ø8Ø :                                  :rem 3
1Ø9Ø PRINT"{CLR}":POKE53272,21:PRINT"
     {WHT}THE SKY HAS FALLEN."        :rem 2Ø5
1Ø95 PRINT"{DOWN}YOUR SCORE WAS:"SC
                                       :rem 55
11ØØ POKE53269,Ø:END                  :rem 1Ø3
111Ø :                                :rem 253
112Ø PRINT"{CLR}{3 DOWN}"SPC(13)"[6]CHI
     CKEN LITTLE"                     :rem 15Ø
113Ø PRINT"{GRN}{DOWN} YOU ARE CHICKEN LI
     TTLE.{2 SPACES}YOU MOVE IN"       :rem 86
114Ø PRINT"{DOWN} THE BOTTOM PART OF THE
     {SPACE}SCREEN WHILE"              :rem 68
115Ø PRINT"{DOWN} PIECES OF SKY (BLOWN RA
     NDOMLY BY THE"                   :rem 122
116Ø PRINT"{DOWN} WIND) FALL TO THE GROUN
     D.{2 SPACES}TO SCORE"            :rem 196
117Ø PRINT"{DOWN} POINTS, TOSS ROCKS AT T
     HE FALLING"                      :rem 242
118Ø PRINT"{DOWN} PIECES OF SKY WITH THE
     {SPACE}FIRE BUTTON."              :rem 62
119Ø PRINT"{2 DOWN} YOU CAN ALSO HIT NON-
     FALLING PIECES."                 :rem 157
12ØØ PRINTSPC(9)"[6]{2 DOWN}PRESS FIRE
     {SPACE}TO CONTINUE":GOSUB132Ø
                                      :rem 25Ø
121Ø :                                :rem 254
122Ø PRINT"{CLR}{GRN}{DOWN} USE A JOYSTIC
     K IN PORT TWO TO MOVE":PRINT"{DOWN}
     {SPACE}ABOUT THE SCREEN."         :rem 12
123Ø PRINT"{DOWN} YOU GET 5Ø POINTS FOR H
     ITTING FALLING"                  :rem 194
124Ø PRINT"{DOWN} SKY, AND 1Ø POINTS FOR
     {SPACE}OTHERS."                  :rem 199
125Ø PRINT"{DOWN} EACH TIME YOU CLEAR THE
      SCREEN AND"                     :rem 194
126Ø PRINT"{DOWN} ADVANCE TO THE NEXT LEV
     EL THE PIECES":PRINT"{DOWN} OF SKY F
     ALL FASTER."                      :rem 44
127Ø PRINT"{DOWN} IF MORE THAN EIGHT PIEC
     ES OF SKY HIT"                    :rem 52
128Ø PRINT"{DOWN} THE GROUND, THE GAME IS
      OVER."                          :rem 138
129Ø PRINTSPC(15)"{2 DOWN}[6]GOOD LUCK!
     ":PRINTSPC(1Ø)"{DOWN}PRESS FIRE TO B
     EGIN":GOSUB132Ø                   :rem 93
```

```
1300 PRINT"{CLR}":RETURN              :rem 66
1310 :                                :rem 255
1320 IF(PEEK(56320)AND16)=0THEN1320
                                      :rem 184
1330 IFPEEK(56320)AND16THEN1320       :rem 251
1340 RETURN                           :rem 168
1350 :                                :rem 3
1360 DATA 173,14,220,41,254,141,14,220,16
     5,1,41,251,133,1,169,208         :rem 103
1370 DATA 133,252,169,48,133,254,169,0,13
     3,251,169,0,133,253,160,0        :rem 174
1380 DATA 162,8,177,251,145,253,200,208,2
     49,230,254,230,252,202,208,242
                                      :rem 165
1390 DATA 165,1,9,4,133,1,173,14,220,9,1,
     141,14,220,96                    :rem 77
1400 :                                :rem 255
1410 READA:IFA<0THENFORI=0TO62:READA:POKE
     704+I,A:NEXT:RETURN              :rem 127
1420 FORI=12288+A*8TO12288+A*8+7:READJ:PO
     KEI,J:NEXT                       :rem 32
1430 GOTO1410                         :rem 199
1440 DATA 0,48,44,100,131,129,195,36,24
                                      :rem 119
1450 DATA 1,12,18,233,33,241,14,9,16
                                      :rem 219
1460 DATA 2,0,0,28,34,36,24,0,0       :rem 215
1470 DATA 4,36,227,1,0,0,195,65,82
                                      :rem 131
1480 DATA 6,0,0,0,0,0,1,3,133         :rem 104
1490 DATA 7,16,8,4,2,1,0,0,1          :rem 70
1500 DATA 10,121,1,1,2,252,68,130,1
                                      :rem 149
1510 DATA 12,1,1,6,4,10,5,2,1         :rem 105
1520 DATA 13,192,48,8,12,9,58,228,8
                                      :rem 195
1530 DATA 16,41,42,60,72,164,135,68,56
                                      :rem 81
1540 DATA 17,252,2,1,1,1,253,3,1,-1
                                      :rem 146
1541 DATA 0,0,0,30,0,0,107,0,0,209:rem 87
1542 DATA 128,0,97,128,0,25,128,2,12
                                      :rem 227
1543 DATA 255,141,12,0,113,12,31,1,12
                                      :rem 252
1544 DATA 97,131,12,131,6,12,4,12,6,0
                                      :rem 7
1545 DATA 56,3,3,224,1,254,0,1,152
                                      :rem 119
1546 DATA 0,1,152,0,1,152,0,1,152 :rem 53
1547 DATA 0,1,152,0,3,108,0           :rem 24
1550 END                              :rem 162
```

Martian Prisoner

(Article on page 68.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

Program 1: VIC/64 Martian Prisoner

```
10 GOSUB5000                          :rem 166
100 GOSUB1000:IFR=6ANDU=0THENR=1:GOTO100
                                      :rem 232
110 GOSUB2000:PRINT                   :rem 155
120 ONVGOSUB3000,3000,3000,3000,3100,3200
    ,3300,3400,3500,3600,3700,3800,3900
                                      :rem 42
125 IFV=14THEN3900                    :rem 26
130 GOTO100                           :rem 94
1000 PRINT:ONRGOSUB1100,1200,1300,1400,15
     00,1600,1700,1800                :rem 87
1005 IFI(4)=-1ANDI(6)=-1THENPRINTNS$
                                      :rem 125
1010 PRINT:PRINT"{CYN}OBJECTS:{WHT}";
                                      :rem 125
1020 FORL=1TO8:IFI(L)=RTHENPRINTTAB(8);N$
     (L)                              :rem 227
1030 NEXT:PRINT:RETURN                :rem 228
1100 PRINT"YOU ARE IN A PRISON{3 SPACES}C
     ELL."                            :rem 105
1110 IFRND(1)>.25THENRETURN           :rem 154
1120 G=1:PRINT"A GUARD HAS TURNED OFFTHE
     {SPACE}FORCE FIELD AND{3 SPACES}ENTE
     RED THE CELL."                   :rem 126
1130 C%(1,1)=2:RETURN                 :rem 149
1200 PRINT"YOU ARE IN A N/S HALL.":RETURN
                                      :rem 122
1300 PRINT"YOU ARE IN THE ENGINE ROOM.":R
     ETURN                            :rem 29
1400 PRINT"YOU ARE IN A SMALL{4 SPACES}RO
     OM. A LARGE SIGN IS ON THE WALL."
                                      :rem 124
1410 IFI(4)=-1THENPRINTNS$            :rem 63
1420 IFI(6)=-1THEN4500                :rem 185
1430 RETURN                           :rem 168
1500 PRINT"YOU ARE IN THE SUPPLY ROOM.":R
     ETURN                            :rem 86
1600 PRINT"YOU ARE IN THE NORTH{2 SPACES}
     SIDE OF THE HALL."               :rem 90
1610 IFU=1THENPRINT"THE GUARDS DON'T
     {6 SPACES}NOTICE YOU."           :rem 10
1620 IFU=0THENPRINT"THE GUARDS TAKE YOU
     {3 SPACES}BACK TO THE CELL.":G=0
                                      :rem 180
1630 RETURN                           :rem 170
1700 PRINT"YOU ARE IN A LARGE{4 SPACES}RO
     OM.":RETURN                      :rem 54
1800 PRINT"YOU ARE IN A STRANGE{2 SPACES}
     GARDEN WHERE FOOD IS{2 SPACES}GROWN
     {SPACE}FOR THE CREW."            :rem 255
1810 IFI(4)=-1THENPRINTNS$            :rem 67
1820 RETURN                           :rem 171
2000 C$="":N=0:V=0:PRINT:INPUT"COMMAND
     {GRN}";C$:PRINT"{WHT}":IFC$=""THEN20
     00                               :rem 226
2015 P=0:IFLEN(C$)<2THEN2050          :rem 73
2020 FORL=2TOLEN(C$)-1                :rem 254
2030 IFMID$(C$,L,1)=" "THENP=L        :rem 104
2040 NEXT                             :rem 5
2050 IFP=0THENV$=C$:N$=""             :rem 141
2060 IFP>0ANDP=LEN(C$)THENV$=C$:N$=""
                                      :rem 134
2070 IFP>0ANDP<LEN(C$)THENV$=LEFT$(C$,P-1
     ):N$=RIGHT$(C$,LEN(C$)-P)        :rem 86
2080 FORL=1TO14:IFLEFT$(V$,3)=V$(L)THENV=
     L                                :rem 23
2100 NEXT:FORL=1TO8:IFLEFT$(N$,3)=A$(L)TH
     ENN=L                            :rem 55
2120 NEXT:IFN>0ANDV>0THENRETURN       :rem 47
2130 IFN=0ANDV>0ANDN$=""THENRETURN
                                      :rem 124
2135 IFN=0ANDV=5THENRETURN            :rem 191
2140 PRINT:PRINT"I DON'T UNDERSTAND.":GOT
     O2000                            :rem 95
```

```
3ØØØ N$=V$:GOTO311Ø                    :rem 36
31ØØ N$=LEFT$(N$,1)                   :rem 226
311Ø IFR=1ANDN$="E"ANDG=ØTHENPRINT"THE FO
     RCE FIELD STOPS YOU.":RETURN :rem 56
312Ø IFR<>1ORN$<>"E"ORG=ØTHEN313Ø:rem 179
3125 PRINT"AS YOU LEAVE THE CELL THE FORC
     E FIELD IS{4 SPACES}ACTIVATED, TRAPP
     ING"                              :rem 17
3127 PRINT"THE GUARD."                 :rem 32
313Ø IFR=2ANDN$="E"ANDC%(2,1)=ØANDI(8)>-1
     THENPRINT"THE LOCKED DOOR STOPS YOU.
     "                                :rem 118
3135 IFR=2ANDN$="E"ANDC%(2,1)=ØANDI(8)>-1
     THENRETURN                       :rem 255
314Ø IFR<>2ORN$<>"E"ORC%(2,1)>ØTHEN315Ø
                                      :rem 186
3145 PRINT"YOU UNLOCK THE DOOR{3 SPACES}W
     ITH THE KEY.":C%(2,1)=5:N$(7)="OPEN
     {SPACE}DOOR"                     :rem 191
315Ø IFN$="N"THEND=Ø                  :rem 121
316Ø IFN$="E"THEND=1                  :rem 114
3165 IFN$="S"THEND=2                  :rem 134
317Ø IFN$="W"THEND=3                  :rem 135
3175 IFC%(R,D)=ØTHENPRINTCN$:RETURN
                                      :rem 21Ø
318Ø PRINT"OK":R=C%(R,D):RETURN  :rem 149
32ØØ IFN=1ORN=2ORN=3ORN=7THENPRINT"YOU CA
     N'T LIFT IT!":RETURN             :rem 129
32Ø3 IFI(N)<>RTHENPRINT"IT'S NOT HERE.":R
     ETURN                            :rem 127
32Ø5 IFN=5THEN372Ø                     :rem 2Ø
321Ø PRINT"OK":I(N)=-1:RETURN           :rem 8
33ØØ PRINT"OK":I(N)=R:RETURN          :rem 252
34ØØ PRINT"YOU ARE CARRYING:"           :rem 6
341Ø FORL=1TO8:IFI(L)=-1THENPRINTTAB(3);N
     $(L)                             :rem 239
342Ø NEXT:RETURN                       :rem 34
35ØØ IFN<>3ORR<>4THENPRINTCN$:RETURN
                                      :rem 126
351Ø PRINT"ATOMIC FUEL NEARBY.{3 SPACES}D
     ON'T BRING ANY RADIO-ACTIVE":rem 128
352Ø PRINT"MATERIALS INTO THIS{3 SPACES}R
     OOM.":RETURN                      :rem 52
36ØØ IFN<>7ORR<>2ORI(8)>-1THENPRINTCN$:RE
     TURN                             :rem 144
361Ø N$="E":GOTO3145                   :rem 66
37ØØ IFI(5)<>RTHENPRINTCN$:RETURN:rem 127
372Ø PRINT"YOU ARE NOW WEARING A UNIFORM.
     ":I(5)=-1:U=1:RETURN             :rem 189
38ØØ IFN<>6THENPRINTRI$:RETURN        :rem 237
381Ø PRINT"YOU QUICKLY BECOME{4 SPACES}SI
     CK AND DIE.":GOTO46ØØ            :rem 161
39ØØ IFN=2THENPRINT"THE GUARD SHOOTS YOU.
     ":GOTO46ØØ                       :rem 18Ø
391Ø PRINTCN$:RETURN                   :rem 41
45ØØ PRINT                             :rem 86
45Ø5 PRINT"THE RADIOACTIVE PLANT EMITS EN
     OUGH NEUTRONS TO START A"        :rem 98
451Ø PRINT"CHAIN REACTION. THE{3 SPACES}S
     HIP EXPLODES."                    :rem 4Ø
4515 PRINT"YOU ESCAPE IN A LIFE- CRAFT."
                                       :rem 17
452Ø PRINT:PRINT"{PUR}YOU WIN!":GOTO461Ø
                                       :rem 73
46ØØ PRINT:PRINT"{PUR}YOU LOSE!"       :rem 79
461Ø PRINT:PRINT:PRINT"{GRN}PLAY AGAIN?"
                                       :rem 29
462Ø GETK$:IFK$="Y"THENRUN             :rem 81
463Ø IFK$="N"THENEND                  :rem 16Ø
464Ø GOTO462Ø                         :rem 211
5ØØØ PRINT"{HOME}{CLR}":POKE36879,8:PRINT
     TAB(3)"{GRN}{RVS}MARTIAN PRISONER
     {OFF}":PRINT                     :rem 135
5Ø8Ø DIMV$(14),C%(8,3),I(8),N$(8),A$(8)
                                      :rem 146
5Ø9Ø R=1:FORL=1TO14:READV$(L):NEXT:rem 87
51ØØ FORL=1TO8:READC%(L,Ø),C%(L,1),C%(L,2
     ),C%(L,3):NEXT                    :rem 31
511Ø FORL=1TO8:READN$(L),A$(L),I(L):NEXT
                                       :rem 97
5115 CN$="YOU CAN'T":RI$="DON'T BE SILLY!
     "                                 :rem 87
512Ø NS$="GEIGER COUNTER IS{5 SPACES}CLIC
     KING.":RETURN                      :rem 9
6ØØØ DATAN,E,S,W,GO,GET,DRO,INV,REA,OPE,W
     EA,EAT,KIL,HIT                   :rem 217
6Ø1Ø DATAØ,Ø,Ø,Ø,6,Ø,3,Ø,2,4,Ø,Ø,Ø,Ø,Ø,3,
     Ø,Ø,Ø,2,7,Ø,2,Ø,Ø,8,6,Ø,Ø,Ø,Ø,7
                                      :rem 1Ø3
6Ø2Ø DATAFORCE FIELD,FOR,1,GUARDS,GUA,6,S
     IGN,SIG,4,GEIGER COUNTER,GEI,5,UNIFO
     RM                               :rem 251
6Ø3Ø DATA UNI,5,PLANT,PLA,8,LOCKED DOOR,D
     OO,2,MAGNETIC KEY,KEY,3          :rem 151
```

Program 2: 64 Formatter

```
1ØØ PRINT "{CLR}{4 SPACES}{RVS}22-COLUMN
    {SPACE}PRINT FORMATTER FOR 64":PRINT
                                      :rem 191
11Ø PRINT "READING DATA"              :rem 119
12Ø FORI=828TO881:READA:CK=CK+A:POKEI,A:N
    EXT:POKE179,883AND255              :rem 92
13Ø IF CK<>6Ø32 THEN PRINT "ERROR IN DATA
    :CHECK TYPING.":END               :rem 227
14Ø PRINT"{DOWN}BEFORE...":SYS 828:PRINT"
    AFTER..."                         :rem 15Ø
15Ø PRINT "{DOWN}PRESS RUN/STOP-RESTORE";
    :PRINT"TO REGAIN 4Ø COLUMNS" :rem 228
16Ø PRINT "{DOWN}ENTER {RVS}SYS 828{OFF}
    {SPACE}TO":PRINT"REACTIVATE, IF":PRIN
    T"NECESSARY."                     :rem 115
17Ø PRINT "{DOWN}DO NOT EDIT ANY":PRINT"L
    INES WHILE IN 22 COL-UMN MODE."
                                       :rem 84
1ØØØ DATA169,71,141,38,3,169,3,141
                                      :rem 18Ø
1Ø1Ø DATA39,3,96,72,152,72,138,72:rem 141
1Ø2Ø DATA56,32,24Ø,255,192,9,176,3
                                      :rem 185
1Ø3Ø DATA76,1ØØ,3,192,31,144,15,169
                                      :rem 226
1Ø4Ø DATA13,32,2Ø2,241,56,32,24Ø,255
                                        :rem 9
1Ø5Ø DATA16Ø,9,24,32,24Ø,255,1Ø4,17Ø
                                       :rem 14
1Ø6Ø DATA1Ø4,168,1Ø4,76,2Ø2,241    :rem 3Ø
```

Munchmath

(Article on page 76).

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

Program 1: Munchmath, VIC Version

```
10 PRINT"{CLR}":POKE36878,15:S=36874:SM=7
   702:CM=38422:L=1:BC=27           :rem 240
15 POKE52,29:POKE56,29:S$="{RVS}{HOME}
   {30 DOWN}":CB=36879               :rem 96
20 J$="{22 I}":P=3:GOTO305          :rem 66
25 FORT=1TO300:NEXTT:RETURN          :rem 45
30 FORT=1TO30:NEXTT:RETURN          :rem 249
35 FORT=1TO70:NEXTT:RETURN            :rem 2
40 D=VAL(AN$):IFASC(AN$)=81THEN280
                                    :rem 223
45 IFD=CTHENP=P+1:R=R+1:M=M+1:SC=SC+10:PO
   KES+2,220:FORT=1TO5:NEXTT:POKES+2,0:GO
   TO240                             :rem 22
50 M=M+1:W=W+1:PRINTLEFT$(S$,11)SPC(11-LE
   N(C$))"{RED}"C:POKES,200:GOSUB25:POKES
   ,0:GOTO250                        :rem 66
55 FORI=1TOLEN(A$):PRINTMID$(A$,I,1);:POK
   ES,250:GOSUB30:POKES,0:NEXTI:A$=""
                                     :rem 94
60 RETURN                            :rem 70
65 PR=PR+1:A=INT(RND(1)*(5*L))+1    :rem 36
70 B=INT(RND(1)*(5*L))+1:IFB>ATHENA=A+B
                                    :rem 239
75 E=A*B:A$=STR$(A):B$=STR$(B)      :rem 179
80 IFQ=1THENC=A+B:X=43:GOTO100       :rem 77
85 IFQ=2THENC=A-B:X=45:GOTO100       :rem 87
90 IFQ=3THENC=A:GOTO120             :rem 175
95 C=E:X=88                         :rem 114
100 C$=STR$(C):PRINTLEFT$(S$,8)SPC(11-LEN
    (A$))A                          :rem 196
105 PRINTLEFT$(S$,9)SPC(9-LEN(B$))" ";CHR
    $(X)B"{DOWN}{3 LEFT}FFF"        :rem 187
110 PRINTLEFT$(S$,11)SPC(10-LEN(C$)):INPU
    TAN$:IFAN$=""THEN110             :rem 91
115 D=VAL(AN$):GOTO40               :rem 214
120 PRINTLEFT$(S$,13)SPC(7)"{9 SPACES}"
                                    :rem 121
125 PRINTLEFT$(S$,11)SPC(9)"{8 SPACES}"
                                    :rem 126
130 PRINTLEFT$(S$,12)SPC(9)"{N}{4 T}
    {DOWN}{5 LEFT}{N}"              :rem 125
135 PRINTLEFT$(S$,13)SPC(9-LEN(B$))B;E
                                    :rem 194
140 PRINTLEFT$(S$,11)SPC(10-LEN(C$)):INPU
    T"{RVS}";AN$:IFAN$=""THEN140 :rem 242
145 GOTO40                           :rem 55
150 FORI=130TO254:POKES+2,I:GOSUB30:NEXTI
    :POKES+2,0                      :rem 183
155 POKECM+P,2:GOSUB30:POKECM+P,5:GOSUB30
                                     :rem 66
160 POKES+3,220:POKESM+P,69:FORI=1TO290:N
    EXTI                             :rem 46
165 POKESM+P,64:FORI=1TO250:NEXTI :rem 80
170 POKESM+P,32:POKES+3,0:SC=SC-50:IFSC<0
    THENSC=0                         :rem 27
175 L=L-1:IFL=0THENL=1               :rem 59
180 P=3:M=0:PRINT"{CLR}":GOTO445 :rem 251
185 POKESM+(M-1),32:FORI=1TO6:POKECM+M,3:
    POKESM+M,60:POKECM+P,5:POKES+2,200
                                    :rem 101
190 POKESM+P,62:GOSUB30:POKESM+P,58:POKEC
    M+M,6:POKESM+M,61:GOSUB30     :rem 182
195 POKESM+M,32:POKESM+P,32:POKES+2,0:P=P
    -1:M=M-1:NEXTI                  :rem 167
200 FORI=12TO9STEP-1:POKECM+I,5:POKESM+I,
    62:POKECM+9,6:POKESM+9,60:GOSUB30
                                    :rem 146
205 POKES+2,200:POKECM+9,3:POKESM+I,58
                                      :rem 6
210 GOSUB30:POKES+2,0:POKESM+I,32:NEXTI
                                    :rem 187
215 FORI=1TO5:PRINT"{HOME}{RVS}{DOWN}
    {RED}"TAB(7)"**100**":POKES+2,220:GOS
    UB25:POKES+2,0                  :rem 182
220 PRINT"{HOME}{DOWN}{RVS}"TAB(6)"
    {8 SPACES}":GOSUB25:NEXTI:L=L+1
                                    :rem 139
225 SC=SC+100:P=3:M=0:BC=BC+1:IFBC>31THEN
    BC=27                           :rem 211
230 POKECB,BC:PRINT"{CLR}{BLU}":PRINTLEFT
    $(S$,4)J$                        :rem 99
235 POKE36869,255:PRINT"{HOME}{DOWN}{PUR}
    {4 SPACES}?????????????????{RVS}{RED}S"
                                    :rem 223
240 POKESM+(P-1),32:POKECM+P,5:POKESM+P,5
    9:GOSUB25:POKESM+P,58            :rem 81
245 IFSM+P=SM+18THEN185             :rem 131
250 POKESM+(M-1),32:POKECM+M,2:POKESM+M,6
    1:GOSUB25:POKECM+M,6:POKESM+M,60
                                      :rem 8
255 IFPEEK(SM+M)=PEEK(SM+P)THEN150:rem 76
260 PRINTLEFT$(S$,17)SPC(7)"{CYN}LEVEL:"L
    "{BLU}"                         :rem 130
265 PRINTLEFT$(S$,18)J$              :rem 68
270 PRINTLEFT$(S$,20)" "N$"'S SCORE:"SC
                                    :rem 139
275 PRINTLEFT$(S$,8)SPC(8)"{4 SPACES}
    {DOWN}{4 LEFT}{4 SPACES}{2 DOWN}
    {5 LEFT}{8 SPACES}":GOTO65    :rem 239
280 POKE36869,240:POKECB,254:PRINT"{CLR}
    {RVS}{DOWN}{2 RIGHT}"N$"' SCOREBOARD
    {OFF}","{2 DOWN}{RIGHT}PROBLEMS:"PR-1
                                    :rem 235
285 PRINT"{2 DOWN}{RIGHT}{GRN}RIGHT ANSWE
    RS:"R,"{2 DOWN}{RIGHT}{RED}WRONG ANSW
    ERS:"W,"{2 DOWN}{RIGHT}{BLK}GRADE:"IN
    T(100/(PR-1))*R"%"               :rem 20
290 INPUT"{2 DOWN}{RIGHT}PLAY AGAIN (Y/N)
    ";A$                             :rem 11
295 IFA$="Y"THENRUN                 :rem 146
300 PRINT"{CLR}":FORI=1TO8:PRINT"{DOWN}
    {2 RIGHT}{RED}G O O D B Y E ! !":GOSU
    B25:NEXTI:END                    :rem 91
305 FORF=7632TO7679:READA:POKEF,A:NEXTF
                                    :rem 198
310 DATA24,60,110,126,126,126,60,24
                                    :rem 215
315 DATA56,124,95,248,224,248,127,56
                                     :rem 48
320 DATA60,126,255,219,255,255,169,169
                                    :rem 146
325 DATA60,126,255,219,255,255,90,180
                                     :rem 89
330 DATA120,116,30,14,30,124,120,0
                                    :rem 148
335 DATA0,0,0,14,14,14,0,0          :rem 8
340 FORK=7424TO7431:READA:POKEK,A:NEXTK
                                    :rem 197
345 DATA,,,,,,,0                     :rem 26
350 POKECB,250:POKE36869,255:PRINTLEFT$(S
    $,11)"{2 RIGHT}{BLU}M {GRN}U{RED} N
    {BLU}C {BLK}H {GRN}M {RED}A {BLU}T
    {CYN}H"                         :rem 223
355 READF:IFF=-1THEN375             :rem 119
360 POKES+2,F:GOSUB30:POKES+2,0:GOSUB30:G
    OTO355                          :rem 169
365 DATA195,209,219,225,225,225,225
                                    :rem 250
370 DATA219,219,219,209,219,209,195,-1
                                    :rem 139
```

```
375 FORI=ØTO15:PRINTMID$(S$,2,1Ø)SPC(I)"
    {SPACE}{CYN}= {RED}<{2 SPACES}{GRN}:"
    ;:GOSUB35                      :rem 1Ø6
38Ø PRINTMID$(S$,2,1Ø)SPC(I)" {BLU}<
    {PUR}={2 SPACES}{GRN};":POKES+2,2ØØ:G
    OSUB35:POKES+2,Ø:NEXTI          :rem 9Ø
385 PRINTMID$(S$,2,1Ø)SPC(15)"{8 SPACES}"
                                   :rem 191
39Ø POKECB,254:POKE36869,24Ø:PRINT"{CLR}
    {3 DOWN}{BLU}":A$="WHAT IS YOUR NAME"
    :GOSUB55                       :rem 198
395 PRINTLEFT$(S$,9):PRINT"{GRN}":INPUTN$
    :PRINT"{BLU}"                   :rem 57
4ØØ PRINT"{CLR}"LEFT$(S$,6):A$="WHAT WOUL
    D YOU":GOSUB55                 :rem 113
4Ø5 PRINTLEFT$(S$,8):A$="LIKE TO PRACTICE
    ,":GOSUB55:PRINTLEFT$(S$,1Ø):A$=N$+":
    "                              :rem 134
41Ø GOSUB55:PRINTLEFT$(S$,12):A$="{RED}1)
    {GRN}ADDITION":GOSUB55          :rem 16
415 PRINTLEFT$(S$,14):A$="{RED}2){GRN}SUB
    TRACTION":GOSUB55              :rem 246
42Ø PRINTLEFT$(S$,16):A$="{RED}3){GRN}DIV
    ISION":GOSUB55                  :rem 12
425 PRINTLEFT$(S$,18):A$="{RED}4){GRN}MUL
    TIPICATION{BLU}":GOSUB55       :rem 176
43Ø GETQ:IFQ<1ORQ>4THEN43Ø         :rem 121
435 PRINT"{CLR}{2 DOWN}{RIGHT}LEVEL (1-9)
    ?"                             :rem 222
44Ø GETL:IFL<1ORL>9THEN44Ø         :rem 113
445 GOTO23Ø                        :rem 1Ø7
```

Program 2: Munchmath, 64 Version

```
1ØØ POKE56,48:CLR:PRINT"{CLR}":SM=1Ø73:CM
    =55345:L=1:BC=3                :rem 142
11Ø FORI=ØTO27:POKE54272+I,Ø:NEXT:POKE542
    96,15:POKE54277,18:POKE54278,165
                                    :rem 56
12Ø S$="{HOME}{21 DOWN}":SF=54272:WV=5427
    6                               :rem 67
13Ø J$="99999999999999999999":P=3:GOTO9
    ØØ                             :rem 111
14Ø :                              :rem 2Ø7
15Ø FORT=1TO3ØØ:NEXT:RETURN          :rem 8
16Ø FORT=1TO4Ø:NEXT:RETURN         :rem 218
17Ø FORT=1TO9Ø:NEXT:RETURN         :rem 224
18Ø :                              :rem 211
19Ø D=VAL(AN$):IFASC(AN$)=81ANDPR>1THEN8Ø
    Ø                              :rem 247
2ØØ IFINT(D)<>INT(C)ORAN$>"9"THEN23Ø
                                   :rem 1Ø9
21Ø P=P+1:R=R+1:M=M+1:SC=SC+1Ø:POKESF,223
    :POKESF+1,29:POKEWV,17          :rem 67
22Ø FORT=1TO5:NEXT:POKEWV,16:GOTO7ØØ
                                    :rem 65
23Ø M=M+1:W=W+1:PRINTLEFT$(S$,1Ø)SPC(2Ø-L
    EN(C$))"{RVS}{RED}"C"{5 SPACES}"
                                    :rem 1Ø
24Ø POKESF+1,8:POKESF,1ØØ:POKEWV,33:GOSUB
    15Ø:POKEWV,32:GOTO72Ø          :rem 136
25Ø :                              :rem 2Ø9
26Ø POKESF+1,4Ø:POKEWV,17          :rem 246
27Ø GOSUB16Ø:POKEWV,16:GOSUB16Ø:RETURN
                                   :rem 196
28Ø :                              :rem 212
29Ø PR=PR+1:A=INT(RND(1)*5*L)+I      :rem 3
3ØØ B=INT(RND(I)*5*L)+1:IFB>ATHENA=A+B
                                   :rem 2Ø2
31Ø E=INT(A*B):A$=STR$(A):B$=STR$(B)
                                    :rem 23
32Ø IFQ=1THENC=A+B:X=43:GOTO36Ø    :rem 13Ø
33Ø IFQ=2THENC=A-B:X=45:GOTO36Ø    :rem 136
34Ø IFQ=3THENC=A:GOTO41Ø           :rem 223
35Ø C=E:X=88                       :rem 156
36Ø C$=STR$(C):PRINTLEFT$(S$,7)SPC(2Ø-LEN
    (A$))"{RVS}"A                   :rem 33
37Ø PRINTLEFT$(S$,8)SPC(18-LEN(B$))"{RVS}
    "CHR$(X)B"{OFF}{DOWN}{3 LEFT}999"
                                   :rem 176
38Ø PRINTLEFT$(S$,1Ø)SPC(19-LEN(C$)):GOSU
    B123Ø:IFAN$=""THEN38Ø          :rem 12Ø
39Ø D=VAL(AN$):GOTO19Ø              :rem 17
4ØØ :                              :rem 2Ø6
41Ø PRINTLEFT$(S$,12)SPC(16)"{9 SPACES}"
                                   :rem 17Ø
42Ø PRINTLEFT$(S$,1Ø)SPC(18)"{8 SPACES}"
                                   :rem 171
43Ø PRINTLEFT$(S$,11)SPC(19)"7777":PRINTS
    PC(18)"8"                      :rem 1Ø9
44Ø PRINTLEFT$(S$,12)SPC(18-LEN(B$))"
    {RVS}"B;E                       :rem 7Ø
45Ø PRINTLEFT$(S$,1Ø)SPC(19-LEN(C$)):GOSU
    B123Ø:IFAN$=""THEN45Ø          :rem 116
46Ø GOTO19Ø                        :rem 1Ø9
47Ø :                              :rem 213
48Ø POKEWV,17:FORI=4TO33:POKESF+1,I:GOSUB
    16Ø:NEXT:POKEWV,16              :rem 54
49Ø POKECM+P,2:GOSUB16Ø:POKECM+P,5:GOSUB1
    6Ø                             :rem 172
5ØØ POKESF+1,14:POKEWV,33:POKESM+P,69:FOR
    I=1TO23Ø:NEXT                  :rem 154
51Ø POKESM+P,64:FORI=1TO25Ø:NEXT     :rem 1
52Ø POKESM+P,32:POKEWV,32:SC=SC-5Ø:IFSC<Ø
    THENSC=Ø                        :rem 75
53Ø L=L-1:IFL=ØTHENL=1              :rem 54
54Ø P=3:M=Ø:PRINT"{CLR}":GOTO121Ø  :rem 34
55Ø :                              :rem 212
56Ø POKESM+M-1,32:FORI=1TO6:POKECM+M,3:PO
    KESM+M,6Ø:POKECM+P,5            :rem 58
57Ø ?POKESF+1,14:POKEWV,129          :rem 47
58Ø POKESM+P,62:GOSUB16Ø:POKESM+P,58:POKE
    CM+M,6:POKESM+M,61:GOSUB16Ø     :rem 33
59Ø POKESM+M,32:POKESM+P,32:POKEWV,128:P=
    P-1:M=M-1:NEXT                 :rem 197
6ØØ FORI=12TO9STEP-1:POKECM+I,5:POKESM+I,
    62:POKECM+9,6:POKESM+9,6Ø:GOSUB16Ø
                                   :rem 2Ø2
61Ø POKESF+1,14:POKEWV,129:POKECM+9,3:POK
    ESM+I,58                       :rem 252
62Ø GOSUB16Ø:POKEWV,128:POKESM+I,32:NEXT
                                    :rem 19
63Ø FORI=1TO5:PRINT"{HOME}{RVS}{DOWN}
    {RED}"TAB(15)"** 1ØØ **":POKESF+1,15:
    POKEWV,33                       :rem 15
64Ø GOSUB15Ø:POKEWV,32              :rem 87
65Ø PRINT"{HOME}{DOWN}{RVS}"TAB(15)"
    {9 SPACES}":GOSUB15Ø:NEXT:L=L+1
                                   :rem 168
66Ø SC=SC+1ØØ:P=3:M=Ø:BC=BC+1:IFBC>31THEN
    BC=7                           :rem 164
67Ø REM SETUP                       :rem 18
68Ø POKE5328Ø,BC:POKE53281,1:PRINT"{CLR}
    {BLU}":PRINTLEFT$(S$,3)SPC(9)J$
                                    :rem 33
69Ø POKE53272,28:PRINT"{HOME}"SPC(13)"
    {DOWN}{PUR}??????????????{RVS}{RED}S"
                                   :rem 132
7ØØ POKESM+P-1,32:POKECM+P,5:POKESM+P,59:
    GOSUB15Ø:POKESM+P,58            :rem 48
71Ø IFSM+P=SM+18THEN56Ø            :rem 125
72Ø POKESM+M-1,32:POKECM+M,2:POKESM+M,61:
```

```
      GOSUB150:POKECM+M,6:POKESM+M,60
                                       :rem 232
730 IFPEEK(SM+M)=PEEK(SM+P)THEN480:rem 80
740 PRINTLEFT$(S$,16)SPC(16)"{RVS}{CYN}LE
      VEL:"L"{BLU}"                    :rem 198
750 PRINTLEFT$(S$,17)SPC(9)J$          :rem 178
760 PRINTLEFT$(S$,19)"{RVS}"SPC(12)N$"'S
      {SPACE}SCORE:"SC                 :rem 67
770 PRINTLEFT$(S$,7)SPC(17)"{4 SPACES}":P
      RINTSPC(17)"{4 SPACES}":PRINTSPC(13)"
      {DOWN}{8 SPACES}"                :rem 233
780 GOTO290                            :rem 115
790 :                                  :rem 218
800 POKE53272,21:POKE53280,6:POKE53281,7
                                       :rem 245
810 PRINT"{CLR}{DOWN}{RVS}"SPC(13-LEN(N$)
      /2)N$"'S SCOREBOARD"             :rem 255
820 PRINTSPC(14)"{2 DOWN}PROBLEMS:"PR-1
                                       :rem 199
830 PRINTSPC(12)"{2 DOWN}{GRN}RIGHT ANSWE
      RS:"R:PRINTSPC(12)"{2 DOWN}{RED}WRONG
       ANSWERS:"W                      :rem 151
840 PRINTSPC(14)"{2 DOWN}{BLK}GRADE:"INT(
      R/(PR-1)*100)"%"                 :rem 138
850 PRINTSPC(12)"{2 DOWN}PLAY AGAIN (Y/N)
      ?":POKE198,0                     :rem 141
860 GETA$:IFA$<>"Y"ANDA$<>"N"THEN860
                                       :rem 57
870 IFA$="Y"THENPR=0:R=0:W=0:SC=0:GOTO110
      0                                :rem 20
880 END                                :rem 119
890 :                                  :rem 219
900 FORF=55TO63:FORI=0TO7:READA:POKEF*8+I
      +12288,A:NEXT:NEXT               :rem 213
910 FORI=0TO7:POKE32*8+I+12288,0:NEXT
                                       :rem 186
920 :                                  :rem 213
930 POKE53281,2:POKE53281,7            :rem 251
940 PRINTLEFT$(S$,10)SPC(11)"{BLU}M {GRN}
      U{RED} N {BLU}C {BLK}H {GRN}M {RED}A
      {SPACE}{BLU}T {GRN}H"            :rem 207
950 POKE56334,PEEK(56334)AND254:POKE1,PEE
      K(1)AND251:Z=13312:Y=53248       :rem 96
960 FORI=0TO519:POKEI+Z,PEEK(I+Y):NEXT:FO
      RI=664TO671:POKEI+Z,PEEK(I+Y):NEXT
                                       :rem 68
970 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334
      )OR1                             :rem 143
980 POKE53272,28:PRINTLEFT$(S$,10)SPC(11)
      "{RVS}{BLU}M {GRN}U{RED} N {BLU}C
      {BLK}H {GRN}M {RED}A {BLU}T {GRN}H"
                                       :rem 231
990 :                                  :rem 220
1000 READF,G:IFF=-1THEN1040            :rem 52
1010 POKESF+1,F:POKESF,G:POKEWV,33:GOSUB1
      60:POKEWV,32:GOSUB160            :rem 190
1020 GOTO1000                          :rem 189
1030 :                                 :rem 254
1040 GOSUB150:FORI=4TO24:PRINTLEFT$(S$,10
      )SPC(I)" {CYN}= {RED}<{2 SPACES}
      {GRN}:";:GOSUB170                :rem 7
1050 PRINTLEFT$(S$,10)SPC(I)" {BLU}<
      {PUR}={2 SPACES}{GRN};"          :rem 72
1060 POKESF,195:POKESF+1,17:POKEWV,17:GOS
      UB170:POKEWV,16:NEXT             :rem 106
1070 PRINTLEFT$(S$,10)SPC(24)"{8 SPACES}"
                                       :rem 218
1080 POKE53280,5:POKE53281,7:POKE53272,21
                                       :rem 37
1090 PRINT"{CLR}"SPC(8)"{3 DOWN}{BLU}WHAT
      IS YOUR NAME";:GOSUB260:INPUTN$
                                       :rem 60
1100 PRINT"{CLR}{BLU}"SPC(13)"{5 DOWN}WHA
      T WOULD YOU":GOSUB260            :rem 135
1110 PRINTSPC(11)"{DOWN}LIKE TO PRACTICE,
      {DOWN}":GOSUB260                 :rem 224
1120 PRINTSPC(20-LEN(N$)/2)N$":":GOSUB260
                                       :rem 92
1130 PRINTSPC(14)"{DOWN}{RED}1){GRN}ADDIT
      ION":GOSUB260                    :rem 117
1140 PRINTSPC(14)"{DOWN}{RED}2){GRN}SUBTR
      ACTION":GOSUB260                 :rem 121
1150 PRINTSPC(14)"{DOWN}{RED}3){GRN}DIVIS
      ION":GOSUB260                    :rem 146
1160 PRINTSPC(14)"{DOWN}{RED}4){GRN}MULTI
      PICATION{BLU}":GOSUB260          :rem 48
1170 GETA$:Q=VAL(A$):IFQ<1ORQ>4THEN1170
                                       :rem 82
1180 PRINTLEFT$(S$,Q*2+10)SPC(14)"{RVS}"M
      ID$(STR$(Q),2)                   :rem 49
1190 PRINTLEFT$(S$,20)SPC(14)"LEVEL (1-9)
      ?"                               :rem 124
1200 GETA$:L=VAL(A$):IFL<1ORL>9THEN1200
                                       :rem 60
1210 GOTO680                           :rem 155
1220 :                                 :rem 255
1230 PRINT"{RVS}? ";:AN$="":POKE198,0
                                       :rem 248
1240 GETZA$:IFZA$=""THEN1240           :rem 101
1250 ZL=LEN(AN$):IFZA$=CHR$(20)ANDZLTHENP
      RINTZA$;:AN$=LEFT$(AN$,ZL-1):rem 227
1260 IFZA$=CHR$(13)THENPRINT:RETURN
                                       :rem 224
1270 IFZA$<>"Q"AND(ZA$<"0"ORZA$>"9")ORZL=
      5THEN1240                        :rem 132
1280 PRINTZA$;:AN$=AN$+ZA$:GOTO1240
                                       :rem 83
1290 :                                 :rem 6
1300 DATA 0,0,0,0,0,0,255,255          :rem 106
1310 DATA 3,3,3,3,3,3,3,3              :rem 171
1320 DATA 0,0,255,255,255,255,0,0 :rem 68
1330 DATA 24,60,110,126,126,126,60,24
                                       :rem 10
1340 DATA 56,124,95,248,224,248,127,56
                                       :rem 95
1350 DATA 60,126,255,219,255,255,169,169
                                       :rem 198
1360 DATA 60,126,255,219,255,255,90,180
                                       :rem 137
1370 DATA 120,116,30,14,30,124,120,0
                                       :rem 201
1380 DATA 0,0,0,14,14,14,0,0           :rem 57
1390 :                                 :rem 7
1400 DATA 16,195,22,96,28,49,33,125,33,12
      5,33,125,33,125                  :rem 195
1410 DATA 28,49,28,49,28,49,22,96,28,49,2
      2,96,16,195,-1,0                 :rem 10
```

64 Timepiece

(Article on page 84).

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

```
10 S=54272                                   :rem 245
12 FORL=STOS+24:POKEL,0:NEXT                 :rem 55
14 POKES+5,9:POKES+15,30:POKES+12,9
                                             :rem 55
20 POKE657,0:POKE53280,6                     :rem 92
30 DIMA(40),B(48)                            :rem 94
40 POKE53280,1:POKE53281,1:C=54272
                                             :rem 121
50 PRINT"{CLR}{RVS}{BLK}"TAB(13)"64 TIMEP
   IECE{OFF}":PRINTTAB(7)"{DOWN}TO SET TI
   ME,PRESS ANY KEY"                         :rem 143
60 PRINT"{DOWN}{3 SPACES}WHEN{2 SPACES}EN
   TERING{2 SPACES}TIME, GIVE HOURS AND M
   INUTES IN THIS MANNER:";                  :rem 234
70 PRINT"{RED}{2 DOWN}{9 LEFT}915{RVS}RET
   URN{OFF} OR{13 LEFT}";                    :rem 243
80 PRINT"{RED}{DOWN}1231{RVS}RETURN{OFF}"
   :PRINTTAB(92)"(PRESS ANY KEY)":rem 254
83 POKE198,0                                 :rem 152
85 GETA$:IFA$=""THEN85                       :rem 253
90 GOSUB1000                                 :rem 170
100 REM DRAW CLOCK                           :rem 15
101 PRINT"{HOME}{BLK}{8 SPACES}{RVS}
    {24 SPACES}{OFF}"                        :rem 170
102 FORI=1TO23:PRINTTAB(8)"{RVS} {OFF}"TA
    B(31)"{RVS} {OFF}":NEXT                  :rem 53
103 PRINT"{8 SPACES}{RVS}{24 SPACES}{OFF}
    ";                                       :rem 68
105 PRINT"{HOME}{2 DOWN}{RED}{RVS}"TAB(21
    )"{9 SPACES}"                            :rem 153
110 PRINT"{BLK}{3 DOWN}{9 RIGHT}{RVS}
    {2 SPACES}{UP}{LEFT}{OFF}{D}{RVS}
    {DOWN}1234{RED}{8 SPACES}"               :rem 178
120 PRINT"{4 DOWN}{9 RIGHT}{BLK}{RVS} 112
    2334455{RED}{5 SPACES}"                  :rem 105
130 PRINT"{9 RIGHT}{BLK}{RVS}5050505050S
    {RED}{5 SPACES}"                         :rem 86
140 PRINT"{9 RIGHT}{BLK}{4 DOWN}{RVS}
    {9 SPACES}111{RED}{5 SPACES}"
150 PRINT"{9 RIGHT}{BLK}{RVS}123456789012
    {RED}{4 SPACES}{OFF}£"                   :rem 213
160 PRINTTAB(26)"{RED}{RVS}£{DOWN}
    {2 LEFT}£{OFF}£{DOWN}{3 LEFT}{RVS}
    £{OFF}£"                                 :rem 220
170 PRINTTAB(9)"{DOWN}{BLU}{5 SPACES}QQQQ
    QQQQQQQQQQQ{5 UP}";:POKE1974,81:POKE
    1974+C,6                                 :rem 251
200 U=0                                      :rem 84
210 FORI=1974TO1094STEP-40:A(U)=I:U=U+1:N
    EXT                                      :rem 176
212 FORI=1093TO1084STEP-1:A(U)=I:U=U+1:NE
    XT                                       :rem 118
214 FORI=1123TO1243STEP40:A(U)=I:U=U+1:NE
    XT                                       :rem 117
216 FORI=1242TO1239STEP-1:A(U)=I:U=U+1:NE
    XT                                       :rem 120
218 U=1                                      :rem 94
220 FORI=1235TO1247:B(U)=I:U=U+1:NEXT
                                             :rem 219
222 FORI=1287TO1447STEP40:B(U)=I:U=U+1:NE
    XT                                       :rem 134
224 B(U)=1448:B(U+1)=1449:U=U+2   :rem 153
226 FORI=1489TO1689STEP40:B(U)=I:U=U+1:NE
    XT                                       :rem 150
227 B(U)=1690:B(U+1)=1730:B(U+2)=1770:B(U
    +3)=1809                                 :rem 48
228 B(U+4)=1848:B(U+5)=1887:B(U+6)=1926:G
    OTO900                                   :rem 42
300 REM START OF BALL MOVEMENT ROUTINE
                                             :rem 70
310 FORX=0TO40                               :rem 71
320 POKEA(X),81:POKEA(X)+C,6                 :rem 102
325 IFX>22THENPOKEA(X-1),32                  :rem 90
330 IFX>1ANDX<24THENPOKEA(X-1),99:POKEA(X
    -1)+C,6                                  :rem 126
340 IFX>2THENPOKEA(X-2),32                   :rem 38
350 GOSUB2000:NEXT:POKE1974,81:POKE1974+C
    ,6                                       :rem 52
400 REM CONTINUE TO MINUTES                  :rem 165
410 L=A(40)                                  :rem 20
420 IFPEEK(L-1)=81THEN500                    :rem 175
430 L=L-1:POKEL+1,32:POKEL,81:POKEL+C,6:I
    FPEEK(L-1)=81THEN900                     :rem 146
440 IFL=A(40)-4THEN900                       :rem 207
450 GOSUB2010:GOTO430                        :rem 230
460 POKES+4,20:POKES+11,20:POKES+24,0
                                             :rem 137
500 REM MINUTES FULL,{9 SPACES}GOTO 5'S
                                             :rem 229
505 Z=1447                                   :rem 1
510 FORX=5TO22:POKEB(X),81:POKEB(X)+C,6:P
    OKEB(X-5),32:GOSUB2040:NEXT   :rem 155
520 FORX=23TO38:POKEB(X),81:POKEB(X)+C,6
                                             :rem 147
525 Z=Z-1:IFPEEK(Z)<>81ANDZ>1432THENPOKEZ
    ,81:POKEZ+C,6:POKEZ+1,32         :rem 44
530 POKEB(X-4),32:GOSUB2080:NEXT:POKES+4,
    20:POKES+11,20:POKES+24,0      :rem 202
600 IFPEEK(1444)<>81THEN900:REM OR 5'S AR
    E FULL                                   :rem 140
605 Z=1435                                   :rem 255
610 POKE1444,81:POKE1444+C,6:GOSUB2080
                                             :rem 178
612 FORX=1445TO1447:POKEX,81:POKEX+C,6:PO
    KEX-12,32:GOSUB2080:NEXT        :rem 28
620 FORX=18TO28:POKEB(X),81:POKEB(X)+C,6
                                             :rem 151
630 Z=Z+1:POKEZ,32:GOSUB2060:NEXT :rem 53
640 FORX=29TO37:POKEB(X),81:POKEB(X)+C,6:
    POKEB(X-12),32                           :rem 16
650 GOSUB2040:NEXT                           :rem 90
659 Z=B(37-11)                               :rem 199
660 FORX=37TO44:POKEB(X),81:POKEB(X)+C,6:
    POKEB(X-11),32                           :rem 14
670 Z=Z-1:IFPEEK(Z)<>81THENPOKEZ,81:POKEZ
    +C,6:POKEZ+1,32                          :rem 248
680 GOSUB2040:NEXT                           :rem 93
690 IFZ-1>1672ANDPEEK(Z-1)<>81THEN692
                                             :rem 168
691 GOTO695                                  :rem 125
692 Z=Z-1:POKEZ,81:POKEZ+C,6:POKEZ+1,32:G
    OSUB2015:GOTO690                         :rem 31
695 POKES+4,20:POKES+11,20:POKES+24,0
                                             :rem 147
700 IFPEEK(1685)<>81THEN900                  :rem 27
710 REM HOURS ARE FILLED UP                  :rem 58
720 FORX=1686TO1690:POKEX,81:POKEX+C,6:PO
    KEX-12,32                                :rem 38
730 GOSUB2100:NEXT                           :rem 86
740 FORX=1TO12:POKEB(X+26),81:POKEB(X+26)
    +C,6:POKE1678+X,32                       :rem 212
750 GOSUB2100:NEXT                           :rem 88
760 FORX=1TO7:POKEB(27+X),32:GOSUB2110:NE
    XT                                       :rem 148
900 REM TIME FOR ACTION                      :rem 81
905 POKES+4,20:POKES+11,20:POKES+24,0
                                             :rem 141
910 IFRIGHT$(TI$,2)>"57"THEN300              :rem 235
920 GETA$:IFA$<>""THENRUN                    :rem 240
930 GOTO900                                  :rem 110
1000 REM TIME SETTING ROUTINE                :rem 24
1010 PRINT"{CLR}"TAB(53)"TIME?{HOME}"TAB(
```

```
     9Ø)"(IN 3 OR 4 DIGITS)"          :rem 4
1Ø2Ø INPUTA$                          :rem 184
1Ø3Ø A$="Ø"+A$:A$=RIGHT$(A$,4)        :rem 254
1Ø4Ø A=VAL(LEFT$(A$,2)):B=VAL(RIGHT$(A$,2
     )):D=INT(B/5):B=B-5*D            :rem 229
1Ø45 IFA>12ORD>11THEN1ØØØ             :rem 176
1Ø5Ø REM SETTING CLOCK                :rem 52
1Ø55 PRINT"{CLR}"                     :rem 47
1Ø6Ø FORX=1TOA:POKE1672+X,81:POKE1672+X+C
     ,6:NEXT:REM HOURS                :rem 93
1Ø7Ø X=Ø                              :rem 141
1Ø8Ø X=X+1:IFX>BTHEN11ØØ              :rem 164
1Ø9Ø POKE1234+X,81:POKE1234+X+C,6:GOTO1Ø8
     Ø:REM MINUTES                    :rem 224
11ØØ X=Ø                              :rem 135
111Ø X=X+1:IFX>DTHEN113Ø              :rem 163
112Ø POKE1432+X,81:POKE1432+X+C,6:GOTO111
     Ø                                :rem 145
113Ø GOTO1ØØ                          :rem 143
2ØØØ REM SOUND                        :rem 47
2ØØ5 GOTO2Ø2Ø                         :rem 196
2Ø1Ø IFX=37THEN219Ø                   :rem 76
2Ø15 POKES+24,1Ø:POKES+11,21:FORV=1TO4:PO
     KES+8,13Ø                        :rem 219
2Ø17 FORW=1TO2:NEXT:POKES+8,24Ø:NEXT:RETU
     RN                               :rem 56
2Ø2Ø IFX<24THENFORY=1TO55:NEXT:RETURN
                                      :rem 2Ø2
2Ø3Ø IFX>33ANDX<37THENPOKES+4,2Ø:POKES+11
     ,2Ø:RETURN                       :rem 176
2Ø35 GOTO2Ø1Ø                         :rem 198
2Ø4Ø IFPEEK(1447)=81THENGOTO219Ø :rem 117
2Ø6Ø IFPEEK(1689)=81THENGOTO219Ø :rem 127
2Ø8Ø IFPEEK(B(29))=81THENGOTO219Ø:rem 167
2Ø85 IFPEEK(B(32))=81THENGOTO219Ø:rem 166
2Ø9Ø IFPEEK(B(26))=81THENGOTO219Ø:rem 165
21ØØ IFPEEK(177Ø)=81THENGOTO219Ø :rem 113
211Ø IFPEEK(1926)=81THENGOTO219Ø :rem 117
212Ø GOTO2Ø15                         :rem 198
218Ø IFX>4THEN2Ø15                    :rem 27
219Ø POKES+24,15:POKES+4,21:POKES+1,13Ø:F
     ORY=1TO9:NEXT                    :rem 48
2195 POKES+4,2Ø:POKES+11,2Ø:FORY=1TO6Ø:NE
     XT:RETURN                        :rem 155
```

64 Aardvark Attack

(Article on page 82.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

```
1ØØ PRINT"{CLR}":S=54272:FORI=ØTO24:POKES
    +I,Ø:NEXT:S1=S+7:S2=S+14:POKES+24,15
                                      :rem 22
11Ø POKE5328Ø,2:POKE53281,14:GOSUB72Ø
                                      :rem 113
12Ø PRINT"{CLR}{2 DOWN}{BLK} INSTRUCTIONS
    (Y/N)?"                           :rem 196
13Ø GETG$:IFG$=""THEN13Ø              :rem 87
14Ø IFG$="Y"THENGOSUB88Ø              :rem 184
15Ø PRINT"{CLR}"SPC(9)"{DOWN}{BLK}{RVS}
    {3 SPACES}AARDVARK{2 SPACES}ATTACK
    {3 SPACES}"                       :rem 1Ø1
16Ø PRINTSPC(1Ø)"{2 DOWN}{WHT}ALPHANUMERI
    C WEAPONS"                        :rem 189
17Ø PRINTSPC(14)"{4 DOWN}{WHT}WHICH LEVEL
    ?":PRINTSPC(1Ø)"{DOWN}{BLK}(1:NOVICE
    {2 SPACES}9:EXPERT)"              :rem 135
18Ø GETHF$:IFHF$<"1"ORHF$>"9"THEN18Ø
                                      :rem 45
19Ø HF=INT(2.5↑(9-VAL(HF$)))          :rem 49
2ØØ SC=Ø:GOSUB79Ø                     :rem 239
21Ø :                                 :rem 2Ø5
22Ø REM NEW LETTER                    :rem 5Ø
23Ø GOSUB82Ø:POKE56334,PEEK(56334)AND254:
    POKE1,PEEK(1)AND251               :rem 1Ø
24Ø FORI=ØTO3:FORJ=ØTO3:C(I,J)=Ø:NEXT:NEX
    T                                 :rem 163
25Ø L=INT(RND(I)*26+1):B=53248+L*8:F=Ø
                                      :rem 53
26Ø FORI=ØTO3                         :rem 11
27Ø F=ABS(F-1):T=PEEK(B+2*I+1-F)      :rem 9
28Ø FORJ=ØTO3:M=(FNM(J)ANDT)/(4↑J):C(I,J)
    =C(I,J)+INT(M*(4↑F)):NEXT         :rem 191
29Ø IFFTHEN27Ø                        :rem 56
3ØØ NEXT:POKE1,PEEK(1)OR4:POKE56334,PEEK(
    56334)OR1                         :rem 251
31Ø :                                 :rem 2Ø6
32Ø REM FLASHING                      :rem 197
33Ø POKES+5,Ø:POKES+6,24Ø:POKES,Ø:POKES+1
    ,Ø:POKES+4,33                     :rem 236
34Ø FORY=1TO7Ø:Z=INT(RND(1)*1Ø):ZZ=1Ø37+Z
    :POKEZZ+54272,1:POKEZZ,16Ø        :rem 81
35Ø POKES+1,RND(1)*15+1Ø:POKEZZ+54272,Ø:N
    EXT:POKES+4,8                     :rem 196
36Ø FORI=ØTO3:FORJ=ØTO3:POKE1135-J+4Ø*I,S
    (C(I,J)):NEXT:NEXT                :rem 123
37Ø LF=Ø:FH=Ø:SH=Ø                    :rem 16
38Ø :                                 :rem 213
39Ø REM START ATTACK                  :rem 198
4ØØ POKES1+5,Ø:POKES1+6,24Ø:POKES1,Ø:POKE
    S1+1,Ø:POKES1+4,17:POKE198,Ø      :rem 72
41Ø FORX=1TO18:POKES1+1,(19-X)*7      :rem 1ØØ
42Ø GETG$:IFG$=""THEN48Ø              :rem 97
43Ø IFLFTHEN46Ø                       :rem 129
44Ø IF(ASC(G$)-64)<>LTHEN48Ø          :rem 51
45Ø LF=1:FH=X:N=S:GOSUB1Ø1Ø           :rem 138
46Ø IFVAL(G$)-1<>ZAND(G$<>"Ø"ORZ=9)THEN48
    Ø                                 :rem 179
47Ø SH=X:POKE198,Ø:GOTO59Ø            :rem 67
48Ø FORY=1TOHF:NEXT:POKE(1Ø37+Z+X*4Ø),81+
    LF*128:NEXT:POKES1+4,8            :rem 79
49Ø :                                 :rem 215
5ØØ REM LOSE ROUND                    :rem 52
51Ø POKE198,Ø:N=S2:GOSUB1Ø1Ø:FORI=2TO18:P
    OKE5328Ø,IAND15:NEXT
52Ø FORXX=ØTO3:PB=(1797+Z+(4Ø*XX)):IFPEEK
    (PB)<>32THEN54Ø                   :rem 9Ø
53Ø NEXT:GOTO62Ø                      :rem 226
54Ø IFXX=3ORLFTHEN56Ø                 :rem 69
55Ø POKEPB+4Ø,32                      :rem 123
56Ø POKEPB,32:GOTO23Ø                 :rem 245
57Ø :                                 :rem 214
58Ø REM SCORING                       :rem 15Ø
59Ø N=S2:GOSUB1Ø1Ø:POKES1+4,8:SC=SC+INT((
    5ØØ-5*FH-2*SH)/SQR(HF)):GOTO23Ø
                                      :rem 2Ø
6ØØ :                                 :rem 2Ø8
61Ø REM LOSE ROUTINE                  :rem 212
62Ø FORY=1TO1ØØ:POKE5328Ø,RND(1)*255:POKE
    53281,RND(1)*255:NEXT             :rem 122
63Ø POKE5328Ø,2:POKE53281,14:PRINT"{CLR}
    {DOWN}{BLK} ";:FORI=1TO38:PRINT"*";:N
    EXT                               :rem 11Ø
```

```
640 PRINTSPC(12)"{DOWN}{WHT}SORRY, YOU LO
    ST."                              :rem 83
650 PRINTSPC(13)"{3 DOWN}{BLU}YOUR SCORE"
    ;SC                              :rem 245
660 PRINTSPC(15)"{DOWN}AT LEVEL ";HF$
                                      :rem 21
670 PRINTSPC(11)"{3 DOWN}{BLK}PLAY AGAIN
    {SPACE}(Y/N)?"                   :rem 198
680 GETY$:IFY$=""THEN680             :rem 143
690 IFY$="N"THENPRINT"{CLR}":END      :rem 30
700 GOTO150                          :rem 102
710 :                                :rem 210
720 DIMS(15),C(3,3)                  :rem 207
730 DEFFNM(X)=3*(4↑X)                :rem 200
740 FORX=0TO15:READS(X):NEXT:RETURN
                                      :rem 53
750 DATA32,108,123,98,124,225,255,254
                                      :rem 86
760 DATA126,127,97,252,226,251,236,160
                                     :rem 139
770 :                                :rem 216
780 REM SCREENS                      :rem 150
790 PRINT"{CLR}"                       :rem 4
800 POKE214,18:PRINT:PRINTSPC(10)"{BLK}
    EQ3{2 SPACES}1234567890{2 SPACES}
    EW3 LEVEL "HF$                   :rem 130
810 FORX=1TO4:PRINTSPC(10)"EQ3
    {2 SPACES}{BLU}{RVS}{10 SPACES}{OFF}
    {BLK}{2 SPACES}EW3":NEXT         :rem 234
820 PRINT"{HOME}"SPC(10)"{BLK}EQ3
    {2 SPACES}{RED}**********{2 SPACES}
    {BLK}EW3"                        :rem 152
830 PRINT"{HOME}":FORI=1TO6:PRINTSPC(27)"
    {RED}{RVS}{6 SPACES}":NEXT       :rem 130
840 PRINT"{HOME}{DOWN}":FORI=1TO4:PRINTSP
    C(28)"{BLK}{RVS}{4 SPACES}":NEXT
                                       :rem 7
850 PRINT"{HOME}":FORI=1TO18:PRINTSPC(10)
    "{BLK}EQ3{2 SPACES}{CYN}{10 SPACES}
    {BLK}{2 SPACES}EW3":NEXT         :rem 158
860 POKE214,16:PRINT:PRINTSPC(27)"SCORE"S
    C:RETURN                         :rem 149
870 :                                :rem 217
880 REM INSTRUCTIONS                  :rem 57
890 PRINT"{CLR}{DOWN}{BLK} ";:FORI=1TO38:
    PRINT"*";:NEXT:PRINT             :rem 119
900 PRINT"{DOWN}ANDROMEDAN AARDVARKS ARE
    {SPACE}ATTACKING EARTH"          :rem 213
910 PRINTSPC(6)"{DOWN}{WHT}THEY HAVE 26 K
    INDS OF BOMBS":PRINTSPC(18)"{DOWN}AND
    "                                :rem 196
920 PRINTSPC(6)"{DOWN}THEY'RE ATTACKING 1
    0 CITIES!"                       :rem 187
930 PRINT"{3 DOWN}{BLK} YOUR ALPHANUMERIC
     RADAR CAN SAVE EARTH"           :rem 215
940 PRINTSPC(4)"{3 DOWN}{WHT}FIRST: RECOG
    NIZE THE BOMB (A-Z)"             :rem 147
950 PRINTSPC(7)"{DOWN}THEN: SAVE THE CITY
     (0-9)"                          :rem 135
960 PRINTSPC(9)"{2 DOWN}{BLK}{RVS}HIT ANY
     KEY TO PLAY"                     :rem 51
970 GETG$:IFG$=""THEN970             :rem 111
980 RETURN                           :rem 129
990 :                                :rem 220
1000 REM EXPLOSION                   :rem 102
1010 POKEN+5,37:POKEN+6,252:POKEN,100:POK
     EN+1,5:POKEN+4,129:POKEN+4,128
                                     :rem 181
1020 RETURN                          :rem 163
```

Connect The Dots

(Article on page 88.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

Program 1:

Connect The Dots — VIC Version

```
1 PRINT"{CLR}{3 SPACES}QQ{3 SHIFT-SPACE}Q
  Q{2 SHIFT-SPACE}QQQ{2 SHIFT-SPACE}QQ
  {6 SHIFT-SPACE}Q{SHIFT-SPACE}Q
  {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
  {2 SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
  {8 SHIFT-SPACE}Q{SHIFT-SPACE}Q
  {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
  {2 SHIFT-SPACE}Q{3 SHIFT-SPACE}Q:rem 88
2 PRINT"{3 SPACES}Q{SHIFT-SPACE}Q
  {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
  {2 SHIFT-SPACE}Q{4 SHIFT-SPACE}Q
  {6 SHIFT-SPACE}QQ{3 SHIFT-SPACE}QQ
  {3 SHIFT-SPACE}Q{2 SHIFT-SPACE}QQ
                                 :rem 126
3 A$="Z":POKE36878,15:GOSUB910   :rem 210
6 PRINT"{5 SPACES}{2 DOWN}{RVS}{PUR}INSTR
  UCTIONS":PRINT"{DOWN}WHEN THE GRID APPE
  ARS,FIND THE SOLID DOT.        :rem 137
7 PRINT"{DOWN}TYPE THE NUMBER OF THEROW A
  ND HIT RETURN.{3 SPACES}{DOWN}TYPE THE
  {SPACE}LETTER OF THE";         :rem 238
8 PRINT"COLUMN AND HIT RETURN.":PRINT"
  {2 DOWN}{BLK} HIT ANY KEY TO PLAY.
                                 :rem 245
9 GETS$:IFS$=""THEN9             :rem 185
10 PRINT"{CLR}{2 DOWN}{RED}{2 SPACES}WHAT
    WOULD YOU LIKE"SPC(9)"TO DRAW?":PRINT
   "{BLU}{DOWN}{RVS}A{OFF}=BUTTERFLY (22
   {SPACE}DOTS)                  :rem 147
11 PRINT"{DOWN}{RVS}B{OFF}=MUSHROOM (12 D
   OTS)":PRINT"{DOWN}{RVS}C{OFF}=DOG (20
   {SPACE}DOTS)                  :rem 180
12 PRINT"{DOWN}{RVS}D{OFF}=HEART (10 DOTS
   )                             :rem 168
18 GETA$:IFA$=""THEN18           :rem 245
19 L=7746:C=38466:POKE36879,26:CD=30720
                                 :rem 152
20 PRINT"{CLR}{2 DOWN}{2 SPACES}ABCDEFGHI
   JKLMNOPQR                     :rem 241
30 FORT=1TO418:POKEL,79:L=L+1:POKEC,3:C=C
   +1:NEXT                       :rem 229
40 PRINT"{HOME}{21 DOWN}{2 SPACES}ABCDEFG
   HIJKLMNOPQR{2 SPACES}";        :rem 19
50 PRINT"{HOME}{3 DOWN} 1"SPC(18)"1
   {2 SPACES}2"SPC(18)"2{2 SPACES}3"SPC(1
   8)"3{2 SPACES}4"SPC(18)"4{2 SPACES}5"S
   PC(18)"5 ";                    :rem 41
60 PRINT" 6"SPC(18)"6{2 SPACES}7"SPC(18)"
   7{2 SPACES}8"SPC(18)"8{2 SPACES}9"SPC(
   18)"9 10"SPC(18)"1011";       :rem 198
65 PRINTSPC(18)"11"               :rem 62
70 PRINT"{UP}12"SPC(18)"1213"SPC(18)"1314
   "SPC(18)"1415"SPC(18)"1516"SPC(18)"161
   7"SPC(18)                      :rem 55
```

```
80 PRINT"1718"SPC(18)"18{HOME}"      :rem 106
89 RESTORE                           :rem 149
90 READB$:IFB$=A$THEN92               :rem 86
91 GOTO90                             :rem 12
92 IFA$="A"THEN500                   :rem 229
93 IFA$="B"THEN550                   :rem 236
94 IFA$="C"THEN600                   :rem 234
95 IFA$="D"THEN650                   :rem 241
99 FORT=1TO500:NEXT                  :rem 204
100 READA,B,E$,S:IFA=0THEN900        :rem 175
101 POKEA,81:POKEA+CD,B               :rem 97
105 PRINT"{HOME}{43 SPACES}{UP}"      :rem 11
110 INPUT"{BLU}{HOME}←NUMBER";N$:rem 109
115 FORT=1TO500:NEXT                 :rem 241
120 PRINTSPC(11)"{2 UP}↑LETTER";:INPUTL$
                                     :rem 194
130 IFE$=N$+L$THEN200                :rem 151
140 PRINT"{UP}{6 SPACES}TRY AGAIN":POKE36
    874,209:FORT=1TO500:NEXT:POKE36874,19
    5                                :rem 102
150 FORT=1TO500:NEXT:POKE36874,0 :rem 193
160 FORT=1TO1500:NEXT:GOTO105         :rem 43
200 FORT=1TO1000:NEXT:FORT=1TO18:POKESQ,1
    60:POKESQ+CD,B:IFSQ=ATHEN220 :rem 178
210 SQ=SQ+S:NEXT                      :rem 15
220 FORZ=200TO235:POKE36876,Z:NEXT:POKE36
    876,0:GOTO99                      :rem 15
500 POKE7799,77:POKE7799+CD,0:POKE7801,78
    :POKE7801+CD,0:POKE7822,160:POKE7822+
    CD,5                             :rem 254
510 SQ=7822:GOTO99                    :rem 47
550 POKE7976,160:POKE7976+CD,4:POKE7977,1
    60:POKE7977+CD,4:POKE8058,160:POKE805
    8+CD,5                           :rem 118
560 SQ=8058:GOTO99                    :rem 54
600 POKE7903,74:POKE7903+CD,0:POKE7904,75
    :POKE7904+CD,0:POKE7882,85:POKE7882+C
    D,0                              :rem 196
610 POKE7783,160:POKE7783+CD,2:SQ=7783:GO
    TO99                             :rem 141
650 PRINT"{8 DOWN}{7 RIGHT}{GRN}MATTHEW
    {2 DOWN}{8 LEFT}{BLU}JONATHAN{2 DOWN}
    {6 LEFT}{YEL}EMILY"               :rem 97
660 POKE7822,160:POKE7822+CD,2:SQ=7822:GO
    TO99                             :rem 128
800 PRINT"{HOME}{6 SPACES}GOOD JOB!
    {12 SPACES}DRAW AGAIN?{27 SPACES}"
                                     :rem 110
810 Y$="":GETY$:IFY$=""THEN810       :rem 189
820 IFY$="Y"THEN1                    :rem 224
830 IFY$="N"THENPRINT"{CLR}{8 DOWN}
    {7 SPACES}BYE NOW!":FORT=1TO3000:NEXT
    :END                              :rem 89
840 GOTO810                          :rem 110
900 PRINT"{HOME}{6 SPACES}GOOD JOB!
    {12 SPACES}DRAW AGAIN?{27 SPACES}"
                                     :rem 111
910 READB$:IFB$="Z"THEN930           :rem 241
920 GOTO910                          :rem 110
930 READP,D:IFP=-1ANDA$="Z"THENPOKE36875,
    0:RETURN                          :rem 32
940 IFP=-1THENPOKE36875,0:GOTO960:rem 240
950 POKE36875,P:FORT=1TOD:NEXT:GOTO930
                                     :rem 168
960 Y$="":GETY$:IFY$=""THEN960       :rem 201
970 IFY$="Y"THEN10                    :rem 22
980 IFY$="N"THENPRINT"{CLR}{8 DOWN}
    {7 SPACES}BYE NOW!":FORT=1TO3000:NEXT
    :END                              :rem 95
990 GOTO960                          :rem 122
1000 DATAA,7888,5,7I,22,7804,2,3M,-21
                                     :rem 137
1010 DATA7806,2,3O,1,7852,2,5Q,23,7940,2,
     9Q,22,7982,2,11O,21,8028,7,13Q,23
                                       :rem 4
1020 DATA8072,7,15Q,22,8093,7,16P,21,8090
     ,7,16M,-1,7998,7,12I,-23,8082,7,16E,
     21                              :rem 241
1030 DATA8079,7,16B,-1,8056,7,15A,-23,801
     2,7,13A,-22,7970,7,11C,-21,7924,2,9A
     ,-23                             :rem 10
1040 DATA7836,2,5A,-22,7794,2,3C,-21,7796
     ,2,3E,1,7888,2,7I,23,8042,5,14I,22,0
     ,0,0,0                          :rem 122
1500 DATAB,8063,5,15H,1,7975,4,11H,-22,79
     72,4,11E,-1,7950,4,10E,-22,7908,4,8G
     ,-21                             :rem 47
1510 DATA7913,4,8L,1,7959,4,10N,23,7981,4
     ,11N,22,7978,4,11K,-1,8066,4,15K,22
                                     :rem 100
1520 DATA8071,5,15P,1,8063,5,15H,-1,0,0,0
     ,0                              :rem 132
2000 DATAC,7867,2,6J,21,7862,2,6E,-1,7883
     ,2,7D,21,7993,2,12D,22,8039,2,14F,23
                                     :rem 123
2010 DATA8040,2,14G,1,8018,2,13G,-22,7995
     ,2,12F,-23,7973,2,11F,-22,7977,2,11J
     ,1                              :rem 184
2020 DATA8046,2,14M,23,8047,2,14N,1,8003,
     2,12N,-22,7980,2,11M,-23,7892,2,7M,-
     22                              :rem 209
2030 DATA7850,2,5O,-21,7852,0,5Q,1,7830,0
     ,4Q,-22,7829,0,4P,-1,7783,2,2N,-23,0
     ,0,0,0                          :rem 139
2500 DATAD,7753,2,1F,-23,7750,2,1C,-1,779
     2,2,3A,21,7924,2,9A,22,8108,2,17I,23
                                      :rem 94
2510 DATA7940,2,9Q,-21,7808,2,3Q,-22,7762
     ,2,1O,-23,7759,2,1L,-1,7822,2,4I,21,
     0,0,0,0                         :rem 188
4000 DATAZ,225,300,231,100,235,200,240,20
     0,235,200,231,200,225,200,231,200,23
     5,300                           :rem 203
4010 DATA231,100,225,200,215,200,225,100,
     0,300,225,100,-1,0               :rem 60
```

Program 2:

Connect The Dots — 64 Version

```
100 REM TITLE PAGE                    :rem 20
110 PRINT"{CLR}{7}":POKE53280,0:POKE532
    81,0                              :rem 31
120 PRINTSPC(10)"QQQ{3 SHIFT-SPACE}QQ
    {2 SHIFT-SPACE}QQQQQ{2 SHIFT-SPACE}QQ
    "                                 :rem 40
130 PRINTSPC(10)"Q{SHIFT-SPACE} Q
    {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
    {2 SHIFT-SPACE} Q{SHIFT-SPACE}
    {SHIFT-SPACE}Q{2 SPACES}Q"   :rem 180
140 PRINTSPC(10)"Q {SHIFT-SPACE}Q
    {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
    {2 SHIFT-SPACE} Q{SHIFT-SPACE}
    {2 SHIFT-SPACE}Q"                :rem 132
150 PRINTSPC(10)"Q {SHIFT-SPACE}Q
    {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
    {2 SHIFT-SPACE} Q{SHIFT-SPACE}
    {2 SPACES}{2 SHIFT-SPACE}Q"  :rem 133
160 PRINTSPC(10)"Q{SHIFT-SPACE} Q
    {SHIFT-SPACE}Q{2 SHIFT-SPACE}Q
    {2 SHIFT-SPACE} Q{SHIFT-SPACE}
    {2 SPACES}Q{2 SHIFT-SPACE}Q"  :rem 87
```

```
170 PRINTSPC(10)"QQQ{2 SHIFT-SPACE} QQ
    {3 SHIFT-SPACE} Q {SHIFT-SPACE}
    {SHIFT-SPACE}QQ"                   :rem 233
180 L=1114:C=55386:CD=54272:WV=54276
                                        :rem 220
190 A$="Z":POKE54296,15:POKE54277,22:POKE
    54278,165:GOSUB840                  :rem 114
200 :                                   :rem 204
210 REM INSTRUCTIONS                     :rem 44
220 PRINTSPC(13)"{2 DOWN}{WHT}INSTRUCTION
    S:"                                  :rem 22
230 PRINTSPC(9)"{DOWN}WHEN THE GRID APPEA
    RS,":PRINTSPC(10)"FIND THE SOLID DOT.
    "                                   :rem 141
240 PRINTSPC(9)"{DOWN}TYPE THE NUMBER OF
    {SPACE}THE":PRINTSPC(10)"ROW AND HIT
    {SPACE}RETURN."                     :rem 171
250 PRINTSPC(6)"{DOWN}THEN TYPE THE LETTE
    R OF THE"                           :rem 126
260 PRINTSPC(9)"COLUMN AND HIT RETURN."
                                        :rem 109
270 PRINTSPC(10)"{2 DOWN}{RVS}HIT ANY KEY
     TO PLAY.":POKE198,0                 :rem 90
280 GETS$:IFS$=""THEN280                :rem 123
290 :                                   :rem 213
300 REM DRAW SELECTION                   :rem 75
310 POKE53281,6:PRINT"{CLR}"SPC(6)"
    {5 DOWN}E3]WHAT WOULD YOU LIKE TO D
    RAW?"                                :rem 83
320 PRINTSPC(8)"E7]{3 DOWN}A - BUTTERFL
    Y (22 DOTS)"                        :rem 192
330 PRINTSPC(9)"{DOWN}B - MUSHROOM (12 DO
    TS)":PRINTSPC(11)"{DOWN}C - HORSE (20
     DOTS)"                              :rem 82
340 PRINTSPC(10)"{DOWN}D - HEART (10 DOTS
    )":POKE198,0                         :rem 73
350 GETA$:IFA$<"A"ORA$>"D"THEN350 :rem 95
360 :                                   :rem 211
370 REM DRAW BOARD                       :rem 20
380 PRINT"{CLR}E3]"SPC(11)"{2 DOWN}ABCD
    EFGHIJKLMNOPQR{HOME}"               :rem 210
390 FORRH=1TO18:FORT=1TO18:POKEL+T+RH*40,
    79:POKEC+T+RH*40,14:NEXT:NEXT:rem 170
400 PRINTSPC(11)"E3]{20 DOWN}ABCDEFGHIJ
    KLMNOPQR"                            :rem 19
410 PRINT"{HOME}{3 DOWN}E7]";  :rem 129
420 FORI=1TO18:PRINTSPC(8)RIGHT$(STR$(I),
    2)SPC(19)"EG]"RIGHT$(STR$(I),2):NEX
    T                                   :rem 137
430 :                                   :rem 209
440 REM FIND DATA                       :rem 183
450 RESTORE                             :rem 189
460 READB$:IFB$<>A$THEN460              :rem 243
470 ONASC(A$)-64GOTO650,690,730,770
                                        :rem 139
480 FORT=1TO500:NEXT                    :rem 246
490 READA,B,E$,S:IFA=0THEN830           :rem 189
500 POKEA,81:POKEA+CD,B                 :rem 100
510 PRINT"{HOME}{39 SPACES}"            :rem 122
520 PRINT"E7]{HOME} (←) NUMBER";:GOSUB
    930:N$=IN$                          :rem 195
530 PRINT"{HOME}"SPC(20)"(↑) LETTER";:GOS
    UB930:L$=IN$                         :rem 11
540 IFE$=N$+L$THEN610                   :rem 161
550 PRINT"{HOME}{BLK}{15 SPACES}TRY AGAIN
    {10 SPACES}"                        :rem 109
560 POKECD,48:POKECD+1,11:POKEWV,33:POKEW
    V,32                                 :rem 18
570 FORT=1TO400:NEXT:POKECD,195:POKECD+1,
    16:POKEWV,33:POKEWV,32              :rem 222
580 FORT=1TO400:NEXT                    :rem 246
590 FORT=1TO1200:NEXT:GOTO510            :rem 47
600 :                                   :rem 208
610 FORT=1TO700:NEXT:FORT=1TO18:POKESQ,16
    0:POKESQ+CD,B:IFSQ=ATHEN630   :rem 146
620 SQ=SQ+S:NEXT                         :rem 20
630 POKEWV,17:FORZ=9TO26:POKECD+1,Z:POKEC
    D,0:NEXT:POKEWV,16:GOTO480     :rem 84
640 :                                   :rem 212
650 POKE1242,77:POKE1242+CD,0:POKE1244,78
    :POKE1244+CD,0                      :rem 126
660 POKE1283,160:POKE1283+CD,5    :rem 166
670 SQ=1283:GOTO480                      :rem 91
680 :                                   :rem 216
690 POKE1563,160:POKE1563+CD,4:POKE1564,1
    60:POKE1564+CD,4                    :rem 241
700 POKE1717,160:POKE1717+CD,5    :rem 165
710 SQ=1717:GOTO480                      :rem 88
720 :                                   :rem 211
730 POKE1436,74:POKE1436+CD,0:POKE1437,75
    :POKE1437+CD,0                      :rem 137
740 POKE1397,85:POKE1397+CD,0     :rem 130
750 POKE1208,160:POKE1208+CD,2:SQ=1208:GO
    TO480                               :rem 146
760 :                                   :rem 215
770 PRINT"{HOME}{7 DOWN}"               :rem 249
780 PRINTSPC(16)"{GRN}MATTHEW"     :rem 70
790 PRINTSPC(16)"{2 DOWN}E7]JONATHAN"
                                         :rem 30
800 PRINTSPC(17)"{2 DOWN}E3]EMILY"
                                         :rem 64
810 POKE1283,160:POKE1283+CD,2:SQ=1283:GO
    TO480                               :rem 152
820 :                                   :rem 212
830 PRINT"{HOME}{10 SPACES}GOOD JOB! DRAW
     AGAIN?{3 SPACES}"                  :rem 113
840 READB$:IFB$<>"Z"THEN840              :rem 48
850 READPL,PH,D:IFPL=-1ANDA$="Z"THENPOKEW
    V,0:RETURN                           :rem 29
860 IFPL=-1THENPOKEWV,0:GOTO890   :rem 223
870 POKECD,PL:POKECD+1,PH:POKEWV,33:FORT=
    1TOD*75:NEXT:POKEWV,32               :rem 85
880 GOTO850                             :rem 118
890 GETY$:IFY$<>"Y"ANDY$<>"N"THEN890
                                        :rem 135
900 IFY$="Y"THEN310                      :rem 66
910 :                                   :rem 212
920 PRINT"{CLR}";:END                    :rem 75
930 PRINT"? ";:IN$=""                    :rem 93
940 PRINT"{RVS} {OFF}{LEFT}";           :rem 234
950 GETA$:IFA$=""THEN940                 :rem 94
960 ZL=LEN(IN$):IFA$=CHR$(20)ANDZLTHENPRI
    NTA$;:IN$=LEFT$(IN$,ZL-1)     :rem 30
970 IFA$=CHR$(13)ANDZLTHENPRINT" ":RETURN
                                         :rem 26
980 IF(A$<"0"ORA$>"R")OR(A$>"9"ANDA$<"A")
    ORLEN(IN$)=2THEN950                  :rem 67
990 PRINTA$;:IN$=IN$+A$:GOTO940     :rem 92
1000 :                                  :rem 251
1010 DATA A,1403,5,7I,40,1247,2,3M,-39
                                        :rem 119
1020 DATA 1249,2,3O,1,1331,2,5Q,41,1491,2
     ,9Q,40,1569,2,11O,39,1651,7,13Q,41
                                        :rem 236
1030 DATA 1731,7,15Q,40,1770,7,16P,39,176
     7,7,16M,-1,1603,7,12I,-41,1759,7,16E
     ,39                                :rem 235
1040 DATA 1756,7,16B,-1,1715,7,15A,-41,16
     35,7,13A,-40,1557,7,11C,-39  :rem 69
1050 DATA 1475,2,9A,-41                 :rem 115
1060 DATA 1315,2,5A,-40,1237,2,3C,-39,123
     9,2,3E,1,1403,2,7I,41,1683,5,14I,40
                                        :rem 216
1070 DATA 0,0,0,0                        :rem 38
```

```
1080 :                                    :rem 3
1090 DATA B,1722,5,15H,1,1562,4,11H,-40,1
     559,4,11E,-1,1519,4,10E,-40   :rem 75
1100 DATA 1441,4,8G,-39            :rem 118
1110 DATA 1446,4,8L,1,1528,4,10N,41,1568,
     4,11N,40,1565,4,11K,-1,1725,4,15K,40
                                    :rem 53
1120 DATA 1730,5,15P,1,1722,5,15H,-1,0,0,
     0,0                           :rem 118
1130 :                             :rem 255
1140 DATA C,1364,2,6J,39,1359,2,6E,-1,139
     8,2,7D,39,1598,2,12D,40,1680,2,14F,4
     1                             :rem 111
1150 DATA 1681,2,14G,1,1641,2,13G,-40,160
     0,2,12F,-41,1560,2,11F,-40,1564,2,11
     J,1                           :rem 136
1160 DATA 1687,2,14M,41,1688,2,14N,1,1608
     ,2,12N,-40,1567,2,11M,-41,1407,2,7M,
     -40                           :rem 206
1170 DATA 1329,2,5O,-39,1331,0,5Q,1,1291,
     0,4Q,-40,1290,0,4P,-1,1208,2,2N,-41
                                   :rem 244
1180 DATA 0,0,0,0                   :rem 40
1190 :                               :rem 5
1200 DATA D,1160,2,1F,-41,1157,2,1C,-1,12
     35,2,3A,39,1475,2,9A,40,1803,2,17I,4
     1                              :rem 56
1210 DATA 1491,2,9Q,-39,1251,2,3Q,-40,116
     9,2,1O,-41,1166,2,1L,-1,1283,2,4I,39
                                    :rem 47
1220 DATA 0,0,0,0                   :rem 35
1230 :
1240 DATA Z,195,16,3, 31,21,1, 30,25,2, 1
     35,33,2                       :rem 169
1250 DATA 30,25,2, 31,21,2, 195,16,2, 31,
     21,2, 30,25,3, 31,21,1        :rem 233
1260 DATA 195,16,2                 :rem 115
1270 DATA 143,12,2, 195,16,1        :rem 36
1280 DATA 0,0,3, 195,16,1, -1,0,0:rem 205
```

VIC Super Expander Graphics

(Article on page 80.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

```
40 B=250:L=250:PI=3.14159265       :rem 230
50 C=3*PI:INC=PI/50                 :rem 89
60 DIM X(151),Y(151)               :rem 237
70 GRAPHIC2:COLOR6,6,7,7           :rem 174
100 FOR Z=2TO12                     :rem 71
110 FOR A=0 TO C STEP INC           :rem 35
120 R=B*COS(A*Z)+L                   :rem 0
130 S=S+1                          :rem 211
140 X(S)=.8*R*COS(A)+512            :rem 20
150 Y(S)=R*SIN(A)+512              :rem 139
160 POINT2,X(S),Y(S)               :rem 164
170 NEXT                           :rem 215
200 FOR K=50 TO 15STEP-5            :rem 13
210 FOR S=1 TO 100 STEP 5          :rem 224
220 DRAW2,X(S),Y(S)TO X(S+K) ,Y(S+K ):NEX
    T                              :rem 114
230 FOR TM=1 TO 500:NEXT            :rem 60
240 SCNCLR:NEXT K:S=0               :rem 25
250 NEXT Z:GOTO 100                 :rem 52
```

VIC-20 Character Developer

(Article on page 130.)

```
10 POKE 52,24: POKE 56,24: PRINT CHR$(147
   ) "ONE MOMENT PLEASE"            :rem 84
20 FORI=6144 TO 7679: POKE I, PEEK(I+2662
   4): NEXT                         :rem 99
30 POKE 36869,254: PRINT CHR$(147);
                                    :rem 89
40 PRINT: PRINT: FORI=1 TO 8: PRINT "
   {2 SPACES}--------": NEXT        :rem 87
100 PRINT CHR$(19); : PRINT: PRINT: FORI=
    1 TO 8                          :rem 86
110 M=8: N=0                        :rem 73
120 INPUT A$                       :rem 136
130 FORJ=0 TO 7: M=M-1             :rem 123
140 IF MID$(A$,J+1,1)="-" THEN 160
                                   :rem 123
150 N=N+2↑M: N(I)=N                 :rem 36
160 NEXTJ                           :rem 32
170 M=4: MS=0                      :rem 157
180 FORJ=0 TO 3: M=M-1             :rem 124
190 IF MID$(A$,J+1,1)="-" THEN 210
                                   :rem 124
200 MS=MS+2↑M                       :rem 23
210 NEXTJ                           :rem 28
220 IF MS<10 THEN X$=STR$(MS): MS$(I)=MID
    $(X$,2,1): GOTO 240            :rem 186
230 MS$(I)=CHR$(MS+55)             :rem 183
240 M=4: LS=0                      :rem 154
250 FORJ=4 TO 7: M=M-1             :rem 130
260 IF MID$(A$,J+1,1)="-" THEN 280
                                   :rem 129
270 LS=LS+2↑M                       :rem 28
280 NEXTJ                           :rem 35
290 IF LS<10 THEN X$=STR$(LS): LS$(I)=MID
    $(X$,2,1): GOTO 310            :rem 188
300 LS$(I)=CHR$(LS+55)             :rem 179
310 NEXTI                           :rem 28
320 PRINT CHR$(19);: PRINT , " DEC
    {3 SPACES}HEX":PRINT           :rem 200
330 FORI=1 TO 8: X$ = STR$(N(I)): X=LEN(X
    $)                             :rem 211
340 PRINT , N(I);: PRINT SPC(6-X) MS$(I)
    {SPACE}LS$(I): NEXT             :rem 96
350 FORI=0 TO 7: POKE 6144+I, N(I+1): NEX
    T: PRINT: PRINT "{6 SPACES}@": PRINT
                                   :rem 125
360 PRINT: PRINT " PRINT-OUT (Y/N)?"
                                    :rem 73
370 A$= "": GETA$: IFA$= "" THEN 370
                                   :rem 119
380 IFA$= "Y" THEN 500              :rem 45
400 FORI=1 TO 21: PRINT CHR$(157); : NEXT
    : PRINT "ARE YOU DONE (Y/N)?" :rem 14
410 A$= "": GETA$: IFA$="" THEN 410
                                   :rem 109
420 IF A$="N" THEN 440              :rem 32
430 END                            :rem 110
440 FORI=1 TO 12: PRINT CHR$(157); : NEXT
    I: PRINT CHR$(145)             :rem 162
450 PRINT "NEW CHARACTER (Y/N)?";:rem 130
460 A$= "": GETA$: IF A$= "" THEN 460
                                   :rem 119
```

```
470 IF A$="Y" THEN RUN 3Ø                  :rem 240
480 PRINT CHR$(19): FORI=1 TO 9: PRINT "
    {SPACE}" , "{1Ø SPACES}": NEXT
                                           :rem 141
490 FORI=1 TO 12: PRINT"{2Ø SPACES}": NEX
    T: RUN 1ØØ                             :rem 133
5ØØ OPEN1,4: CMD1                          :rem 151
51Ø SC=PEEK(648)*256: FOR P=SC TO SC+33Ø
                                           :rem 134
52Ø CH=PEEK(P): C$="": IF (P-SC)/22 = INT
    ((P-SC)/22) THEN PRINT CHR$(13);
                                           :rem 36
53Ø IF CH<32 THEN CH=CH+64: GOTO 55Ø
                                           :rem 119
54Ø IF CH>95 THEN CH=32                    :rem 187
55Ø C$=CHR$(CH): PRINT C$; : NEXT: PRINT:
     PRINT: PRINT#1: CLOSE1,4              :rem 231
56Ø GOTO 4ØØ                               :rem 104
```

How To Make Custom Characters On The 64

(Article on page 120.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

Program 1:
Character Set Transfer To RAM

```
1Ø POKE 56334,Ø:REM TURN OFF INTERRUPTS
                                          :rem 83
2Ø POKE 1,51:REM TURN OFF VIDEO CHIP TO E
   XPOSE CHARACTER GENERATOR              :rem 22Ø
3Ø FOR ADDRESS=2Ø48 TO 6143               :rem 2Ø4
4Ø POKE ADDRESS,PEEK(ADDRESS+512ØØ):REM C
   OPY CHARACTERS TO RAM                  :rem 32
5Ø NEXT ADDRESS                           :rem 17Ø
6Ø POKE 1,55:REM TURN ON VIDEO CHIP
                                          :rem 251
7Ø POKE 56334,129:REM TURN ON INTERRUPTS
                                          :rem 135
8Ø END                                    :rem 63
```

Program 2: Chred 64

```
1ØØ REM "CHRED 64"                         :rem 137
12Ø POKE5328Ø,11:POKE53281,Ø:PRINT"{5}"
                                           :rem 189
13Ø V=53248:SC=1Ø24:CB=2Ø48:CC=SC+4Ø*21+9
                                           :rem 222
14Ø SZ=7:FP=Ø:FO=Ø:TP=Ø:TY=Ø:SL=Ø:R$="Ø":
    C$="Ø"                                 :rem 199
15Ø CO$="* {RIGHT}{LEFT}{DOWN}{UP}{HOME}
    {CLR}{F1}{F3}{F5}{F7}SLRQ"             :rem 85
16Ø DN$="{HOME}{17 DOWN}"                  :rem 2
17Ø BL$="{18 SPACES}"                      :rem 2Ø3
18Ø NU$="Ø123456789ABCDEF"                 :rem 131
19Ø DEFFNA(F)=SC+62+4Ø*CY+CX               :rem 86
2ØØ DEFFNB(F)=CB+((FO+TP)*8)+TY            :rem 251
21Ø PRINT"{CLR}{5 DOWN}{9 SPACES}{YEL}CHA
    RACTER SET LOADING"                    :rem 97
22Ø GOSUB15ØØ                              :rem 218
225 POKE 49276,8:REM ENTER THIS LINE FOR
    {SPACE}DISK ONLY -- OMIT FOR CASSETTE
                                           :rem 94
23Ø SYS49152:POKEV+24,19                   :rem 127
24Ø POKE 53281,1:PRINT"{CLR}":POKE 53281,
    Ø:GOSUB1Ø6Ø                            :rem 12
25Ø CY=Ø:CX=Ø                              :rem 226
26Ø POKEFNA(Ø),PEEK(FNA(Ø))OR128           :rem 81
27Ø POKE198,Ø                              :rem 198
28Ø GETCH$:IFCH$=""THEN28Ø                 :rem 235
29Ø FORCH=1TOLEN(CO$):IFMID$(CO$,CH,1)=CH
    $THEN31Ø                               :rem 149
3ØØ NEXTCH:GOTO27Ø                         :rem 1Ø5
31Ø POKEFNA(Ø),PEEK(FNA(Ø))ANDNOT128
                                           :rem 112
32Ø ONCHGOTO33Ø,37Ø,41Ø,43Ø,45Ø,47Ø,49Ø,5
    ØØ,59Ø,77Ø,8ØØ,1Ø4Ø,91Ø,9ØØ,84Ø,1Ø2Ø
                                           :rem 119
33Ø POKEFNA(Ø),42                          :rem 173
34Ø GOSUB55Ø                               :rem 177
35Ø POKEFNB(Ø),PEEK(FNB(Ø))OR2↑(ABS(TX-7)
    )                                      :rem 2Ø8
36Ø GOTO26Ø                                :rem 1Ø6
37Ø POKEFNA(Ø),32                          :rem 176
38Ø GOSUB55Ø                               :rem 181
39Ø POKEFNB(Ø),PEEK(FNB(Ø))ANDNOT2↑(ABS(T
    X-7))                                  :rem 247
4ØØ GOTO26Ø                                :rem 1Ø1
41Ø CX=CX+1:IFCX>SZTHENCX=Ø                :rem 234
42Ø GOTO26Ø                                :rem 1Ø3
43Ø CX=CX-1:IFCX<ØTHENCX=SZ                :rem 236
44Ø GOTO26Ø                                :rem 1Ø5
45Ø CY=CY+1:IFCY>SZTHENCY=Ø                :rem 242
46Ø GOTO26Ø                                :rem 1Ø7
47Ø CY=CY-1:IFCY<ØTHENCY=SZ                :rem 244
48Ø GOTO26Ø                                :rem 1Ø9
49Ø GOTO25Ø                                :rem 1Ø9
5ØØ FORCY=ØTOSZ:FORCX=ØTOSZ:POKEFNA(Ø),32
                                           :rem 158
51Ø GOSUB55Ø                               :rem 176
52Ø POKEFNB(Ø),Ø                           :rem 121
53Ø NEXTCX,CY                              :rem 58
54Ø GOTO25Ø                                :rem 1Ø5
55Ø TP=FP:TX=CX:TY=CY:IFTX>7ANDTY<8THENTP
    =TP+1:TX=TX-8                          :rem 177
56Ø IFTX<8ANDTY>7THENTP=TP+2:TY=TY-8
                                           :rem 134
57Ø IFTY>7ANDTX>7THENTP=TP+3:TY=TY-8:TX=T
    X-8                                    :rem 189
58Ø RETURN                                 :rem 125
59Ø PRINTDN$"{RVS}{YEL}ROW, COLUMN?{OFF}
    {5} ";                                 :rem 1Ø7
6ØØ POKE198,Ø                              :rem 195
61Ø GETR$:IFR$=""THEN61Ø                   :rem 115
62Ø IFVAL(R$)<>ØANDVAL(R$)<4ORR$="Ø"THENR
    =VAL(R$):GOTO64Ø                       :rem 165
63Ø GOTO6ØØ                                :rem 1Ø4
64Ø PRINTR$",";                            :rem 72
65Ø POKE198,Ø                              :rem 2ØØ
66Ø GETC$:IFC$=""THEN66Ø                   :rem 95
67Ø IFC$=CHR$(2Ø)THENPRINT"{2 LEFT}
    {2 SPACES}";:GOTO59Ø                   :rem 1Ø
68Ø IFASC(C$)>64THENC=ASC(C$)-55:IFC>15TH
    EN6ØØ                                  :rem 174
69Ø IFVAL(C$)<>ØORC$="Ø"THENC=VAL(C$)
                                           :rem 118
7ØØ IFC>15THEN65Ø                          :rem 215
71Ø PRINTC$                                :rem 14Ø
72Ø FP=R*16+C                              :rem 189
73Ø IFSZ=15ANDFP>6ØTHENFP=6Ø:C$="C":C=12
                                           :rem 112
74Ø GOSUB129Ø                              :rem 231
75Ø PRINTDN$;BL$                           :rem 2Ø4
76Ø GOTO25Ø                                :rem 1Ø9
```

```
770 IFFO<191THENFO=FO+64:GOTO790 :rem 215
780 FO=Ø                          :rem 161
790 FP=Ø:R$="Ø":C$="Ø":GOSUB1240:GOTO250
                                  :rem 225
800 IFSZ=15THENSZ=7:GOTO830       :rem 213
810 IFFP>60THENFP=60:C$="C":C=12  :rem 76
820 SZ=15                         :rem 234
830 POKE 53281,1:PRINT"{CLR}":POKE 53281,
    Ø:CX=Ø:CY=Ø:GOSUB1060:GOTO250:rem 160
840 PRINTDN$;"{RVS}{YEL}ARE YOU SURE? ";
                                  :rem 156
850 POKE198,Ø                     :rem 202
860 GETCH$:IFCH$="N"THENPRINTDN$;"{OFF}
    Ɛ5Ǝ";BL$:GOTO250           :rem 134
870 IFCH$<>"Y"THEN860             :rem 193
880 PRINT"YES{OFF}Ɛ5Ǝ"         :rem 140
890 SYS49152:GOSUB1310:PRINTDN$;BL$:GOTO2
    50                            :rem 152
900 SL=1                          :rem 166
910 PRINTDN$;:INPUT"{RVS}{YEL}FILE NAME";
    NA$                           :rem 5
920 POKE253,LEN(NA$)              :rem 115
930 IFLEN(NA$)=ØTHEN970           :rem 74
940 FORL=1TOLEN(NA$)              :rem 196
950 POKE49359+L,ASC(MID$(NA$,L,1))
                                  :rem 125
960 NEXT                          :rem 222
970 SYS49269                      :rem 173
980 PRINTDN$;BL$;DN$;"{6 UP}"     :rem 167
990 IFSL=ØTHENSYS49292:GOTO1010   :rem 170
1000 SYS49310                     :rem 193
1010 POKE 53281,1:PRINT"{CLR}Ɛ5Ǝ":POKE
     {SPACE}53281,Ø:GOSUB1060:SL=Ø:GOTO25
     Ø                            :rem 32
1020 POKEV+24,21                  :rem 104
1030 PRINT"{CLR}{3 DOWN}":END     :rem 108
1040 IFPEEK(V+24)=19 THEN POKEV+24,21:GOT
     O260                         :rem 55
1050 POKEV+24,19:GOTO260          :rem 125
1060 PRINT"{HOME}{RVS}{YEL}CHARACTER EDIT
     OR{OFF}"                     :rem 65
1070 PRINT"{DOWN}{RVS}{YEL}F1{OFF}Ɛ5Ǝ E
     DIT NEW CHAR."               :rem 87
1080 PRINT"{RVS}{YEL}F3{OFF}Ɛ5Ǝ NEXT CH
     AR. BLOCK"                   :rem 227
1090 PRINT"{RVS}{YEL}F5{OFF}Ɛ5Ǝ BLOCK S
     IZE"                         :rem 150
1100 PRINT"{RVS}{YEL}F7{OFF}Ɛ5Ǝ FLIP CH
     ARACTER SET"                 :rem 142
1110 PRINT"{RVS}{YEL} R{OFF}Ɛ5Ǝ RESTORE
      FONT"                       :rem 27
1120 PRINT"{RVS}{YEL} S{OFF}Ɛ5Ǝ SAVE CH
     AR. SET"                     :rem 41
1130 PRINT"{RVS}{YEL} L{OFF}Ɛ5Ǝ LOAD CH
     AR. SET"                     :rem 20
1140 PRINT"{RVS}{YEL} Q{OFF}Ɛ5Ǝ QUIT"
                                  :rem 5
1150 PRINT"{HOME}{19 DOWN}{RVS}"TAB(21);"
      ";NU$;" {OFF}"              :rem 29
1160 FORL=1TO4:PRINTTAB(21)"{RVS}"MID$(NU
     $,L,1);SPC(16);" ":NEXT      :rem 164
1170 PRINTTAB(21)"{RVS}{18 SPACES}{OFF}
     {2 UP}"                      :rem 235
1180 PRINT"{HOME}"TAB(21);        :rem 116
1190 PRINT"{RVS} ";MID$(NU$,1,SZ+1);:PRIN
     T" {OFF}"                    :rem 105
1200 FORL=1TOSZ+1                 :rem 16
1210 PRINTTAB(21)"{RVS}"MID$(NU$,L,1);SPC
     (SZ+1) ;" {OFF}"             :rem 169
1220 NEXTL                        :rem 80
1230 PRINTTAB(21)"{RVS}";:FORL=ØTOSZ+2:PR
     INT" ";:NEXT:PRINT"{OFF}"    :rem 82
1240 CH=FO                        :rem 36
1250 FORY=1TO4                    :rem 77
1260 FORX=1TO16                   :rem 128
1270 POKESC+781+X+Y*40,CH:CH=CH+1:rem 143
1280 NEXTX,Y                      :rem 231
1290 PRINT"{HOME}{19 DOWN}{5 SPACES}{RVS}
     EDITING ";R$","C$"{OFF}":POKECC,FP+F
     O                            :rem 216
1300 IFSZ=15THENPOKECC+1,FP+FO+1:POKECC+4
     Ø,FP+FO+2:POKECC+41,FP+FO+3 :rem 125
1310 X=Ø:Y=Ø:CX=Ø:CY=Ø            :rem 15
1320 GOSUB1390                    :rem 19
1330 IFSZ<>15THEN1380             :rem 222
1340 X=8:Y=Ø:FP=FP+1:GOSUB1390    :rem 27
1350 X=Ø:Y=8:FP=FP+1:GOSUB1390    :rem 28
1360 X=8:Y=8:FP=FP+1:GOSUB1390    :rem 37
1370 FP=FP-3                      :rem 148
1380 RETURN                       :rem 172
1390 TP=FP:TX=CX:TY=CY:IFTX>7ANDTY<8THENT
     P=TP+1:TX=TX-8               :rem 228
1400 IFTX<8ANDTY>7THENTP=TP+2:TY=TY-8
                                  :rem 176
1410 IFTY>7ANDTX>7THENTP=TP+3:TY=TY-8:TX=
     TX-8                         :rem 231
1420 TE=8*(FO+TP)+CB:REM CHAR. POINTER
                                  :rem 239
1430 POKE251,TE-INT(TE/256)*256   :rem 233
1440 POKE252,INT(TE/256)          :rem 94
1450 TE=FNA(Ø)+X+40*Y:REM SCREEN LOC.
                                  :rem 117
1460 POKE253,TE-INT(TE/256)*256   :rem 238
1470 POKE254,INT(TE/256)          :rem 99
1480 SYS49209                     :rem 212
1490 RETURN                       :rem 174
1500 FORL=49152TO49319            :rem 232
1510 READD:POKEL,D:NEXT           :rem 197
1520 RETURN                       :rem 168
1530 REM FONT COPIER ROUTINE      :rem 204
1540 DATA120,169,51,133,1,169,1,141,13,22
     Ø,169,Ø,133,251,133,253,169,208,133
                                  :rem 189
1550 DATA252,169,8,133,254,160,Ø,177,251,
     145,253,230,251,230,253,208,246,230
                                  :rem 205
1560 DATA252,230,254,165,252,201,225,208,
     236,169,129,141,13,220,169,55,133,1
                                  :rem 205
1570 DATA88,96                    :rem 242
1580 REM CHAR EXPAND AND DISPLAY :rem 121
1590 DATA160,Ø,162,Ø,169,128,133,250,177,
     251,37,250,208,4,169,32,208,2,169,42
                                  :rem 3
1600 DATA145,253,24,102,250,240,8,230,253
     ,208,2,230,254,208,229,230,251,208,2
                                  :rem 230
1610 DATA230,252,165,253,24,105,33,133,25
     3,165,254,105,Ø,133,254,232,224,8,20
     8                            :rem 33
1620 DATA201,96                   :rem 17
1630 REM SAVE AND LOAD ROUTINES   :rem 73
1640 DATA169,128,133,157,169,1,162,1,160,
     1,32,186,255,165,253,162,208,160,192
                                  :rem 11
1650 DATA32,189,255,96,169,Ø,133,251,169,
     8,133,252,169,251,162,16,160,25
                                  :rem 33
1660 DATA32,216,255,96            :rem 116
1670 DATA169,Ø,162,Ø,160,8,32,213,255,96
                                  :rem 226
```

Binary Castle

(Article on page 38.)

> **BEFORE TYPING...**
> Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

Program 1: VIC Version

```
10 DIMBI$(15):D=21:D$="{20 DOWN}":PRINTCH
   R$(147)CHR$(152)                :rem 213
20 DATA"LƐ2 @Ǝ@","LLLƐ+Ǝ","LLƐ+Ǝ@",
   "LLƐ2 +Ǝ","LƐ+Ǝ@@","LƐ+ƎLƐ+Ǝ",
   "LƐ2 +Ǝ@","LƐ3 +Ǝ","Ɛ+Ǝ@@@","
   Ɛ+Ǝ@@Ɛ+Ǝ","Ɛ+ƎLƐ+Ǝ@"  :rem 148
30 DATA"Ɛ+Ǝ@Ɛ2 +Ǝ","Ɛ2 +Ǝ@@","
   Ɛ2 +ƎLƐ+Ǝ","Ɛ3 +Ǝ@","Ɛ4 +Ǝ"
                                   :rem 239
35 POKE53281,1:POKE53280,1         :rem 191
40 FORI=ØTO15:READBI$(I):NEXT:POKE53281,1
   :POKE53280,1                    :rem 179
50 PRINT"{HOME}"LEFT$(D$,D):RAN=INT(15*RN
   D(Ø)):PRINTSPC(8);BI$(RAN):INPUT"
   {HOME} WHAT NUMBER";N            :rem 67
60 IFN<>RANTHENPRINTLEFT$(D$,D)"SORRY,":P
   RINT"WRONG NUMBER":FORT=1TO1000:NEXTT
                                    :rem 86
70 IFN<>RANTHENPRINTD$:D=21:GOSUB200:PRIN
   TCHR$(147):GOTO50               :rem 186
80 D=D-1:IFD<=1THENPRINT"{HOME} YOU DID I
   T!!!":GOSUB 200:D=21:PRINT CHR$(147)
                                    :rem 66
90 PRINT"{HOME}{16 SPACES}":GOTO50:rem 37
100 END                            :rem 104
200 REM BELLS & WHISTLES           :rem 129
205 FOR J=30 TO 60 STEP 10          :rem 14
210 POKE54296,15:POKE54277,5:POKE54278,5:
    FORI=ØTO15:POKE53280,I:POKE53281,15-I
                                    :rem 57
220 POKE 54272,Ø:POKE54273,J+I:POKE54276,
    16+GT:GT=1-GT:NEXT:NEXT         :rem 66
230 POKE53281,1:POKE53280,1:POKE54276,16:
    POKE54296,Ø:RETURN             :rem 217
```

Program 2: 64 Version

```
10 DIMBI$(15):D=21:D$="{20 DOWN}":PRINTCH
   R$(147)                         :rem 235
20 DATA"LƐ2 @Ǝ@","LLLƐ+Ǝ","LLƐ+Ǝ@",
   "LLƐ2 +Ǝ","LƐ+Ǝ@@","LƐ+ƎLƐ+Ǝ",
   "LƐ2 +Ǝ@","LƐ3 +Ǝ","Ɛ+Ǝ@@@","
   Ɛ+Ǝ@@Ɛ+Ǝ","Ɛ+ƎLƐ+Ǝ@"  :rem 148
30 DATA"Ɛ+Ǝ@Ɛ2 +Ǝ","Ɛ2 +Ǝ@@","
   Ɛ2 +ƎLƐ+Ǝ","Ɛ3 +Ǝ@","Ɛ4 +Ǝ"
                                   :rem 239
40 FORI=ØTO15:READBI$(I):NEXT       :rem 34
50 PRINT"{HOME}"LEFT$(D$,D):RAN=INT(15*RN
   D(Ø)):PRINTSPC(8);BI$(RAN):INPUT"
   {HOME} WHAT NUMBER";N            :rem 67
60 IFN<>RANTHENPRINTLEFT$(D$,D)"SORRY,":P
   RINT"WRONG NUMBER":FORT=1TO1000:NEXTT
                                    :rem 86
70 IFN<>RANTHENPRINTD$:D=21:GOSUB200:PRIN
   TCHR$(147):GOTO50               :rem 186
80 D=D-1:IFD<=1THENPRINT"{HOME} YOU DID I
   T!!!":GOSUB 200:D=21:PRINTCHR$(147)
                                    :rem 66
90 PRINT"{HOME}{16 SPACES}":GOTO50:rem 37
100 END                            :rem 104
200 REM BELLS & WHISTLES           :rem 129
210 FORI=8TO24:POKE36879,I:POKE36878,10:P
    OKE36876,I+128:FORT=1TO200:NEXT:NEXT
                                   :rem 242
220 POKE36878,Ø:RETURN              :rem 73
```

Understanding Sound On The 64

(Article on page 136.)

```
100 I=52992                         :rem 34
110 READ A:IF A=256 THEN 190       :rem 156
120 POKE I,A:I=I+1:GOTO 110        :rem 226
130 DATA 24,5,6,Ø,1,2,3             :rem 77
140 DATA 21,12,13,7,8,9,10         :rem 234
150 DATA 11,19,20,14,15,16,17      :rem 128
160 DATA 23,4,11,18,162,Ø,188      :rem 135
170 DATA Ø,207,185,Ø,192,153,Ø     :rem 182
180 DATA 212,232,224,25,208,242,96,256
                                    :rem 80
190 POKE53281,1:POKE53280,1        :rem 241
200 POKE650,128                     :rem 35
210 F$="{19 SPACES}"               :rem 126
220 S=49152:D=Ø:Q=54272:P=53017:M$="VOICE
    ":Z$="{4 SPACES}{4 LEFT}":KE=197
                                   :rem 158
230 FORT=STOS+30:POKET,Ø:NEXT:SYSP
                                   :rem 255
240 PRINT"{CLR}";:FI$=" NONE "     :rem 211
250 FORA=1TO11:ON A GOSUB500,510,520,530,
    540,550,560,570,590,600,610:NEXT
                                   :rem 138
270 GETE$:U=PEEK(KE):IFU=64ANDPEEK(S+4)TH
    ENPOKES+4,PEEK(S+4)AND254:SYSP
                                   :rem 207
280 IFU=64ANDPEEK(S+7+4)THENPOKES+7+4,PEE
    K(S+7+4)AND254:SYSP            :rem 161
290 IFU=64ANDPEEK(S+14+4)THENPOKES+14+4,P
    EEK(S+14+4)AND254:SYSP          :rem 44
300 IFU=62THENSYSP:GOTO1330        :rem 212
310 IFE$="1"ORE$="2"ORE$="3"THEND=(ASC(E$
    )-49)*7:PRINT"{CLR}";TAB(25);M$;E$:GO
    TO250                            :rem 8
320 IFD>7THENPOKES+24,(PEEK(S+24)AND127):
    SYSP                           :rem 236
330 IFU=4THENPOKES+4+D,33:SYSP     :rem 133
340 IFU=5THENPOKES+4+D,17:SYSP     :rem 137
350 IFU=6THENPOKES+4+D,129:SYSP    :rem 191
360 IFU=3THENPOKES+4+D,65:SYSP     :rem 140
370 IF U=39THENPOKES+24,(PEEK(S+24)AND255
    ):FI$=" NONE{6 SPACES}":POKES+23,Ø:SY
    SP                             :rem 126
380 IF U=60 THENFORT=ØTO14STEP7:POKES+4+T
    ,PEEK(S+4+T)OR1:NEXT:SYSP      :rem 181
390 IFU=57THENPOKES+4+D,PEEK(S+4+D)OR3:SY
    SP                             :rem 200
400 IFU=54THENPOKES+4+D,21:SYSP    :rem 181
410 V=2↑(D/7)                      :rem 179
420 IFU=42THENFI$=" LOWPASS ":POKES+23,V:
    POKES+24,(PEEK(S+24)OR16):SYSP:rem 44
430 IFU=29THENFI$=" HIGHPASS ":POKES+23,V
    :POKES+24,(PEEK(S+24)OR64):SYSP
                                    :rem 99
440 IFU=28THENFI$=" BANDPASS ":POKES+23,V
    :POKES+24,(PEEK(S+24)OR32):SYSP
                                    :rem 83
```

```
450 N$="ADSROYTVFPW":FORJ=1TO LEN(N$):G$=
    MID$(N$,J):IF LEFT$(G$,1)=E$THEN48Ø
                                    :rem 206
46Ø NEXT                            :rem 217
47Ø GOTO27Ø                         :rem 109
48Ø ONLEN(G$)GOSUB61Ø,6ØØ,59Ø,57Ø,56Ø,55Ø
    ,54Ø,53Ø,52Ø,51Ø,5ØØ            :rem 155
49Ø GOTO27Ø                         :rem 111
5ØØ PRINT"{BLK}{HOME}{RVS}A{OFF}TTACK
    {2 SPACES}RATE +-":GOSUB62Ø:RETURN
                                    :rem 85
51Ø PRINT"{BLU}{HOME}{2 DOWN}{RVS}D{OFF}E
    CAY{2 SPACES}RATE +-":GOSUB7ØØ:RETURN
                                    :rem 180
52Ø PRINT"{RED}{HOME}{4 DOWN}{RVS}S{OFF}U
    STAIN LEVEL +-":GOSUB77Ø:RETURN
                                    :rem 232
53Ø PRINT"{GRN}{HOME}{6 DOWN}{RVS}R{OFF}E
    LEASE RATE{2 SPACES}+-":GOSUB84Ø:RETU
    RN                              :rem 153
54Ø PRINT"[1]{HOME}{8 DOWN}{RVS}O{OFF}V
    ERALL VOLUME +-":GOSUB91Ø:RETURN
                                    :rem 221
55Ø PRINT"[2]{HOME}{1Ø DOWN}PITCH (HIGH
     B{RVS}Y{OFF}TE)+-":GOSUB97Ø:RETURN
                                    :rem 74
56Ø PRINT"{PUR}{HOME}{12 DOWN}PI{RVS}T
    {OFF}CH (LOW BYTE)+-":GOSUB1Ø3Ø:RETUR
    N                               :rem 106
57Ø IFD>ØTHENPRINT"{HOME}{14 DOWN}NO RING
    /SYNC FOR VOICES TWO AND THREE":RETUR
    N                               :rem 38
58Ø PRINT"[7]{HOME}{14 DOWN}PITCH {RVS}
    V{OFF}OICE 3 (FOR RING)+-":GOSUB1Ø9Ø:
    RETURN                          :rem 44
59Ø PRINT"[4]{HOME}{16 DOWN}{RVS}F{OFF}
    ILTERS{2 SPACES}CUTOFF{2 SPACES}+-":G
    OSUB115Ø:RETURN                 :rem 160
6ØØ PRINT"[3]{HOME}{18 DOWN}{RVS}P{OFF}
    ULSE WAVE HIGH{2 SPACES}+-":GOSUB121Ø
    :RETURN                         :rem 178
61Ø PRINT"[2]{HOME}{2Ø DOWN}PULSE {RVS}
    W{OFF}AVE LOW{3 SPACES}+-":GOSUB127Ø:
    RETURN                          :rem 172
62Ø POKE198,Ø:GETA$:IF A$<>""THEN62Ø
                                    :rem 247
63Ø IF PEEK(KE)<>4ØANDPEEK(KE)<>43ANDPEEK
    (KE)<>1THEN68Ø                  :rem 23
64Ø IFPEEK(KE)=4ØANDX1<15THENX1=X1+1
                                    :rem 168
65Ø IFPEEK(KE)=43ANDX1>ØTHENX1=X1-1
                                    :rem 122
66Ø IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:F
    ORT=1TO5ØØ:NEXT:POKE198,Ø:PRINT:RETUR
    N                               :rem 61
67Ø PRINT"{RVS}";LEFT$(F$,X1);"{OFF}";RIG
    HT$(F$,15-X1);Z$;(PEEK(S+D+5)AND24Ø);
    "{2 UP}"                        :rem 82
68Ø POKES+D+5,(X1*16)+(PEEK(S+D+5)AND15):
    POKEQ+D+5,(PEEK(S+D+5))         :rem 113
69Ø GOTO63Ø                         :rem 113
7ØØ POKE198,Ø:IF PEEK(KE)<>4ØANDPEEK(KE)<
    >43ANDPEEK(KE)<>1THEN75Ø        :rem 122
71Ø IFPEEK(KE)=4ØANDX2<15THENX2=X2+1
                                    :rem 169
72Ø IFPEEK(KE)=43ANDX2>ØTHENX2=X2-1
                                    :rem 123
73Ø IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:F
    ORT=1TO5ØØ:NEXT:POKE198,Ø:PRINT:RETUR
    N                               :rem 59
74Ø PRINT"{RVS}";LEFT$(F$,X2);"{OFF}";RIG
    HT$(F$,15-X2);Z$;(PEEK(S+D+5)AND15);"
    {UP}"                           :rem 145
75Ø POKES+D+5,X2+(PEEK(S+D+5)AND24Ø):POKE
    Q+D+5,PEEK(S+D+5)               :rem 109
76Ø GOTO7ØØ                         :rem 109
77Ø POKE198,Ø:IF PEEK(KE)<>4ØANDPEEK(KE)<
    >43ANDPEEK(KE)<>1THEN82Ø        :rem 127
78Ø IFPEEK(KE)=4ØANDX3<15THENX3=X3+1
                                    :rem 179
79Ø IFPEEK(KE)=43ANDX3>ØTHENX3=X3-1
                                    :rem 133
8ØØ IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:F
    ORT=1TO5ØØ:NEXT:POKE198,Ø:PRINT:RETUR
    N                               :rem 57
81Ø PRINT"{RVS}";LEFT$(F$,X3);"{OFF}";RIG
    HT$(F$,15-X3);Z$;(PEEK(S+D+6)AND24Ø);
    "{UP}"                          :rem 194
82Ø POKES+D+6,(X3*16)+(PEEK(S+D+6)AND15):
    POKEQ+D+6,PEEK(S+D+6)           :rem 34
83Ø GOTO77Ø                         :rem 114
84Ø POKE198,Ø:IF PEEK(KE)<>4ØANDPEEK(KE)<
    >43ANDPEEK(KE)<>1THEN89Ø        :rem 132
85Ø IFPEEK(KE)=4ØANDX4<15THENX4=X4+1
                                    :rem 180
86Ø IFPEEK(KE)=43ANDX4>ØTHENX4=X4-1
                                    :rem 134
87Ø IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:F
    ORT=1TO5ØØ:NEXT:POKE198,Ø:PRINT:RETUR
    N                               :rem 64
88Ø PRINT"{RVS}";LEFT$(F$,X4);"{OFF}";RIG
    HT$(F$,15-X4);Z$;(PEEK(S+D+6)AND15);"
    {UP}"                           :rem 155
89Ø POKES+D+6,X4+(PEEK(S+D+6)AND24Ø):POKE
    Q+D+6,PEEK(S+D+6)               :rem 120
9ØØ GOTO84Ø                         :rem 110
91Ø POKE198,Ø:IF PEEK(KE)<>4ØANDPEEK(KE)<
    >43ANDPEEK(KE)<>1THEN96Ø        :rem 128
92Ø IFPEEK(KE)=4ØANDX5<15THENX5=X5+1
                                    :rem 181
93Ø IFPEEK(KE)=43ANDX5>ØTHENX5=X5-1
                                    :rem 135
94Ø IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:F
    ORT=1TO5ØØ:NEXT:POKE198,Ø:PRINT:RETUR
    N                               :rem 62
95Ø PRINT"{RVS}";LEFT$(F$,X5);"{OFF}";RIG
    HT$(F$,15-X5);Z$;(PEEK(S+24)AND15);"
    {UP}"                           :rem 92
96Ø POKES+24,(X5+(PEEK(S+24)AND24Ø)):SYSP
    :GOTO91Ø                        :rem 145
97Ø POKE198,Ø:IF PEEK(KE)<>4ØANDPEEK(KE)<
    >43ANDPEEK(KE)<>1THEN1Ø2Ø       :rem 170
98Ø IFPEEK(KE)=4ØANDX6<15THENX6=X6+1
                                    :rem 190
99Ø IFPEEK(KE)=43ANDX6>ØTHENX6=X6-1
                                    :rem 144
1ØØØ IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:
     FORT=1TO5ØØ:NEXT:POKE198,Ø:PRINT:RET
     URN                            :rem 98
1Ø1Ø PRINT"{RVS}";LEFT$(F$,X6);"{OFF}";RI
     GHT$(F$,15-X6);Z$;PEEK(S+D+1);"{UP}"
                                    :rem 50
1Ø2Ø POKES+1+D,16*X6:POKEQ+1+D,PEEK(S+1+D
     ):GOTO97Ø                      :rem 179
1Ø3Ø POKE198,Ø:IF PEEK(KE)<>4ØANDPEEK(KE)
     <>43ANDPEEK(KE)<>1THEN1Ø8Ø     :rem 212
1Ø4Ø IFPEEK(KE)=4ØANDX7<15THENX7=X7+1
                                    :rem 229
1Ø5Ø IFPEEK(KE)=43ANDX7>ØTHENX7=X7-1
                                    :rem 183
1Ø6Ø IFPEEK(KE)=1THENPOKE197,Ø:POKE198,Ø:
```

```
     FORT=1TO500:NEXT:POKE198,0:PRINT:RET
     URN                                 :rem 104
1070 PRINT"{RVS}";LEFT$(F$,X7);"{OFF}";RI
     GHT$(F$,15-X7);Z$;PEEK(S+D);"{UP}"
                                         :rem 222
1080 POKES+D,16*X7:POKEQ+D,PEEK(S+D):GOTO
     1030                                :rem 202
1090 POKE198,0:IF PEEK(KE)<>40ANDPEEK(KE)
     <>43ANDPEEK(KE)<>1THEN1140   :rem 215
1100 IFPEEK(KE)=40ANDX8<15THENX8=X8+1
                                         :rem 229
1110 IFPEEK(KE)=43ANDX8>0THENX8=X8-1
                                         :rem 183
1120 IFPEEK(KE)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RET
     URN                                 :rem 101
1130 PRINT"{RVS}";LEFT$(F$,X8);"{OFF}";RI
     GHT$(F$,15-X8);Z$;PEEK(S+15+D);"{UP}
     "                                   :rem 110
1140 POKEQ+24,PEEK(S+24)OR128:POKES+15+D,
     X8*16:POKEQ+15+D,X8*16:GOTO1090
                                         :rem 230
1150 POKE198,0:IF PEEK(KE)<>40ANDPEEK(KE)
     <>43ANDPEEK(KE)<>1THEN1200   :rem 209
1160 IFPEEK(KE)=40ANDX9<15THENX9=X9+1
                                         :rem 238
1170 IFPEEK(KE)=43ANDX9>0THENX9=X9-1
                                         :rem 192
1180 IFPEEK(KE)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RET
     URN                                 :rem 107
1190 PRINT"{RVS}";LEFT$(F$,X9);"{OFF}";RI
     GHT$(F$,15-X9);Z$;PEEK(S+22);"
     {6 RIGHT}";FI$;"{UP}"         :rem 32
1200 POKES+21,X9/2:POKES+22,(X9*16):POKEQ
     +21,7:POKEQ+22,(X9*16):GOTO1150
                                         :rem 168
1210 POKE198,0:IF PEEK(KE)<>40ANDPEEK(KE)
     <>43ANDPEEK(KE)<>1THEN1260   :rem 212
1220 IFPEEK(KE)=40ANDXA<15THENXA=XA+1
                                         :rem 3
1230 IFPEEK(KE)=43ANDXA>0THENXA=XA-1
                                         :rem 213
1240 IFPEEK(KE)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RET
     URN                                 :rem 104
1250 PRINT"{RVS}";LEFT$(F$,XA);"{OFF}";RI
     GHT$(F$,15-XA);Z$;PEEK(S+D+2);"{UP}"
                                         :rem 79
1260 POKES+D+2,XA*16:POKEQ+D+2,PEEK(S+D+2
     ):GOTO1210                    :rem 235
1270 POKE198,0:IF PEEK(KE)<>40ANDPEEK(KE)
     <>43ANDPEEK(KE)<>1THEN1320   :rem 215
1280 IFPEEK(KE)=40ANDXB<15THENXB=XB+1
                                         :rem 12
1290 IFPEEK(KE)=43ANDXB>0THENXB=XB-1
                                         :rem 222
1300 IFPEEK(KE)=1THENPOKE197,0:POKE198,0:
     FORT=1TO500:NEXT:POKE198,0:PRINT:RET
     URN                                 :rem 101
1310 PRINT"{RVS}";LEFT$(F$,XB);"{OFF}";RI
     GHT$(F$,15-XB);Z$;PEEK(S+D+3);"{UP}"
                                         :rem 79
1320 POKES+D+3,XB*16:GOTO1270     :rem 169
1330 REM SAVE ROUTINE
1340 S=49152:CO=52992             :rem 113
1350 PRINT"{CLR}":DIMQ(45),ML(45):rem 203
1360 FORT=0TO44:Q(T)=PEEK(S+T):ML(T)=PEEK
     (CO+T):NEXT                   :rem 231
1370 PRINT"1 RP=52992:FORR=RPTORP+44:READ
     GP:POKER,GP:NEXT"             :rem 197
1380 PG=0:FORA=0TO4:PG=PG+3          :rem 121
1390 PRINT PG"DATA";:FORT=0TO8 :PRINTML(T
     +9*A);:IF T<8 THENPRINT"{LEFT},";
                                     :rem 235
1400 NEXT:PRINT:NEXT                  :rem 68
1410 PRINT"20S=49152:FORT=STOS+24:POKET,0
     :NEXT:P=53017{2 SPACES}"     :rem 115
1420 PRINT"30FORT=STOS+25:READDS:POKET,DS
     :NEXT:SYSP{3 SPACES}"        :rem 189
1430 PO=30:FORW=0TO2:PO=PO+10          :rem 2
1440 PRINTPO"DATA";:FORT=0TO8:PRINTQ(T+9*
     W);:IFT<8THENPRINT"{LEFT},";:rem 189
1450 NEXT:PRINT:NEXT                  :rem 73
```

Machine Language For Beginners

(Article on page 164.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

The Assembler

```
10 H=1:REM IF H = 0 THEN ASSEMBLY IS IN D
   ECIMAL                              :rem 42
50 HE$="0123456789ABCDEF":SZ=1:ZO$="000"
                                      :rem 166
100 PRINT"{3 SPACES}SIMPLE{3 SPACES}ASSEM
    BLER{2 SPACES}CONVENTIONS:"   :rem 90
110 DIMM$(56),TY(56),OP(56)         :rem 181
120 FORI=1TO56:READM$(I)            :rem 160
122 ROP$=MID$(M$(I),4,1):TY(I)=VAL(ROP$)
                                       :rem 5
124 OP$=RIGHT$(M$(I),3):OP(I)=VAL(OP$)
                                      :rem 155
126 M$(I)=LEFT$(M$(I),3)            :rem 235
140 NEXTI: PRINT                    :rem 228
150 PRINT"IMMEDIATE{5 SPACES}LDA #15
                                       :rem 46
155 PRINT"ABSOLUTE{6 SPACES}LDA 1500
                                       :rem 64
160 PRINT"ZERO PAGE{5 SPACES}LDA 15
                                      :rem 218
165 PRINT"ACCUMULATOR{3 SPACES}ASL
                                      :rem 107
170 PRINT"INDIRECT X{4 SPACES}LDA (15X)
                                      :rem 209
175 PRINT"INDIRECT Y{4 SPACES}LDA (15)Y
                                      :rem 216
177 PRINT"ZERO PAGE X{3 SPACES}LDA 15X
                                      :rem 146
179 PRINT"ZERO PAGE Y{3 SPACES}LDX 15Y
                                      :rem 173
180 PRINT"ABSOLUTE X{4 SPACES}LDA 1500X
                                      :rem 238
185 PRINT"ABSOLUTE Y{4 SPACES}LDA 1500Y
                                      :rem 245
189 PRINT:PRINT"{4 SPACES}ENTER ALL NUMBE
    RS IN ";                         :rem 127
190 IFH=1 THENPRINT"HEX":GOTO200 :rem 201
195 PRINT"DECIMAL"                   :rem 95
```

```
200 PRINT:PRINT"PLEASE INPUT STARTING ADD
    RESS FOR ML PROGRAM":INPUT SA$
210 IFH=1THENH$=SA$:GOSUB5000:SA=DE:GOTO2
    20                              :rem 130
215 SA=VAL(SA$)                     :rem 85
220 TA=SA:PRINT"{CLR}":REM CLEAR THE SCRE
    EN                              :rem 190
230 IFH=1THENDE=SA:SZ=3:GOSUB4000:PRINTH$
    ;:GOTO240                       :rem 175
235 PRINTSA" ";                     :rem 58
240 INPUTMN$:PRINT"{UP}"SPC(20);:REM GO U
    P ONE LINE AND OVER 20 SPACES:rem 232
241 REM ADD NEW PSEUDO-OPS HERE    :rem 65
242 IFRIGHT$(MN$,7)="FORWARD"THENFB=SA
                                    :rem 90
243 IFRIGHT$(MN$,7)="RESOLVE"THENFR=SA-FB
    :POKEFB+1,FR-2:PRINT"{2 SPACES}OK":GO
    TO230                           :rem 72
244 IFRIGHT$(MN$,4)="POKE"THENPRINT"ADDR,
    NUMBER(DEC)";:INPUTADR,NUM:POKEADR,NU
    M:GOTO230                       :rem 116
250 IFMN$="END"THENPRINT:PRINT"{6 SPACES}
    PROGRAM IS FROM"TA"TO"SA:END   :rem 13
260 L=LEN(MN$):L$=LEFT$(MN$,3)     :rem 181
270 FORI=1TO56:IFL$=M$(I)THEN300 :rem 136
280 NEXTI                           :rem 34
290 GOTO850                         :rem 113
300 REM PRIMARY OPCODE CATEGORIES :rem 59
301 TY=TY(I):OP=OP(I)              :rem 20
305 IFFB=SATHENTN=0:GOTO2010       :rem 244
310 IFTY=0THENGOTO1000             :rem 102
320 IFTY=3THENTY=1:IFL=3THENOP=OP+8:GOTO1
    000                             :rem 81
330 R$=RIGHT$(MN$,L-4):IFH=1THENGOSUB6000
                                    :rem 200
340 LR$=LEFT$(R$,1):LL=LEN(R$):IFLR$="#"T
    HEN480                          :rem 184
350 IFLR$="("THEN520               :rem 88
360 IFTY=8THEN600                  :rem 15
370 IFTY=3THENOP=OP+8:GOTO1000    :rem 135
380 IFRIGHT$(R$,1)="X"ORRIGHT$(R$,1)="Y"T
    HEN630                          :rem 210
390 IFLEFT$(L$,1)="J"THEN820       :rem 44
400 TN=VAL(R$):IFTN>255THEN430     :rem 40
410 IFTY=1ORTY=3ORTY=4ORTY=5THENOP=OP+4
                                    :rem 133
420 GOTO2000                        :rem 145
430 H%=TN/256:L%=TN-256*H%:IFTY=2ORTY=7TH
    ENOP=OP+8:GOTO470               :rem 92
440 IFTY=1ORTY=3ORTY=4ORTY=5THENOP=OP+12:
    GOTO470                         :rem 197
450 IFTY=6ORTY=9THEN470            :rem 214
460 GOTO850                         :rem 112
470 GOTO3000                        :rem 151
480 TN=VAL(RIGHT$(R$,LL-1))        :rem 58
490 IFTY=1THENOP=OP+8:GOTO2000    :rem 137
500 IFTY=4ORTY=5THENGOTO2000       :rem 44
510 GOTO850                         :rem 108
520 IFRIGHT$(R$,2)=")Y"THEN540     :rem 184
530 IFRIGHT$(R$,2)="X)"THEN570     :rem 187
540 TN=VAL(MID$(R$,2,LL-3))        :rem 243
550 IFTY=1THENOP=OP+16:GOTO2000   :rem 181
560 GOTO850                         :rem 113
570 TN=VAL(MID$(R$,2,LL-3))        :rem 246
580 IFTY=1THENGOTO2000             :rem 113
590 GOTO850                         :rem 116
600 TN=VAL(R$):TN=TN-SA-2:IFTN<-128ORTN>1
    27THENPRINT"TOO FAR ";:GOTO850
                                    :rem 154
610 IFTN<0THENTN=TN+256            :rem 172
620 GOTO2000                        :rem 147
```

```
630 IFRIGHT$(R$,2)=")Y"THEN540     :rem 186
640 IFRIGHT$(R$,1)="X"THEN720      :rem 144
650 REM *ZERO Y                     :rem 66
660 TN=VAL(LEFT$(R$,LL-1)):IFTN>255THEN68
    0                               :rem 249
670 IFTY=2ORTY=5THEN730            :rem 209
675 IFTY=1THEN760                   :rem 24
680 GOSUB770:IFTY=1THENOP=OP+24:GOTO710
                                    :rem 230
690 IFTY=5THENOP=OP+28:GOTO710    :rem 151
700 GOTO850                         :rem 109
710 GOTO3000                        :rem 148
720 TN=VAL(LEFT$(R$,LL-1)):IFTN>255THENGO
    SUB770:GOTO780                  :rem 136
730 IFTY=2THENOP=OP+16:GOTO760    :rem 145
740 IFTY=1ORTY=3ORTY=5THENOP=OP+20:GOTO76
    0                               :rem 10
750 GOTO850                         :rem 114
760 GOTO2000                        :rem 152
770 H%=TN/256:L%=TN-256*H%:RETURN:rem 187
780 IFTY=2THENOP=OP+24:GOTO810    :rem 145
790 IFTY=1ORTY=3ORTY=5THENOP=OP+28:GOTO81
    0                               :rem 19
800 GOTO850                         :rem 110
810 GOTO3000                        :rem 149
820 TN=VAL(R$)                      :rem 35
830 GOSUB770                        :rem 185
840 GOTO710                         :rem 109
850 PRINT"{RVS} ERROR ":GOTO230    :rem 18
1000 REM 1 BYTE INSTRUCTIONS       :rem 191
1010 POKESA,OP:SA=SA+1:IFH=1THEN 1030
                                    :rem 189
1020 PRINTOP:GOTO230               :rem 247
1030 DE = OP:GOSUB4000:PRINTH$:GOTO230
                                    :rem 226
2000 REM 2 BYTE INSTRUCTIONS       :rem 193
2005 IFTN>256THENPRINT" INCORRECT ARGUMEN
     T. (#5 IN HEX IS #05)":GOTO230
                                    :rem 94
2010 POKESA,OP:POKESA+1,TN:SA=SA+2:IFH=1T
     HEN2030                        :rem 231
2020 PRINTOP;TN:GOTO230            :rem 213
2030 DE = OP:GOSUB4000:PRINTH$" ";:rem 90
2040 DE = TN:GOSUB4000:PRINTH$:GOTO230
                                    :rem 231
3000 REM 3 BYTE INSTRUCTIONS       :rem 195
3010 POKESA,OP:POKESA+1,L%:POKESA+2,H%:SA
     =SA+3:IFH=1THEN3030            :rem 172
3020 PRINTOP;L%;H%:GOTO230         :rem 77
3030 DE = OP:GOSUB4000:PRINTH$" ";:rem 91
3040 DE = L%:GOSUB4000:PRINTH$" ";:rem 46
3050 DE = H%:GOSUB4000:PRINTH$:GOTO230
                                    :rem 180
4000 REM{2 SPACES}DECIMAL TO HEX (DE TO H
     $)                             :rem 8
4010 H$="":FORM=SZTO0STEP-1:N%=DE/(16↑M):
     DE=DE-N%*16↑M:H$=H$+MID$(HE$,N%+1,1)
                                    :rem 179
4020 NEXT:SZ=1:RETURN              :rem 116
5000 REM{2 SPACES}HEX TO DECIMAL (H$ TO D
     E)                             :rem 9
5010 D=0:Q=3:FORM=1TO4:FORW=0TO15:IFMID$(
     H$,M,1)=MID$(HE$,W+1,1)THEN5030
                                    :rem 221
5020 NEXTW                          :rem 93
5030 D1=W*(16↑(Q)):D=D+D1:Q=Q-1:NEXTM:DE=
     INT(D):RETURN                  :rem 41
6000 REM ACCEPT HEX OPCODE INPUT AND TRAN
     SLATE IT TO DECIMAL            :rem 57
6010 IFLEFT$(R$,1)="#"THENH$="00"+RIGHT$(
```

```
     R$,2):GOSUB5000:R$="#"+STR$(DE):RETU
     RN                                :rem 234
6020 LS=LEN(R$):AZ$=LEFT$(R$,1):ZA$=MID$(
     R$,LS,1):IFAZ$<>"("THEN6050 :rem 126
6030 IFZA$="Y"THENH$="00"+MID$(R$,2,2):GO
     SUB5000:R$="("+STR$(DE)+")Y":RETURN
                                       :rem 30
6040 IFZA$=")"THENH$="00"+MID$(R$,2,2):GO
     SUB5000:R$="("+STR$(DE)+"X)":RETURN
                                       :rem 238
6050 IFZA$="X"ORZA$="Y"THEN6070    :rem 40
6060 H$=LEFT$(ZO$,4-LS)+R$:GOSUB5000:R$=S
     TR$(DE):RETURN                    :rem 44
6070 IFLS=5THENH$=LEFT$(R$,4):GOTO6090
                                       :rem 253
6080 H$="00"+LEFT$(R$,2)          :rem 186
6090 GOSUB5000:R$=STR$(DE)+ZA$:RETURN
                                       :rem 252
20000 DATAADC1097,AND1033,ASL3002,BCC8114
      ,BCS8176,BEQ8240,BIT7036,BMI8048
                                       :rem 93
20010 DATABNE8208,BPL8016,BRK0000,BVC8080
      ,BVS8112,CLC0024,CLD0216,CLI0088
                                       :rem 114
20020 DATACLV0184,CMP1193,CPX4224,CPY4192
      ,DEC2198,DEX0202,DEY0136,EOR1065
                                       :rem 184
20030 DATAINC2230,INX0232,INY0200,JMP6076
      ,JSR9032,LDA1161,LDX5162,LDY5160
                                       :rem 200
20040 DATALSR3066,NOP0234,ORA1001,PHA0072
      ,PHP0008,PLA0104,PLP0040,ROL3034
                                       :rem 185
20050 DATAROR3098,RTI0064,RTS0096,SBC1225
      ,SEC0056,SED0248,SEI0120,STA1129
                                       :rem 216
20060 DATASTX2134,STY2132,TAX0170,TAY0168
      ,TSX0186,TXA0138,TXS0154,TYA0152
                                       :rem 79
50000 PRINTX:POKE5,X:GOTO530        :rem 7
```

One-Touch Commands For The 64

(Article on page 159.)

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

```
1 POKE56,208:POKE55,0:F=0:C=PEEK(55)-120
  :IFC<0THENC=C+256:F=-1
2 D=PEEK(56)+F:POKE55,C:POKE56,D
3 S=828:I=146:GOSUB100
10 DATA32,198,3,165,55,133,251,133,253,1
   65,56,133,252,133,254,169
15 DATA49,133,167,169,133,133,168,169,13
   ,32,210,255,169,70,32,210
20 DATA255,165,167,32,210,255,169,61,32,
   210,255,169,63,32,210,255
25 DATA169,32,32,210,255,32,207,255,72,1
   60,0,165,168,145,55,104
30 DATA32,198,3,201,13,240,14,201,95,208
   ,2,169,13,145,55,32
35 DATA207,255,76,124,3,230,167,165,167,
   41,1,208,10,24,165,168
40 DATA105,4,133,168,76,170,3,56,165,168
   ,233,3,133,168,165,167
45 DATA201,57,144,163,120,169,L0,141,20,
   3,169,H0,141,21,3,88
50 DATA169,0,133,167,32,68,166,76,116,16
   4,166,55,208,2,198,56
55 DATA198,55,96
56 S=PEEK(55)+256*PEEK(56):I=120:GOSUB10
   0
57 SYS(828)
58 END
60 DATA165,167,240,59,160,0,177,251,32,L
   99,H0,176,12,165,55,197
65 DATA251,208,21,165,56,197,252,208,15,
   169,0,133,167,165,253,133
70 DATA251,165,254,133,252,76,49,234,166
   ,198,177,251,157,119,2,230
75 DATA198,32,L111,H0,165,198,201,11,144
   ,204,230,167,76,49,234,165
80 DATA215,32,L99,H0,176,3,76,49,234,165
   ,8,41,1,208,247,160
85 DATA0,177,251,197,215,208,6,32,L111,H
   0,76,L6,H0,32,L111,H0
90 DATA76,L81,H0,201,133,144,6,201,141,1
   76,2,56,96,24,96,166
95 DATA251,208,2,198,252,198,251,96,0,0
100 F=0:FORD=STOS+I:READA$:IFASC(A$)<58T
    HENA=VAL(A$):GOTO115
105 IFASC(A$)=76THENA=VAL(RIGHT$(A$,LEN(
    A$)-1))+PEEK(55):IFA>255THENA=A-256:
    F=1
110 IFASC(A$)=72THENA=VAL(RIGHT$(A$,LEN(
    A$)-1))+PEEK(56)+F:F=0
115 POKED,A:NEXT:RETURN
```

The Beginner's Corner

(Article on page 150.)

Bake A Cake

```
1 DIMI$(24),B$(24,1),M(11),C$(11):Z=24
                                       :rem 126
2 POKE 53281,1                         :rem 193
3 FORN=0TOZ:READA$,B$(N,0):I$(N)=A$+" "+B
  $(N,0):NEXT                          :rem 118
5 PRINT"{CLR}{BLU}{5 DOWN}{5 RIGHT}BAKE A
   CAKE"                               :rem 6
7 PRINT"{2 DOWN}CHOOSE:":PRINT"{DOWN} 1 N
  EED TO KNOW":PRINT"{3 SPACES}WHAT CAN B
  E MADE."                             :rem 77
9 PRINT"{DOWN} 2 WANT TO SEE":PRINT"
  {3 SPACES}A CERTAIN RECIPE":PRINT"
  {DOWN} 3 END PROGRAM"                :rem 79
11 GETE$:IFE$="3"THEN200               :rem 82
13 IFE$="2"THEN61                      :rem 165
15 IFE$<>"1"THEN11                     :rem 222
17 PRINT"{CLR}{BLU}IN THE FOLLOWING LIST,
    PRESS":PRINT"{2 SPACES}'Y' IF YOU HAV
   E THE INGREDIENT"                   :rem 53
19 PRINT"{2 SPACES}'N' IF YOU DO NOT
   {DOWN}":PRINT"{2 SPACES}'S' TO START O
   VER.{2 DOWN}"                       :rem 97
```

```
21 Y=Ø:FORN=ØTOZ:PRINTB$(N,Ø);"{2 SPACES}
   {RED}--{BLK}";:GOSUB15Ø             :rem 58
23 GETE$:IFE$="S"THEN17                :rem 75
25 IFE$="N"THENPRINT"N{BLU}":GOTO31
                                       :rem 114
27 IFE$<>"Y"THEN23                     :rem 12
29 PRINT"Y{BLU}":Y=Y+1                 :rem 57
31 B$(N,1)=E$:NEXTN:C=Ø:PRINT"{2 DOWN}YOU
    CAN MAKE:"                         :rem 113
33 IFB$(1,1)="N"ORB$(2,1)="N"ORB$(5,1)="N
   "THEN37                             :rem 17
35 IFY>7THEN47                         :rem 95
37 PRINT"NOTHING TODAY.":PRINT"YOU NEED M
   ORE SUPPLIES"                       :rem 205
39 PRINT"{2 DOWN}{GRN}PRESS RETURN{BLU}";
   :GOSUB15Ø                           :rem 148
41 GETE$:IFE$=""THEN41                 :rem 245
43 IF ASC(E$)<>13THEN41                :rem 249
45 GOTO5                               :rem 215
47 RESTORE:FORN=ØTOZ*2+1:READE$:NEXT:READ
   A$                                  :rem 19
49 FORN=ØTOZ:READE$:IFE$=""ORE$="Ø"THEN53
                                       :rem 208
51 IFB$(N,1)="N"THENFORI=N+1TOZ:READE$:NE
   XTI:GOTO55                          :rem 199
53 NEXTN:PRINTA$;" CAKE":C=C+1         :rem 13
55 READA$:IFA$<>"Z"THEN49              :rem 205
57 IFC=ØTHEN37                         :rem 68
59 PRINT"{DOWN}GO AHEAD AND BAKE!":GOTO39
                                       :rem 31
61 PRINT"{CLR}{BLU}{DOWN}CHOOSE:{2 DOWN}"
   :PRINT"A BANANA CAKE":PRINT"B CHERRY C
   AKE":PRINT"C CHOCOLATE CAKE"        :rem 59
63 PRINT"D DEVIL'S FOOD CAKE":PRINT"E GOL
   D LAYER CAKE":PRINT "F OATMEAL CAKE"
                                       :rem 23
65 PRINT"G RED VELVET CAKE":PRINT"H SAUER
   KRAUT CAKE":PRINT"I SPICE CAKE"
                                       :rem 146
67 PRINT"J TWO-EGG CAKE":PRINT"K WACKY CA
   KE":PRINT"L WHITE CAKE":GOSUB15Ø
                                       :rem 187
69 GETE$:IFE$=""THEN69                 :rem 9
71 A=ASC(E$):IFA<65ORA>76THEN69        :rem 83
73 RESTORE:PRINT"{CLR}{BLU}";:FORN=ØTO2*Z
   +1:READE$:NEXT                      :rem 79
75 IFA=65THEN79                        :rem 131
77 FORN=1TOA-65:READA$:FORI=ØTOZ:READE$:N
   EXTI,N                              :rem 23
79 READA$:PRINTA$;" CAKE{DOWN}":I=Ø
                                       :rem 177
81 FORN=ØTOZ:READE$:IFE$=""ORVAL(E$)=ØTHE
   N85                                 :rem 193
83 M(I)=VAL(E$):C$(I)=I$(N):PRINTM(I);TAB
   (6);C$(I):I=I+1                     :rem 152
85 NEXTN                               :rem 25Ø
87 PRINT"{2 DOWN}{RED}CONVERT RECIPE? (Y/
   N){BLU}"                            :rem 22Ø
89 GETE$:IFE$="N"THEN39                :rem 86
91 IFE$<>"Y"THEN89                     :rem 25
93 PRINT"{DOWN}MULTIPLY BY WHAT":PRINT"NU
   MBER OR DECIMAL?":PRINT"{RED} --{BLU}"
   ;                                   :rem 27
95 INPUTF:IFF<=ØTHENPRINT"SORRY, F>Ø":GOT
   O93                                 :rem 9Ø
97 F=INT(F*1ØØ)/1ØØ:PRINT"{CLR}{BLU}";F;"
   TIMES ORIGINAL{DOWN}":PRINTA$;" CAKE
   {DOWN}"                             :rem 1Ø1
99 FOR N=Ø TO I-1:PRINT INT(F*M(N)*1ØØ)/1
   ØØ;TAB(6);C$(N):NEXT:GOTO 87        :rem 1Ø9
1Ø1 DATAC.,SHORTENING,C.,FLOUR,C.,SUGAR,C
    .,BROWN SUGAR,TSP.,BAKING PDR,TSP.
                                       :rem 25
1Ø3 DATASALT,TSP.,SODA,C.,CHERRY JUICE,,C
    HERRIES,C.,BANANAS,C.,SAUERKRAUT,C.,M
    ILK                                :rem 188
1Ø5 DATAC.,BUTTERMILK,,EGGS,,EGG WHITES,T
    SP.,RED COLOR,ØZ.,CHOCOLATE,TBSP.,COC
    OA,TSP.                            :rem 187
1Ø7 DATAVANILLA,TSP.,CINNAMON,TSP.,NUTMEG
    ,TSP.,VINEGAR,C.,SALAD OIL,C.,WATER,C
    .,OATMEAL                          :rem 137
1Ø9 DATABANANA,.67,2.5,1.67,,1,1,1,,,1,,,
    .67,2,,,,,,,,,,,,CHERRY,.5,2.25,1.33,
    ,3,.5,,.25                         :rem 161
111 DATA16,,,.5,,,4,,,,,,,,,,CHOCOLATE,.
    67,2.5,1.75,,,.5,1,,,,,,,2,,,2,,1,,,,
    ,1.25,Ø                            :rem 143
113 DATADEVIL'S FOOD,.67,2.25,2,,1,1,1,,,
    ,,1.25,,3,,1,3,,,,,,,,,GOLD LAYER,.5,
    2.25,1.5,,3                        :rem 49
115 DATA1,,,,,,1.67,,2,,,,,1.5,,,,,,,OATM
    EAL,.5,1.5,1,1,,.5,1,,,,,,,2,,,,,1,.7
    5,.25,,,1.25                       :rem 76
117 DATA1,RED VELVET,.5,2.75,1.5,,,.5,1.5
    ,,,,,,1,2,,6,,2,1,,,1,,,,SAUERKRAUT,.
    67,2.25,1.25                       :rem 211
119 DATAØ,1,.25,,,,,,.67,,,3,,,,8,1,,,,,1.
    25,,SPICE,.75,2.25,1,,1,1,1,,,,,,1,3,
    ,,,,,1,.5,,Ø                       :rem 37
121 DATAØ,,TWO-EGG,.5,2.25,1.5,,2.5,1,,,,
    ,,1,,2,,,,,1,,,,,,,WACKY,,2.5,1.5,,,1
    ,1,,,,,,,,Ø                        :rem 142
123 DATAØ,,6,1,,,1.5,.75,1.5,,WHITE,.75,2
    .25,1.5,,3,1,,,,,,1,,,5,,,,1.5,,,,,,,
    Z                                  :rem 1Ø3
15Ø FORI=ØTO9:GETE$:NEXTI:RETURN :rem 11Ø
2ØØ PRINT"{CLR}{BLU}":END              :rem 38
```

VIC/64 Program Lifesaver

(Article on page 132.)

UNNEW

```
1Ø I=525
2Ø READ A:IF A=256 THEN 4Ø
3Ø POKE I,A:I=I+1:GOTO 2Ø
4Ø POKE 43,525 AND 255:POKE 44,2:REM SET
   {SPACE}BOTTOM OF MEMORY
5Ø POKE 45,578 AND 255:POKE 46,2:REM SET
   {SPACE}TOP OF MEMORY
6Ø CLR : SAVE"Ø:UNNEW",8
7Ø REM FOR TAPE USE SAVE"UNNEW",1,1
525 DATA 16Ø,3,2ØØ,177,43,2Ø8,251
532 DATA 2ØØ,2ØØ,152,16Ø,Ø,145,43
539 DATA 165,44,2ØØ,145,43,133,6Ø
546 DATA 16Ø,Ø,132,59,162,Ø,2ØØ
553 DATA 2Ø8,2,23Ø,6Ø,177,59,2Ø8
56Ø DATA 245,232,224,3,2Ø8,242,2ØØ
567 DATA 2Ø8,2,23Ø,6Ø,132,45,164
574 DATA 6Ø,132,46,96,256
```

BEFORE TYPING...

Before typing in programs, please refer to "How To Type COMPUTE!'s Gazette Programs," "A Beginner's Guide To Typing In Programs," and "The Automatic Proofreader" that appear before the Program Listings.

WORDPROCESSING FOR THE VIC-20 AND C-64

Be more creative and efficient by using the *Scribe's* easy-to-learn menu-driven features. Within minutes use:

- Screen edit
- Up to 10 tabs
- Search and replace
- Save/load/merge files
- 1024 color combinations
- Move
- Delete
- Word wrap
- Insert
- Scroll
- Center
- Formatted printing; Margin adjust; right justify; Single, double, or triple space
- With some printers: graphics, underline, and special characters

For a limited time you can obtain the *Scribe* at a *Special Introductory Price:*

VIC-20 (Specify 16K or 24K version) ~~$29.95~~ ***$16.95***
Commodore 64 version ~~$24.95~~ ***$19.95***

Add $3.00 for discs. Specify your type and model printer.

Send a check or money order (including $3.00 for mailing) to:

***Scribe* Associates**
P.O. Box 292647, Dayton, OH 45429

Allow 3 weeks for personal checks
Ohio residents add 6% sales tax

VIC-20 and C-64 are trademarks of Commodore Business Machines, Inc.
Dealer Inquiries Welcomed

LEARN

Having trouble learning to use your computer?

Reference manuals don't teach. Most BASIC texts don't cover specific personal computers.

TIS solves these problems with step-by-step books tailored for your machine.

For PET/CBM

Understanding your PET/CBM $14.95
Vol. 1: Basic Programming
PET Graphics .... $ 4.95

For VIC

Understanding Your VIC $11.95
Vol. 1: Basic Programming

Software for PET, VIC, & C64

Software products on cassette or floppy disk with complete instruction manual.
Cassette $24.95
Diskette $29.95

SW-1 MAIL 8 mailing list system
SW-2 CHECKBOOK record
SW-3 ACCOUNTS keep track of who owes you how much
SW-4MEDIT create and maintain data files
SW-5 CALENDAR appointments, meetings at-a-glance

Money Back Guarantee. VISA/MC accepted.
All prices include UPS or 1st Class postage.

TIS INC

Total Information Services, Inc.
Box 921, Dept. G
Los Alamos, NM 87544

NEW FOR VIC 20

16k with Battery Backup $89

NO MORE DATA LOSS!

If power fails — data and programs are retained for up to 2 weeks!

No other 16k expansion can make this claim!

PLUS:

- FULL BLOCK SWITCHING
- WRITE PROTECT
- RESET SWITCH, AND MORE

FULLY GUARANTEED

To order, or for free detailed application literature, write to:

ABARIS, INC.
PO Box 2501
Vancouver, WA 98668

Add $3 shipping

Washington residents, add 6% sales tax.

VIC & 64

AT LAST! The information you need, without going back to back to the manual. These durable plastic coated overlays contain program starting locations, function key labeling, commands and additional aids in center cutout.

Please send me the following Leroy's Cheatsheet™ keyboard overlays

20	64		20	64	
☐		Programmer's Aid	☐	☐	Graphic printer (1515 & 1525)
☐		Vicmon	☐		UMI Wordcraft 20
☐		Super Expander	☐		HES Vic Forth
☐		Vic Typewriter	☐	☐	HES Writer
☐		Victerm 1		☐	Wordpro 3 plus
	☐	Term 64		☐	Easy Script
☐	☐	Quick Brown Fox	☐	☐	Basic

Send check or money order plus $2.00 (postage and handling)
PA residents add 6% sales tax

Name ____________

Address ____________

City ____________ State ____________ Zip ____________

CHEATSHEET PRODUCTS™ CG1183
P.O Box 8209 Pittsburgh PA 15218

YOUR NIFTY LITTLE C-64 JUST BECAME A GRAPHICS POWERHOUSE

PROSCREEN 64

Finally! Unlock the powerful hi-res and multi-color hi-res graphics potential of your C-64. Now you can create professional graphics for games, graphs, or just for fun!

- Use all of Commodore's characters and/or characters of your own design. Use both C-64 fonts simultaneously.
- Characters may be normal or reverse, positive or negative, upright or rotated. All combinations may be used.
- Place characters anywhere on the screen, not just in the usual character cells. Overlap or erase without changing the background!
- Call hi-res routines from your BASIC program or create graphics in the immediate mode. Switch at will from hi-res screen to standard screen and back.
- Draw or erase with a joystick using any character as the stylus.
- Lines, circles, and arcs drawn automatically. PROSCREEN spline connects discrete dots with a smooth line!
- Machine code program uses not a single byte of user BASIC RAM.

If you're not convinced, send $5.00 (refundable upon Purchase) for users' guide to get the inside scoop. If you're sold, $34.95 gets the whole thing on tape, $37.95 on disk. Check or M.O. to

WETHERBEE SOFTWARE
P.O. Box 4512, Ann Arbor, MI 48106

RAINBOW SALES & SERVICES of HOUSTON

10 for 10 **for COMMODORE 64**

- 10 PROGRAMS FOR $10.00, POSTAGE PAID ANYWHERE ON EARTH.
- PROGRAMS CAN BE USED AS POINT OF PURCHASE DISPLAY ADVERTISING AS THEY SHOW COMMODORE 64's AUDIO/VIDEO CAPABILITIES BLENDED INTO UNIQUELY PERPETUAL PRESENTATIONS OF SOUND & COLOR.
- A BARGAIN FOR COMPUTER ARTWORK APPLICATIONS AS PROGRAMS EDUCATE USER AND EXPLICITLY SHOW ALL POSSIBLE COMBINATIONS OF CHARACTER, COLOR, VOICE AND SPRITE CAPABILITIES IN A RANDOM GENERATED FORMAT FOR YOUR COSMIC IMPLICATIONS...
- AVAILABLE IN CASSETTE ONLY.

*** ***$10.00*** ***

SEND CHECK OR MONEY ORDER.
RAINBOW SALES & SERVICES
P.O. BOX 772253
HOUSTON, TEXAS 77215

3-D OTHELLO

OJAI SOFTWARE

Offers for the unexpanded

VIC-20

An added dimension to the game of Othello.

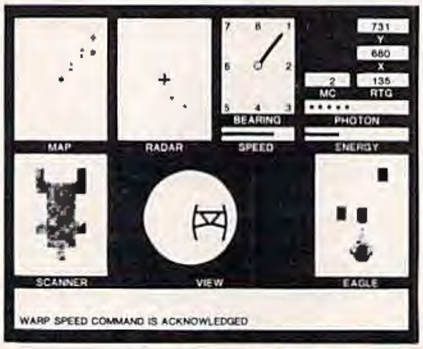
Play against friend or computer

$15 per cassette
(Joystick required)

Dealer inquiry invited.

Calif. Res. please add 6% sales tax.

OJAI SOFTWARE
P.O. BOX 1860
OJAI, CALIF. 93023



QUEST OF THE EAGLE

YOUR MISSION THIS TIME...SECURE PLUTONIUM AND DELIVER TO PLANET JEMSIO...ENERGY LOW... RADAR SHOWS CRAFT APPROACHING...TRANSPORT ROOM TOO DAMAGED TO BEAM UP ENERGY...IS CRAFT FRIEND OR FOE?...

CAN YOU COMPLETE THIS MISSION AND ADVANCE TO MORE CHALLENGING MISSIONS OR WILL THEY DESTROY YOU?

COMMODORE 64 | **FERIN ENTERPRISES, 6310 UNDERWOOD AVE. S.W., CEDAR RAPIDS IA. 52404** | **$20.75 SPECIFY CASSETTE OR DISKETTE**

GET THE MOST FROM YOUR VIC-20/C64

CASSETTE INTERFACE

- USE ANY STANDARD CASSETTE RECORDER
- CONNECTS TO THE CASSETTE PORT
- CONTROLS THE CASSETTE MOTOR
- NEEDS NO BATTERIES
- COPY FROM RECORDER TO RECORDER
- THIS IS THE BEST ONE FOR LESS $$$
- ONLY $34.95 POSTPAID. ADD $2.50 FOR SHIPPING OUTSIDE US, CANADA, MEXICO

THE MODEM INTERFACE

- CONNECTS TO THE USER I/O PORT
- NEEDS NO BATTERIES/SOLDERING
- CONNECTS ANY STANDARD MODEM
- USE MODEM AUTODIAL ANSWER FEATURES
- COMES WITH A FREE TYPE IN BASIC TERMINAL PROGRAM
- ONLY $24.95 POSTPAID. ADD $2.50 FOR SHIPPING OUTSIDE US, CANADA, MEXICO

CHARGE OR COD ORDERS CALL
1-800-227-3800
ASK FOR OPERATOR 225

BYTESIZE MICRO TECHNOLOGY PO BOX 12309 DEPT HD SEATTLE, WA 98111 (206) 236-BYTE

CALL OR WRITE FOR DEALER INFORMATION

Writers, programmers... *COMPUTE!'s Gazette* is looking for articles on the VIC and 64.

Advertisers Index

New From Cardco

Five Slot Expansion Interface for the C-64

The CARDBOARD/5 (CB/5) is an enclosed five slot, fully switch selectable, expansion interface for the Commodore 64™. This quality product allows the user to switch select any cartridge slot or combination of cartridge slots. Twenty-two color coded light emitting diodes give status indication. Each slot has four LEDs and two toggle switches for indication and control. Two master toggle switches allow the user to manually override any situation.

All Cardco products are individually tested to insure quality and reliability.

Some of the features of the CARDBOARD/5 are:

- high quality glass/epoxy circuit board
- gold plated contacts
- logic lines are switched by solid state IC switches
- full LED status indication
- convenient toggle switches
- full support under the board to prevent flexing
- full plastic enclosure to insure safety
- fused to protect your computer
- convenient reset button
- CARDCO, Inc.'s exclusive Lifetime Guarantee

See a complete line of American made Cardco Products at a computer store near you, today.

313 Mathewson • Wichita, Kansas 67214 • (316) 267-6525

cardco, inc.