YOU CAN MAKE A FORTUNE IN MICROS!

THE MICRO COMPUTER BUSINESS WILL GROW FROM $10 TO $100 BILLION IN THE NEXT EIGHT YEARS! ARE YOU READY TO CASH IN?

The **micro computer business** is predicted to grow from its present $10 billion to **$100 billion** before 1990! Imagine the possibilities this opens for you! No matter where you live, if you're starting up or presently in business, **no other industry offers you more opportunities!**

Now, finally, all the inside information you need to secure a prosperous future in this dynamic industry is available in **one place - THE COMPUTER ENTREPRENEUR MANUAL!** – An **immense** information source, compiled by our **inquisitive** research team, aided by a panel of **experts** and **business people** from **all areas** of the **computer industry!**

We present the inside story of **more than 100 lucrative computer businesses you can enter,** where you'll find the **real opportunities** for the eighties: from one man operations like **Programming Author, Word Processing Center** or **Consulting,** to **Systems House, Service Bureau, Computer Store etc!** Many at little or no investment! All the invaluable facts and figures: **How to start, Capital needs, Profit estimates and Margins, How to Sell and Market, How missing** technical or business experience **need not stand in your way, Source of Suppliers, etc!** Details that could take **years** to find out on your own!

We'll show you **inside tricks,** like how to **never again pay retail** for computer products and consumer electronics, **even for one item - right now,** while you're starting your business! How to get **free merchandise** and trade show **invitations, etc. This alone** will **more than pay** for the **manual!** You'll read **actual case histories** of other computer entrepreneurs, so you can **learn** from **their mistakes,** and **profit** from their **success stories! Where** you'll be **one year** from now depends on **your actions today!** Let us **show you how** to take the **first crucial steps!**

Order now and take **advantage** of our **limited introduction special, THE COMPUTER ENTREPRENEUR MANUAL,** and a **six month subscription to THE COMPUTER ENTREPRENEUR REPORT/NEWSLETTER** (so you're **always up-to-date** with the industry), **both** for **only $29.95! You must** be **convinced** on **how easy** you can **strike it rich** in the micro computer business - or you may **return** the manual for **a full refund** within thirty days! **USE OUR TOLL FREE NUMBER TO ORDER!**

EVERYTHING YOU NEED TO KNOW TO SUCCEED IN THE COMPUTER BUSINESS IS ALL IN THIS MANUAL!

THE COMPUTER ENTREPRENEUR MANUAL has the **answers** to **all your questions** about **selecting, starting** and **succesfully running a computer business!** There has **never** been such a **comprehensive collection** of **know-how** and **information** about this business **in one place! All the facts you need** to **plan** and **acheive your goals** in **easy-to-follow, step-by-step instructions!**

These are some of the **100-plus businesses** covered in **PART ONE** of the manual, with the facts on **How to start and run, Start-up Cost (Even** how to operate **on a shoestring**), What **profits to expect, Wholesale prices, Mark-ups, Suppliers, future outlook, case histories** for **each, etc:**

Systems House, Software Author (who to sell to and who to avoid), **Service Bureau, Software Publisher** (How to find programs that sell, **Word Processing Service, Consulting** and **Consultant Broker** (use your skills or those of others, **make $150 - $1000 a day!**), **The incredible Games Business, Computer Store** (Franchises: Pro and Contra, or a low inventory store **in your home!), OEM, Hardware Mfg, Data base** and **Teletext Service (big prospects!), Used Computers, Repairs, Rent-A-Computer, Promote Fests** and **Trade Shows, Turnkey Systems, Bartering, Mail Order, Compile and rent mailing lists, Specialized Data Headhunting** and **Temp Help Service, Tech Writer Shop, Custom Engineering,** The **highly profitable Seminars** and **Training Business , and many more!**

Many **new ideas** and **ground floor opportunities! Interviews** and **success stories** on **companies** of **all sizes! Privy info on the profits made:** How some **computer store** operators **net $100 - $250,000!** Little known outfits that made their owners **millionaires,** one of these low-key companies, making simple boards, went from **nil to $20,000,000** and **100 employees** in **four years!** Programmers that make **$300,000, Thousands** of **micro millionaires** in the making, etc!

Whatever your goal is - **Silicon Valley Tycoon,** or just a **business at home - we guarantee** you'll find a **business** to **suit you - or your money back!**

PART TWO of the manual is **loaded** with the **know-how** and **"streetfighting" savvy** you need, both as a **novice** or business **veteran,** to get started, to stay and to **prosper** in the micro computer business! **A goldmine of information** in clear and easy-to-use instructions: How to prepare your **Business Plan, Outside financing,** The **mistakes** you **must avoid,** How to **hire** and **manage employees, Incorporation** (when, and how to do it **cheaply**), **Surviving bad times, Record Keeping,** how to **estimate your market before you start,** Use **multiple locations** to **maximize profits,** how to **promote** and **stay** steps **ahead of the competition!** How to get **free advertising, free merchandise, free advice, Power negotiating** with suppliers to **double** your **profit margins, etc!** Even how to **keep a present job while starting a business part time!**

Don't miss this opportunity to be part of this **great industry - the next success story** could be **your own! Order the manual today!** Part one and two, bound in **a deluxe ring binder,** where you can also **collect our newsletter (free for six months with the manual - a $32.50 value!) - all for only $29.95!**

THE COMPUTER ENTREPRENEUR NEWSLETTER – ALL THE LATEST INSIDE BUSINESS NEWS! NOW! SIX MONTHS FREE WITH YOUR MANUAL!

You're **always attuned** to the industry, and **your manual kept up-to-date,** with our **newsletter!** Each issue has the **latest business news, ideas, new suppliers,** our **indispensible "watchdog" column** on **profits, discounts** (don't miss mfg's promos, like recently, when top video monitor sold at $80 - that's half wholesale, **one third of the retail price!), the competition, the big deals, etc! Feature stories** with start-up info and case histories **on new micro businesses!**

You'll get **invitations** to trade shows and conventions, the **usage** of our **advisory service** and our **discount buying service** for your purchases!

You'll find **many items** in our newsletter that will **save you the cost of your manual many times over!**

CALL TOLL FREE! CHARGE IT!
Credit Card Orders (MC, VISA only) accepted 24 hours/day
1-800-227-3800
Ask for extension 1135

MasterCard VISA

Order by phone (Credit cards only), or use the coupon:

Mail to **THE COMPUTER ENTREPRENEUR PUBLISHING CO. PO BOX 456, Grand Central Station, New York, N.Y. 10163**

Please send me **THE COMPUTER ENTREPRENEUR MANUAL,** and the **six month free subscription** to **THE COMPUTER ENTREPRENEUR REPORT/NEWSLETTER. All for only $29.95,** plus $3 for postage/handling **(NY residents:** add $2.64 for sales tax). If I decide not to keep the manual, I may **return** it **within 30 days** for **a full refund.**

NAME: _____

ADDRESS: _____

CITY, STATE, ZIP: _____

☐ Check or M.O. enclosed Charge to ☐ VISA ☐ MC

CARD# _____

Exp. Date: _____

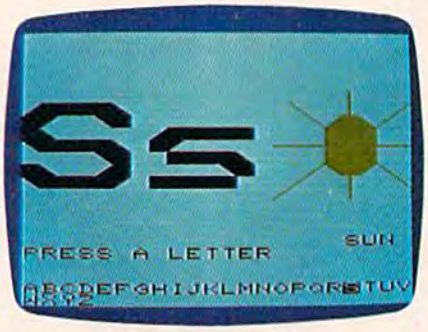
SIGNATURE: _____

GA184

REVIEWS

Children's Educational Games

Tony Roberts, Assistant Managing Editor



ALPHA-BECi *uses simple graphics to teach letters of the alphabet.*

The goal of Boston Educational Computing is to provide owners of the most elementary computer systems with educational software that can be used easily by those with little knowledge of computing.

In its Child Development Series, BECi (pronounced Becky) meets this goal. Among the software in the series are a counting program and an alphabet program aimed at preschool children and an addition-subtraction program targeted for the slightly older child.

All of the programs are on tape and are designed to work on an unexpanded VIC-20, but they perform just as well with a memory expander. *NUMER-BECi* and *ADD/SUB* are also available for the Commodore 64 and Atari computers.

Introducing The Alphabet

ALPHA-BECi is intended to help preschool children learn the letters of the alphabet, both upper- and lowercase, associate each letter with a word beginning with that letter, and see how each letter relates to the others in the alphabet.

For each letter, the program provides a screen which includes the capital letter, the small letter, and a picture of an object. The screen is slowly drawn, giving a child the opportunity to call out the name of the letter or object as soon as he recognizes it.

First, the capital letter is drawn, then the associated small letter, then the object with its name below. The entire alphabet is then printed at the bottom of the screen with the target letter highlighted.

ALPHA-BECi can be run in one of three ways. From a menu you decide whether to run the program sequentially, randomly, or under keyboard control. That is, the program will either step through the alphabet from A to Z, will display random letters, or will show screens for the letters selected by the user. The mode can be changed at any time by pressing RUN/STOP, then rerunning the program.

For a child, watching the colorful objects appear on screen is like opening a present. The program's only sound effect is a boop-boop-boop that comes as the letters of the alphabet are being printed along the bottom of the screen. For an adult, it may become a little annoying, but it seems to be music to a child's ears.

Shapes And Colors

NUMER-BECi teaches the preschooler to identify the numbers from 1 to 12, to identify shapes, and to identify colors. And it is set up so the youngster can learn these new ideas at his own pace.

The menu offers the following options:

1. COUNT LIKE THINGS
2. COUNT UNLIKE THINGS
3. COUNT COLORS
4. COUNT SHAPES
5. COUNT COLORS AND SHAPES

A second menu allows you to select a time limit for the answers. Your options here are to let the program run itself, filling in the answers after a specific time delay; have it wait until the child fills in an answer; or have it set a time limit.

When running the program with a time limit set, a correct answer will shorten the time allowed for the next problem. As long as the child continues providing correct answers, the time limit is shortened until he misses. Then the time limit is

Write For FREE Catalog

VISA

(602) 855-3357

Call your order in and get a FREE program

NEW VIC SOFTWARE CBM 64

Call your order in and get a FREE program

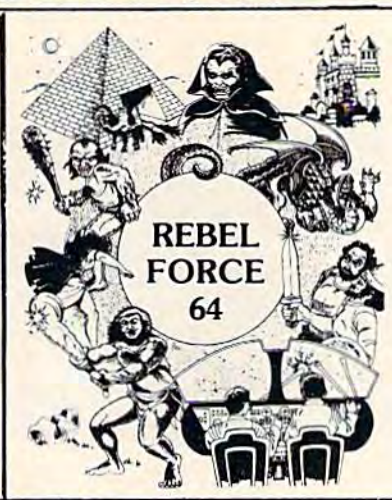
Write For FREE Catalog

MasterCard

(602) 855-3357

CRICKET

Software from ComputerMat will turn your VIC or 64 into a home arcade.



PROGRAMMERS CONTEST

Send Us Your Best VIC or 64 Program

Winners will receive royalties plus $1,000 in prizes.

BUG BLAST

PARATROOPER

POP TOP

MUSIC MAKER 64

TARGET COMMAND

MOW

COSMIC CRUZER

We have more games and programs than you can shake a joystick at!

SHOGUN 64-VIC

CHRISTMAS SPECIALS

Buy 2 Programs and Get 1 Free!

For every 2 VIC or 64 programs you buy, we will send you a coupon for a third program free!

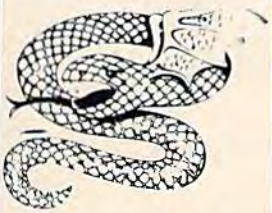
SPACE PAK

ALIEN INVASION

64 ARCADE PAK

STOMPERS - 64

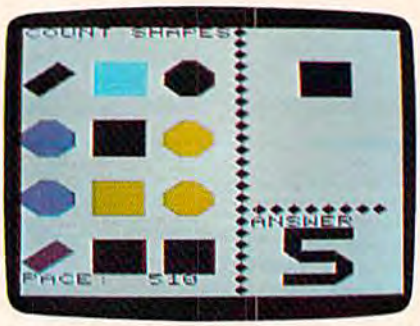
SUPER PAK - 64



SNAKE OUT

HEAD ON

ComputerMat • Box 1664 M • Lake Havasu City, AZ 86403 *Add $1.00 For Shipping*



NUMER-BECi *teaches youngsters to recognize and count various colored shapes.*

The top arrow in ADD/SUB *reminds youngsters when to carry or borrow.*

incremented slightly. This feature challenges a child to increase the speed at which he inputs his answers.

When the program is run in the wait-for-answer mode, the child is simply asked to count the number of shapes printed on the screen. If the answer is correct, he moves on to the next problem. If the answer is wrong, the word WHOOPS! is printed in red and the correct answer is displayed.

Under the Count Colors option, the program prints shapes of different colors on the screen and the child is shown a color block and asked to count the number of shapes that match the color block. Count Shapes is a similar exercise with shapes. Shapes and colors are mixed in the final exercise. The child must count the number of black squares, or red triangles, for example.

The child's answer to each problem is printed in large block letters. It takes a second or two to appear for the answer to be printed on the screen after it has been typed in, and this can be a bit disconcerting if you're used to seeing what you type appear on the screen instantly.

Putting 2 And 2 Together

ADD/SUB is BECi's program for older children. The program, which is available for the Commodore 64 as well as the VIC-20, offers choices of addition or subtraction, one to four digits, and decimal or whole numbers. In addition, the user can decide whether to include problems that require carrying or borrowing. For those just learning about carrying and borrowing, there is an option that provides a hint in the form of a large arrow pointing to the column from which a borrow has been made or to which a carry must go.

With *ADD/SUB* you also can select a pace for solving the problems. The standard pace gives the child about seven seconds to answer. When the time is up, the computer will fill in the answer. If the wait-for-answer option is selected, the computer will wait until the numbers are filled in. The set pace option can be used to speed up or slow down the pace to meet a child's needs.

In *ADD/SUB* the problems are displayed in large black numbers with a green plus or minus sign. In multidigit problems, an arrow points to the first digits to be added or subtracted. The answers are filled in from right-to-left, just as they would be done on paper.

Using The Programs

The programs are, as advertised, easy to use. Simply load them from tape and type RUN. Options for play are presented in simple menus. To change the mode of play, press the STOP key, type RUN, and select a new option.

Loading the programs takes a few minutes; a preschooler with a short attention span might spend less time using the program than it takes to load.

The programs are low key. No scores are kept, and the rewards for correct answers are understated. The adjustable nature of the programs and the pacing options, however, provide these programs with an extended life. They are entertaining to a child just beginning to grasp the concepts covered, and they can challenge the more advanced child to solve the problems more quickly.

ALPHA-BECi
NUMER-BECi
ADD/SUB
Boston Educational Computing Inc.
78 Dartmouth Street
Boston, MA 02116
(617) 536-5116
$16.95 each

COMPUTE!'s Gazette Back Issues

JULY 1983: Commodore 64 Video Update, Snake Escape, Alfabug, VIC Marquee, Word Hunt, VIC Timepiece, product reviews, Learning To Program In BASIC, Quickfind, 64 Paddle Reader, Machine Language For Beginners, Enlivening Programs With Sound, Using Joysticks On The 64, Simple Answers To Common Questions, VICreations — Speedy Variables, 64 Explorer.

AUGUST 1983: Your First Hour With A Computer, Should You Join A Users Group, Guide To Commodore Users Groups, The Viper, Cylon Zap, product reviews, VIC/64 Mailing List, Word Spell, Global Scan For VIC/64, Machine Language For Beginners, VIC Title Screens, 64 Hi-Res Graphics Made Easy, VIC/64 Four-Speed Brake, Disk Menu, Using A 1540 Disk Drive With The 64, Playing Computer Music, Simple Answers To Common Questions, HOTWARE, VICreations — Caring For Disk Drives/Cassettes, 64 Explorer, News & Products.

SEPTEMBER 1983: Telecomputing Today, Telecomputing Glossary, Commodore's Nationwide Party Line, Commodore Bulletin Boards, Demon Star For VIC/64, Potholes, product reviews, Checkbook Reporter, States & Capitals Tutor For VIC/ 64, MiniTerm-20, TeleTerm 64, POKEing Graphics, Machine Language For Beginners, 64 Searcher, Better Commodore Input, Using The Function Keys, Simple Answers To Common Questions, HOTWARE, VICreations — Understanding Random Numbers.

OCTOBER 1983: The Anatomy of Computers, Telegaming Today And Tomorrow, Commodore's Public Domain Programs, Oil Tycoon, Re-Beep, product reviews, Aardvark Attack, Word Match, A SHIFTy Solution: The WAIT Command, Program Transfers, Machine Language For Beginners, Improved Paddle Reader Routine, How To Use Tape And Disk Files, Understanding 64 Sound — Part 1, Speeding Up The VIC, Simple Answers To Common Questions, HOTWARE, Horizons 64 — Improving 64 Video Quality, VICreations — Using The VIC's Clock, News & Products.

NOVEMBER 1983: Binary Numbers — Part 1, Getting Started With A Disk Drive — Part 1, Chicken Little, Martian Prisoner, product reviews, Munchmath, VIC Super Expander Graphics, 64 Aardvark Attack, 64 Timepiece, Connect The Dots, Custom Characters For VIC/64, Making Custom Characters On The 64, Making Custom Characters On The VIC, VIC/64 Program Lifesaver, Understanding 64 Sound — Part 2, Merging Programs On The 64, Tutorial On DATA, READ, RESTORE Statements, One-Touch Commands For The 64, VIC/64 Disk Defaulter, Machine Language For Beginners, Simple Answers To Common Questions, HOTWARE, VICreations — Animation With Custom Characters, Horizons 64 — Software And Hardware Reviews, News & Products, Automatic Proofreader.

Back issues of July, August, and September 1983 are $2.50 each. Issues from October forward are $3. Bulk rates are 6 issues for $15 or 12 issues for $30. All prices include freight in the U.S. Outside the U.S. add $1 per magazine order for surface postage. $4 per magazine for air mail postage. ALL BACK ISSUES ARE SUBJECT TO AVAILABILITY.

In the Continental U.S. call
TOLL FREE 800-334-0868
(in North Carolina call 919-275-9809)

Or write to:
COMPUTE!'s Gazette for Commodore Back Issues
P.O. Box 5406
Greensboro, North Carolina, 27403, USA

Prepayment required in U.S. funds. MasterCard, VISA, and American Express accepted. North Carolina residents please add 4% sales tax.

Computer Baseball And *Ringside Seat* For Commodore 64

Gregg Keizer, Assistant Book Editor

Ever wanted to manage your own major league team? Most people who follow baseball closely just *know* that they could take their team to the World Series, if only *they* were manager. *Computer Baseball* can give you the feeling of the close game, a pennant race, even the final game of the Series. And it costs far less than buying the Minnesota Twins, the latest team up for sale. You won't even have to pay players' salaries.

This isn't an arcade baseball game like others you may have played. Although players move and hits cross the field, the animation is minimal. *Computer Baseball* is a simulation of the strategy and tactics of a major league team. Using actual statistics, opposing managers can select lineups, choose starting and relief pitchers, set up defense for the bunt or double play, and signal runners to steal. When you put on the manager's cap, *you* make the decisions for your team, hoping to outwit the opposing manager's strategy.

The game includes 28 team rosters, World Series teams all, which you can use to play another opponent, or even the computer manager, Casey. You can even enter data for new teams yourself, or send for a disk containing the most recent American and National League teams, so you can replay an entire season if you want.

How To Play

After you've booted the disk for *Computer Baseball*, you'll be offered several choices. Do you' want to play a two-player game, play against the computer, enter new player data, or watch a demonstration game? Playing against Casey, the computer manager, is a good way to learn the game. Choosing this option presents more decisions. Do you want Casey to manage the home team, the visiting team, or both? I let Casey have the visiting team. But which team?

You'll see 28 teams listed on the screen, ranging from the '06 White Sox to the '81 Dodgers. All 28 played in a World Series, so you can replay a complete seven-game Series, or you can play a What If. What if the '27 Yankees, with Babe Ruth, could have met the '81 Yankees? Who was the better team? Has baseball gotten better, or worse? Once you select Casey's team, as well as your own, the computer will display the team lineups.

Now the decisions become more important. You can choose the starting pitcher(s) and set the batting order for your team. If you want, you can set the order for Casey's team too, but I let him do that. Although setting your batting order takes time, it adds to the game's realism. You have to fill each of the fielding positions, and you only have so many players. You could force a catcher to play outfield, but it's probably not a good idea. As you enter the order, take your time, for any errors you make means you'll have to go through it all again. Unfortunately, there's no option to change your mind in this section of the game.

The screen display appears once you've chosen your team and selected the batting order. The display shows a playing field, players, a scoreboard, and pitcher and batter status information. Once the game starts, you'll be able to tell if the batter is right- or left-handed, where the defense is playing, and the base-running abilities of men on base.

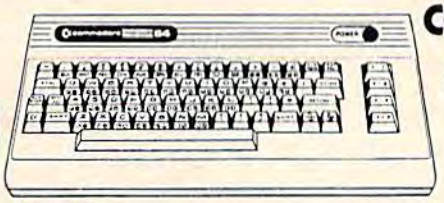
Whether your team is on the field or up to bat, you have several decisions to make. I took the field first, being the home team, so I had to choose my defensive alignment. You can play your outfielders shallow, or in their normal positions. Your infielders you can place even more carefully, moving them in, guarding the lines or setting up for the double play. You can hold base runners, if there are any, tight or loosely, depending on the situation. You can even visit the mound and talk to your pitcher, check to see if he's tired, and perhaps bring in a reliever. Make sure that your reliever is warmed up, though, or he could easily be hit off of.

When your team is up to bat, you have fewer choices to make. You can hit away, hit and

COMPUTER MAIL ORDER

commodore

VIC-20 $89



COMMODORE 64. $219

1520 Color Printer/Plotter	$169.00
1525 80 Column Printer	$219.00
1530 Datasette	$69.00
1541 Single Disk Drive	$249.00
1600 VIC Modem	$59.00
1610 VIC Term 40	$49.00
1650 AD/AA Modem	$89.00
1702 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

PRINTERS

Epson (MX100, RX80, FX80, FX100)	CALL
Okidata (82, 83, 84, 92, 93)	CALL
Star Gemini 10X	$299.00
Star Gemini Delta 10	$559.00
Smith Corona TP-2	$469.00
C.Itoh Prowriter 8510P	$379.00
C.Itoh Gorilla	$209.00

CARDCO.

Light Pen	$32.00
Cassette Inferface	$29.00
Parallel Printer Interface	$69.00
3 Slot VIC Expansion Interface	$32.00
6 Slot VIC Expansion Interface	$79.00

SOFTWARE

commodore

CBM 64 Reference Guide	$18.00
C-64 DISKS	
EasyCalc	$65.00
EasyFinance I, II, III, IV	$19.00
EasyMail	$19.00
EasyScript	$39.00
Word/Name Machine	$19.00
EasySpell	$19.00
Accounts Receivable	$39.00
Accounts Payable	$39.00
General Ledger	$39.00
Assembler	$19.00
Logo	$39.00
Pilot	$39.00
Pet Emulator	$19.00
Screen Editor	$19.00
Music Machine	$15.00
Music Composer	$15.00
VIC 20 CARTRIDGES & DISKS	
Gortek & the Micro Chips (D)	$19.00
Super Slot (R)	$14.00
Super Alien (R)	$14.00
Jupiter Lander (R)	$14.00
Radar Rat Race (R)	$14.00
Count Adventure (R)	$21.00
Pinball Spectacular (R)	$19.00
VIC Reference Guide	$15.00

ARTWORX

C-64/VIC 20 CASSETTES	
Bridge 3.0	$15.00
Teacher's Pet	$12.00

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
Centicod	$23.00
Frogee	$23.00
C-64 CASSETTES	
Road Toad	$24.00

CREATIVE SOFTWARE

C-64 CASSETTES	
Home Inventory	$11.00
Household Finance	$23.00
C-64 CARTRIDGES	
Trashman	$29.00
C-64 DISKS	
Home Inventory	$11.00
Household Finance	$29.00
VIC 20 CASSETTES	
Home Inventory	$12.00
Household Finance	$14.00
VIC 20 CARTRIDGES	
Astro Blitz	$21.00
Black Hole	$32.00
Trashman	$21.00
Choplifter	$21.00

DYNATECH

C-64 DISK	
Codewriter	$75.00

EPYX

VIC 20 CASSETTES	
Ricochet	$32.00
Sword of Fargoal	$24.00
Rescue at Rigel	$24.00

QUICK BROWN FOX

C-64/VIC 20 CARTRIDGE	
Word Processor	$49.00

K-BYTE

VIC 20 CARTRIDGES	
K-Razy Antics	$29.00
K-Star Patrol	$29.00

UMI

VIC 20 CARTRIDGES	
Amok	$30.00
Meteor Run	$40.00
Alien Blitz	$30.00
VIC 20 CASSETTES	
Cloud Burst	$15.00
Video Verman	$19.00

DUST COVERS

C-64 Cover	$9.99
VIC 20 Cover	$9.99

HES

VIC 20 CASSETTES	
Torg	$7.99
Skier	$14.95
Tank Wars	$14.95
Pak Bomber	$12.95
Maze of Mikor	$14.95
Laser Blitz	$14.95
Tank Trap	$14.95
VIC 20 CARTRIDGES	
VIC Forth	$32.00
HES Mon	$29.00
HES Writer	$29.00
Aggressor	$29.00
Synthesound	$21.00
Shamus	$29.00
Protector	$29.00
Turtle Graphics	$29.00
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 CASSETTE	
3-D Man	$16.00

ROMOX

Typo (20/64)	$29.00

RAINBOW

C-64 DISKS	
Personal Finance	$48.00
Writers Assistant	$95.00
Spreadsheet Assistant	$95.00

QUANTUM

40/80 Column Video Board	$95.00
40/80 Col. Video Board (16K)	$179.00

SIRIUS

VIC 20 CARTRIDGES	
Deadly Duck	$21.00
Spider City	$29.00

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
Payroll	$79.00
Data Base	$69.00
CalcResult	$139.00
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	$59.00

We carry a selection from the above manufacturers plus...

Synapse, Thorn, InfoCom, Onslow, Practicalc, Spinnaker & Timeworks.

WEST

1-800-648-3311

In NV call (702)588-5654, Dept. 0113
P.O. Box 6689, Stateline, NV 89449
Order Status #: 588-5654

CANADA

1-800-268-4559

In Toronto call (416)828-0866, Dept. 0113
2505 Dunwin Ct., Unit 1B,
Mississauga, Ontario, Canada L5LIT1
Order Status #: 828-0866

EAST

1-800-233-8950

In PA call (717)327-9575, Dept. 0113
477 E. Third St. Williamsport, PA 17701
Order Status #: 327-9576

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 $5.00) 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 third party software for most all computers on the market. Call today for our new catalog.

CANADIAN ORDERS: All prices are subject to shipping, tax and currency exchange fluctuations. Call for exa[illegible]

run, bunt, steal, or signal your runners to edge off the bag a bit more. If you want, you can send in a pinch runner or hitter.

Each pitcher-batter confrontation is displayed on the screen by a single pitch. There are no balls and strikes called, although batters can be struck out. A message appears at the bottom of the screen after each pitch, telling you what's happened. Pop flies, grounders, and base hits are shown to you this way. Runners advance, are thrown out, or score. Although only nine innings are displayed on the screen, extra innings are possible.

Managing It All

Computer Baseball is not for the impatient. It's not a fast-moving, arcade-style game, but it's not meant to be. Instead, the game is for those who love baseball, who have always wanted to step into a manager's cleats. As a simulation, it gives you the feeling of managing a team. Just as in reality, once the basic decisions are made, the players run, hit, and throw in their own ways. A major league manager cannot hit for a player, and neither can you in this game. The statistics of each player determine that.

Some games are pitchers' battles, with low scores. Other games display hitters' powers, and the scores run up quickly. No two games are the same. Again, that reflects the simulation's excellence. As the innings pass, your decisions on pitching and running change, just as in a real game. Strategy is vital. Outguessing the opposing manager is just as important.

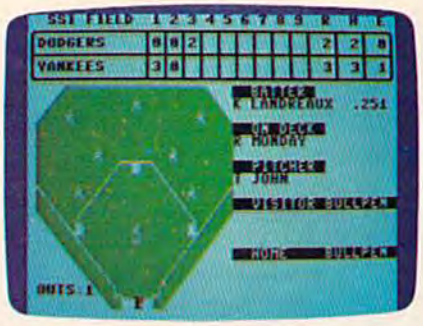
If you enjoy baseball, you'll enjoy this game. The computer takes all the routine drudgery out of keeping track of statistics and lets you concentrate on decision-making. You'll think you've paced and worried in the dugout just like a major league manager.

Ringside Seat

In many ways like *Computer Baseball*, *Ringside Seat* is a game of strategy and statistics. But instead of managing a major league baseball team, you play the part of a boxer's manager. You're not the fighter, but his manager, telling him how to fight when he's in the corner between rounds. Once the bell rings, he boxes in his own way, simply following your strategic suggestions.

When the game is loaded from disk, you have the option of managing either, both, or neither of the fighters. If you want, you can also act as the third judge in the fight, or let the computer handle it. Then you select the two boxers.

The game includes a variety of boxers available to you. Divided into weight classifications, from bantamweight to heavyweight, you simply enter the fighters' names and weight divisions. To see a complete list of the fighters on the game disk, you should press *E* the first time the computer asks you to enter a choice. After a short delay, you should press *L* to see the fighter lists, and then the weight classification. The screen will then show the fighter's identification number, his name, rating, style (slugger or boxer), and his weight division. The lists contain



The Yankees and Dodgers face off in a demo game of Computer Baseball.

The famous Dempsey-Tunney heavyweight title fight is reenacted with Ringside Seat.

current fighters, as well as famous boxers from the past. If you wanted to see how Tunney would have matched up against Foster, for example, you can create this What If scenario.

Once you've chosen the fighters and started the game, you'll see a display on the screen. The boxing ring is shown, as well as representations of the fighters, each still in his corner. Other information, such as the fighters' names, the strategy picked by each manager, condition of the fighter, from cuts to stamina remaining, and even the cumulative judges' score will be displayed. At the top of the screen you'll see descriptions of the fighters' movements during a round. Near the bottom of the screen you'll see messages displayed for the color commentary and blow-by-blow descriptions of the fight.

As in a real fight, you decide how you want your fighter to box that round. The options range from fighting flatfooted, which lets him rest in a round, saving his strength, to going for the knockout. Your fighter's abilities in each of the strategies are listed at the bottom of the screen before the round starts. Some fighters are better at charging in, while others are more effective in the stick and move. Choosing your fighter's style for that round is the most important part of managing. You have to use your fighter wisely, not overworking him, for each strategy reduces the boxer's stamina level. A more aggressive style, such as charging in, or sticking and moving, uses up more stamina than a

defensive posture, like staying away or protecting a cut.

Once the round begins, the fighter is on his own. You'll see a blow-by-blow description of the fight at the bottom of the screen, as well as see the animated figures move in the ring. The only thing you can do once the bell rings is to tell your fighter to cover up. This is helpful if he is getting badly beaten, or if he has a cut opened. Fighters jab, hook, and punch as their statistics allow. Sometimes they'll tie each other up, or back an opponent against the ropes. All you can do is chew at your fingernails, in the true manager tradition.

After each round, the judges' scores are displayed at the top of the screen, showing how each judge awarded points. Each fighter's stamina is changed to reflect how tired he is, and you can choose a different strategy for the next round. The fight continues as many rounds as you selected earlier, from a three-round preliminary bout, to a fifteen-round title fight. When the fight ends, the judges will tabulate their scores and announce a winner, unless a knockout or technical knockout has been called earlier in the fight. Whatever the decision, you can see the judges' scorecards after the fight, seeing how each awarded points and how many knockdowns each fighter had.

As with *Computer Baseball*, you have the option in *Ringside Seat* to enter new data for other fighters, or even to create a fictional boxer, giving him abilities of your own choice.

Fighting It Out

This game is much faster moving than *Computer Baseball*. In a way, that's a disadvantage, for it moves quicker because there are fewer decisions for you to make. I didn't feel as involved in this game as in the baseball simulation for that reason.

The game still gives you the flavor of managing a fighter, and of a bout itself. Pacing is important in the game, for if you expend too much energy early in the fight by constantly choosing to go for the knockout, or to charge in, your fighter will be weak before the fight ends. As the fighter's stamina falls, so does his effectiveness in many of the strategies. When his effectiveness falls below 2, his punches come with less frequency, and land less often.

As in *Computer Baseball*, the animation in *Ringside Seat* is not up to arcade standards. The sound is weak as well. But these detract little from the game's attractions. No arcade-style boxing game gives you the strategic choices and actual fighters that *Ringside Seat* does. Learning how to direct a fighter takes time and practice. This game lets you experience the thrills and agonies of professional boxing, without ever stepping into a gymnasium. Maybe *your* fighter can be a contender.

Computer Baseball
Ringside Seat
Strategic Simulations, Inc.
883 Stierlin Road, Building A-200
Mountain View, CA 94043
$39.95 each

C. REGENA

Built-In Functions

In my previous columns I've tried to show some *fun* things you could do with your computer. This month I'm going to discuss some built-in functions so you can see that your computer can perform technical tasks, and not just games, music, and graphics. Your microcomputer can do many things that the "mainframe" computers can.

The computer can be a very powerful tool in mathematical calculations. A computer can go beyond a calculator by putting procedures into a program for repetitious work or for logic-dependent answers. The slide rule generation was limited to three significant figures, and problems may have taken hours of work and pads of paper to solve. The calculators streamlined problem-solving and took the drudgery out of mathematics. Now the microcomputers can solve problems with even less time and effort—and to many decimal places of accuracy. The VIC-20 and Commodore 64 display numbers with nine significant figures (ten are stored internally).

The π key is handy in any calculations involving pi (the ratio of the circumference to the diameter of a circle). π has a built-in value on the VIC and 64 and is available directly from the keyboard. Use SHIFT and the up-arrow key to get the π symbol. Try the command PRINT π and you will get the decimal equivalent of pi. To use π in any calculations, just use the symbol. For example, try PRINT 3* π and press RETURN. This short program, "Circles," illustrates the use of this key:

Circles

```
100 PRINT "{CLR} ** CIRCLES **"   :rem 162
110 PRINT "{DOWN}ENTER THE RADIUS.{DOWN}"
                                  :rem 218
120 INPUT "R = ";R                :rem 131
130 PRINT "{DOWN}AREA = ";↑*R*R   :rem 254
140 PRINT "{DOWN}CIRCUMFERENCE = ";↑*2*R
                                  :rem 129
150 PRINT "{3 DOWN}ANOTHER CIRCLE? (Y/N)
    {SPACE}";                     :rem 254
160 GET A$                        :rem 220
170 IF A$="Y" THEN 100            :rem 38
180 IF A$<>"N" THEN 160           :rem 95
190 PRINT A$                      :rem 140
200 END                           :rem 105
```

The square root function is available on our computers. SQR(X) will return the square root of a number with the variable name of X. Try PRINT SQR(16) to get the square root of 16, or 4. Some valid statements are:

```
10 X=SQR(T)                       :rem 145
30 A=SQR(B)+3+Y                   :rem 76
70 Z=SQR(C*3+F)                   :rem 86
40 ON SQR(R) GOTO 80,100,150      :rem 186
```

"Radius" is a sample program that illustrates the use of the square root function to calculate the radius of a circle if the area is given ($A = \pi r^2$).

Radius

```
100 PRINT "{CLR}RADIUS"           :rem 189
110 PRINT "{DOWN}ENTER AREA OF CIRCLE."
                                  :rem 128
120 INPUT "A = ";A                :rem 97
130 IF A>0 THEN 160               :rem 152
140 PRINT "{DOWN}SORRY, MUST BE > 0."
                                  :rem 174
150 GOTO 110                      :rem 97
160 R=SQR(A/↑)                    :rem 220
170 PRINT "{DOWN}RADIUS = ";R     :rem 12
180 PRINT "{3 DOWN}ANOTHER CIRCLE? (Y/N)
    {SPACE}";                     :rem 1
190 GET A$                        :rem 223
200 IF A$="Y" THEN 100            :rem 32
210 IF A$<>"N" THEN 190           :rem 92
220 PRINT A$                      :rem 134
230 END                           :rem 108
```

ABS(X) is a function that returns the absolute value of a number X. The absolute value of a number is the numeric value without regard for the sign. The absolute value of a negative number is the number without the minus sign. Some valid statements are:

```
10 A=ABS(Y)                       :rem 95
20 IF ABS(SC)=100 THEN 350        :rem 67
30 T=T+ABS(T1-T2)                 :rem 210
```

Merry Christmas!

JUST AS THE DRUIDS COMPILED THE FIRST COMPUTER (Stonehenge) so they also brought the first decorated tree into the home. The winter celebration of these mystics has filled us with the desire to continue their special ways. LISTEN . . .

CASSETTALOG!

DID YOU HEAR IT? Imagine a cassette containing • Descriptions of all our games • Programming hints • Other products • **FREE GAME** • Much more! One side is programmed for the Commodore 64™. The other for the VIC 20™. Send just $2.00 to cover costs and then **DEDUCT $3.00 FROM YOUR FIRST ORDER!** Actually, the very fun of our cassettalog would make a nice Christmas present itself!

GAMES!

Standard VIC 20™ Cassette

Alien Panic	$9.95
Krazy Kong	$9.95
Racefun	$12.95
The Catch	$12.95
Antimatter Splatter	$16.95
Exterminator	$19.95
Music Writer III	$16.95
Kings Ransom	$16.95

Expanded Memory VIC 20™ Cassette

3-D Man (+3K)	$12.95
Space Quest (+8K)	$16.95
Defender on Tri (+3K)	$12.95

Commodore 64™ Diskette

3D-64 Man	$19.95
Exterminator 64	$19.95
Widows Revenge	$19.95
Quiz Pro	$19.95

BONUS!

The first 250 orders over $20 placed from this advertisement will receive a Nufekop dust cover for the 64 or VIC 20 FREE OF CHARGE!

nüFEKOP

P.O. Box 156, 21255 Hwy. 62, Shady Cove, Oregon 97539 1-800-525-2929

INT(X) returns the integer value of a number X, or the whole number part of a number which contains a decimal. The integer function truncates the decimal portion of a number, but it does not round the number. The result is always the largest whole number smaller than the given number, or the whole number to the left of a given number on the number line. Thus, INT(4.56) will be 4, but for negative numbers, INT(−4.56) will be −5. Some valid statements are:

```
10 I=INT(X/Y)                          :rem 3
30 J=J+INT(A)                         :rem 220
50 ON INT(S) GOSUB 200,250,270        :rem 43
```

SGN(X) returns the sign of a number X. The value will be 1 for positive numbers, 0 for zero, and −1 for negative numbers. This function is useful in games where the position of an object may be positive, negative, or zero in relationship to another object. The score could also be tested with the SGN function. Valid statements are:

```
10 S=SGN(X-Y)                          :rem 8
20 ON SGN(SC-T) GOTO 150,370,370 :rem 170
40 IF SGN(R)=-1 THEN 430         :rem 223
```

The following program illustrates the absolute value function, integer function, and sign function for several numbers.

```
100 PRINT "{CLR}NUMBER{3 SPACES}ABS
    {2 SPACES}INT{2 SPACES}SGN"  :rem 103
110 FOR I=1 TO 7                  :rem 10
120 READ N                       :rem 253
130 PRINT N;TAB(8);ABS(N);TAB(14);INT(N);
    TAB(19);SGN(N)               :rem 214
140 NEXT I                        :rem 29
150 DATA 3.4,0,0.6,-2.1,-5,7.2,-5.3
                                 :rem 139
160 END                          :rem 110
```

The VIC-20 and Commodore 64 have several built-in trigonometric functions. Specify a number, numeric variable, or numeric expression within the parentheses (called the "argument" of the function).

SIN(X) returns the sine of an angle specified as X radians.

COS(X) returns the cosine of an angle specified as X radians.

TAN(X) returns the tangent of an angle specified as X radians.

ATN(X) returns the arctangent of a number X. Arctangent is the angle with the tangent of X. The angle will be expressed in radians.

When you are working with angles, remember that the computer uses angles expressed in radians. Since 180 degrees equals pi radians, you can convert D number of degrees to R radians with R=D* π/180. The conversion from radians to degrees is D=R*180/ π .

If you need some of the other trigonometric functions, remember these conversions.

Cotangent (X)=cosine(X)/sine(X) or 1/tangent(X)

Secant(X)=1/cosine(X)

Cosecant(X)=1/sine(X)

Some functions are not defined for certain angles (such as the tangent of 90 degrees), and you need to be careful of overflow conditions or division by zero for the reciprocal functions.

The programs following this column illustrate the use of these trigonometric functions. Enter an angle expressed in degrees, D. Line 170 converts the degrees to radians. The sine, cosine, tangent, cotangent, secant, and cosecant of the angle are printed.

Two more technical functions are the exponential and logarithmic functions. EXP(X) returns e to the power of X, where X is a numeric expression that must be less than or equal to 88.02969191. LOG(X) returns the natural logarithm of X, and X must be a number greater than zero. No longer do you need a book of math tables, nor do you need to calculate interpolations—your computer can calculate logarithms and exponentials almost instantly. Sample valid statements are:

```
10 PRINT LOG(X/Y)                      :rem 1
20 A=EXP(B)                           :rem 96
50 G=LOG(H)-LOG(I)                    :rem 13
70 IF EXP(F)>=50 THEN 200             :rem 27
```

If the computer does not have a built-in function that you need, you can define your own function. The definition procedure is useful if you have a long mathematical formula that is used several places in the program. You can save computer memory and typing time by defining the function at the beginning of the program, then every time you need the function, it is called by the function name.

To define a function, use DEF FN with a variable name (one or two letters long) including a variable name within parentheses. For example,

```
10 DEF FNG(X)=3*X*X+4*X+2
```

Here a function G(X) is defined with a formula. Later in the program you can use a statement such as

```
50 PRINT FNG(7)
```

and G(X) will be evaluated with X=7.

The definition statement needs to be executed before the function is used in the program, so it is a good idea to put all definitions at the beginning of the program.

The above example used a function dependent upon a variable X. The defined formula does not have to contain a variable. For example, we could define a function R(Y) as follows.

```
10 DEF FNR(Y)=INT(8*RND(1))+1         :rem 93
```

R(Y) is defined as a random number from 1 to 8.

Commodore 64 and VIC-20

SuperTerm

$149.95

Telecommunications with a difference!

Unexcelled communications power and compatibility, especially for professionals and serious computer users. Look us over; **SuperTerm** isn't just "another" terminal program. Like our famous Terminal-40, **it's the one others will be judged by.**

- **EMULATION**—Most popular terminal protocols: cursor addressing, clear, home, etc.
- **EDITING**—Full-screen editing of Receive Buffer
- **UP/DOWNLOAD FORMATS**—CBM, Xon-Xoff, ACK-NAK, CompuServe, etc.
- **FLEXIBILITY**—Select baud, duplex, parity, stopbits, etc. Even work off-line, then upload to system!
- **DISPLAY MODES**—40 column; 80/132 with side-scrolling
- **FUNCTION KEYS**—8 standard, 52 user-defined
- **BUFFERS**—Receive, Transmit, Program, and Screen
- **PRINTING**—Continuous printing with Smart ASCII interface and parallel printer; buffered printing otherwise
- **DISK SUPPORT**—Directory, Copy, Rename, Scratch

Program options are selected by menus and function keys. For maximum convenience, an EXEC file sets all options on start-up. SuperTerm may be backed-up for safety. Software on disk with special cartridge module.

Write for the full story on SuperTerm; or, if you already want that difference, order today!

Requires: Commodore 64 or VIC-20, disk drive or Datasette, and compatible modem. VIC version requires 16K memory expansion. Please specify VIC or 64 when ordering.

Smart ASCII Plus . . . $59.95

The only interface which supports streaming —sending characters simultaneously to the screen and printer — with SuperTerm.

Also great for use with your own programs or most application programs, i.e., word processors. **Print modes:** CBM Graphics, TRANSLATE, DaisyTRANSLATE, CBM/True ASCII, and PIPELINE.

Complete with printer cable and manual. On disk or cassette.

VIC 20 and Commodore 64 are trademarks of Commodore Electronics, Ltd.

(816) 333-7200

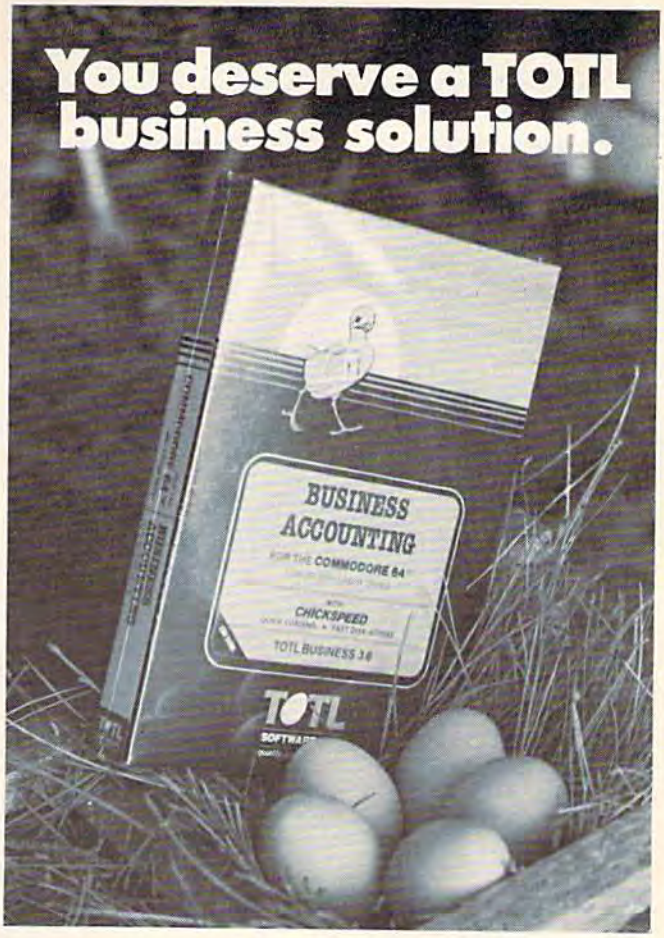
Send for a free brochure.

MIDWEST MICRO inc.

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.

311 WEST 72nd ST. • KANSAS CITY • MO • 64114

You deserve a TOTL business solution.



WORD PROCESSING
TOTL.TEXT

MAILING LIST AND LABELS
TOTL.LABEL

TIME MANAGEMENT
TOTL TIME MANAGER

KEYWORD CROSS REFERENCE
RESEARCH ASSISTANT

For Commodore 64™ and VIC 20™

Announcing the newest members of the family...

BUSINESS ACCOUNTING FOR **$95** (SUG. RETAIL)
TOTL.BUSINESS

SPELLING CHECKER FOR **$35** (SUG. RETAIL)
TOTL.SPELLER (64 only)

DATABASE MANAGEMENT FOR **$50** (SUG. RETAIL)
TOTL.INFOBASE

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 _____

Now, within the program every time we need a random number from 1 to 8, we can use R(Y):

```
50 C=R(Y)                              :rem 225
80 A=R(Y)+R(Y)                          :rem 9
90 IF R(Y)>4 THEN X=2                  :rem 92
```

A defined function may combine other functions, such as

```
10 DEF FNF(X)=1-SIN(X)                 :rem 225
```

You can see that by using user-defined functions you can really customize your programs and make technical calculations less complicated.

I hope this discussion of the built-in numeric functions and the user-defined functions has shown you the powerful potential of your "home" computer. Technical applications which used to be possible only with large computers or with hours of calculation time are now possible with a combination of built-in functions on our home computers. In a later column I'll discuss the versatility of the string functions available on the VIC and 64.

Angles (VIC Version)

```
100 PRINT "{CLR}** ANGLES **"          :rem 87
110 PRINT "{DOWN}ENTER ANGLE IN DEGREES"
                                       :rem 239
120 INPUT "D = ";D                     :rem 103
130 IF D>=0 THEN 160                   :rem 216
140 PRINT "SORRY, 0<D<360"             :rem 182
150 GOTO 110                            :rem 97
160 IF D>360 THEN 140                    :rem 5
170 R=D*↑/180                          :rem 92
180 PRINT "EQUIVALENT RADIANS = ",R
                                       :rem 37
190 PRINT "SINE OF ANGLE =",SIN(R)
                                      :rem 140
200 PRINT "COSINE OF ANGLE =",COS(R)
                                       :rem 17
210 IF D=90 THEN T$="INFINITY":GOTO 240
                                      :rem 167
220 IF D=270 THEN T$="INFINITY":GOTO 240
                                      :rem 216
230 T$=STR$(TAN(R))                     :rem 62
240 PRINT "TANGENT OF ANGLE = ",T$:rem 85
250 IF T$="INFINITY" THEN C$="0":GOTO 280
                                      :rem 221
260 IF TAN(R)=0 THEN C$="INFINITY":GOTO 2
    80                                :rem 168
270 C$=STR$(1/TAN(R))                 :rem 145
280 PRINT "COTANGENT OF ANGLE = ",C$
                                      :rem 218
290 IF COS(R)=0 THEN S$="INFINITY":GOTO 3
    10                                :rem 183
300 S$=STR$(1/COS(R))                 :rem 157
310 PRINT "SECANT OF ANGLE = ",S$:rem 255
320 IF D=0 OR D=180 OR D=360 THEN CS$="IN
    FINITY":GOTO 340                   :rem 41
330 CS$=STR$(1/SIN(R))                :rem 232
340 PRINT "COSECANT OF ANGLE = ",CS$
                                      :rem 215
350 PRINT "{DOWN}ANOTHER ANGLE? (Y/N)"
                                       :rem 88
360 GET A$                            :rem 222
370 IF A$="Y" THEN 100                 :rem 40
380 IF A$<>"N" THEN 360                :rem 99
390 PRINT A$                          :rem 142
400 END                               :rem 107
```

Angles (64 Version)

```
100 PRINT "{CLR}** ANGLES **"          :rem 87
110 PRINT "{DOWN}ENTER ANGLE IN DEGREES"
                                       :rem 239
120 INPUT "D = ";D                     :rem 103
130 IF D>=0 THEN 160                   :rem 216
140 PRINT "SORRY, 0<D<360"             :rem 182
150 GOTO 110                            :rem 97
160 IF D>360 THEN 140                    :rem 5
170 R=D*↑/180                          :rem 92
180 PRINT "{DOWN}EQUIVALENT RADIANS = ";R
                                       :rem 69
190 PRINT "{DOWN}SINE OF ANGLE =";SIN(R)
                                      :rem 172
200 PRINT "{DOWN}COSINE OF ANGLE =";COS(R
    )                                  :rem 49
210 IF D=90 THEN T$="INFINITY":GOTO 240
                                      :rem 167
220 IF D=270 THEN T$="INFINITY":GOTO 240
                                      :rem 216
230 T$=STR$(TAN(R))                     :rem 62
240 PRINT "{DOWN}TANGENT OF ANGLE = ";T$
                                      :rem 117
250 IF T$="INFINITY" THEN C$="0":GOTO 280
                                      :rem 221
260 IF TAN(R)=0 THEN C$="INFINITY":GOTO 2
    80                                :rem 168
270 C$=STR$(1/TAN(R))                 :rem 145
280 PRINT "{DOWN}COTANGENT OF ANGLE = ";C
    $                                 :rem 250
290 IF COS(R)=0 THEN S$="INFINITY":GOTO 3
    10                                :rem 183
300 S$=STR$(1/COS(R))                 :rem 157
310 PRINT "{DOWN}SECANT OF ANGLE = ";S$
                                       :rem 31
320 IF D=0 OR D=180 OR D=360 THEN CS$="IN
    FINITY":GOTO 340                   :rem 41
330 CS$=STR$(1/SIN(R))                :rem 232
340 PRINT "{DOWN}COSECANT OF ANGLE = ";CS
    $                                 :rem 247
350 PRINT "{3 DOWN}ANOTHER ANGLE? (Y/N)"
                                      :rem 122
360 GET A$                            :rem 222
370 IF A$="Y" THEN 100                 :rem 40
380 IF A$<>"N" THEN 360                :rem 99
390 PRINT A$                          :rem 142
400 END                               :rem 107
```

the 'THINK TANK'

Mirage Concepts' Power Packed Software & Your Commodore 64™!

If you've been using other software with your Commodore 64™, you have probably learned to compromise a lot! Maybe you have even considered investing in more expensive hardware. *Hold everything*...arm that Commodore 64™ with explosive firepower as well as Phi-Beta intellect! All you really need is a "Think Tank" working for you.

Mirage Concepts software:

- **Ingeniously simple to operate**
- **Unrestrained user flexibility**
- **Awesomely powerful**

Mirage Concepts brings a new definition to the phrase "Think-Tank" with the introduction of their companion WORD PROCESSOR and DATABASE MANAGER programs. Together, they bulldoze your Commodore 64™ through any obstacle...with fingertip ease! **User Confidence**–once you've experienced the simplicity and raw-power of Mirage Concepts software, you will never again settle for less! **Perfection** is the only standard we tolerate at Mirage Concepts. Put a "Think-Tank" in your Commodore 64™... ***ask your dealer for Mirage Concepts' WORD PROCESSOR and DATABASE MANAGER today. Only $99.95 each.***

MIRAGE CONCEPTS, INC.

2519 W. Shaw Ave., #106 • Fresno, CA 93711

FOR DEALER INFORMATION, CALL: (800) 641-1441 • IN CALIFORNIA (800) 641-1442

HORIZONS: 64

CHARLES BRANNON
PROGRAM EDITOR

Mailbag

Time to bounce back some of the ideas and suggestions you've sent me. In talking about Commodore 64 video, I mentioned that the new 8-pin 64s have improved video clarity, but said that there are no functions assigned to the additional pins (previous 64s had five-pin plugs for audio/video).

J. Robinson of Santa Monica, California, differs. He's rigged up a cable using an 8-pin DIN plug (sorry, Radio Shack doesn't carry them) with the chroma (color) signal coming from pin 8. Normally, chroma is slightly distorted by the luminance mixed with it, but the signal from pin 8 is pure, and the picture is better than ever, as evidenced by some color slides Mr. Robinson sent me. If you can find an 8-pin DIN plug, try it out. By the way, DIN stands for Deutsche Industre Norm (German Industrial Standard).

We've received some letters about the Automatic Proofreader, asking how the checksum is computed on each line. We send program listings to a disk, then run our Lister program, which reads the programs straight from the disk and formats the listings with cursor controls and graphics spelled out.

In addition, the Lister automatically generates the ":rem" checksums by adding together the ASCII values of all the characters in the line. The reason that the numbers are never larger than 255 is that the addition is done internally in only 8 bits, so it will wrap around from 255 to zero (like an odometer past 99999) if the sum is too large. That's why some numbers for long lines are smaller than other numbers for short lines. It all depends where the number wraps around.

Printer Interfaces

A few issues back, columnist Larry Isaacs talked about a parallel printer interface by CardCo called Card/Print (also written as "Card/?"). We've used this interface here and have been generally pleased with its performance, although it will not translate certain 64 control codes in the listing mode.

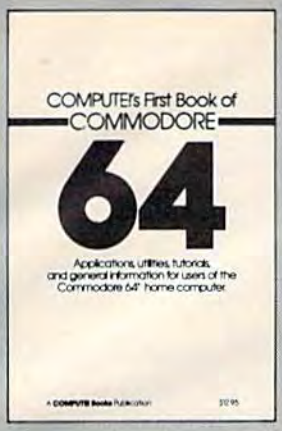
Another interface I've been using is the Tymac Connection. Unlike the Card/Print, the Connection is set up specifically for your printer (ROM chips are available for several printers). The Card/Print will work with almost any Centronics parallel printer, but the Connection uses the graphics capabilities of your Epson, Prowriter, or other dot-matrix graphics printer.

The Connection offers "almost total emulation" of the VIC printers, such as the 1525. In the emulation mode, it will respond identically to the control codes the VIC printer uses, such as dot graphics, elongated/normal text, cursor-up mode, cursor-down mode, and more. You can therefore use all the special features of your printer (high-quality print, italics, double-strike, etc.) and still be able to run programs specific to the VIC printers, such as high-resolution screen dumps.

In addition, the Connection uses your printer's dot graphics capabilities to actually print the built-in graphics characters on the keyboard. Program listings with graphics symbols will appear as they do on the screen. Unfortunately, the graphics characters are formed rather strangely. Characters which should connect, such as Commodore Q and SHIFT-asterisk, do not. The cursor symbols are hard to read. The Connection also cannot reproduce reverse-video text, since your printer's normal character set is used for alphanumerics. But support for graphics is better than no graphics symbols at all.

The Connection has a listing mode, though it cannot interpret some characters. There is also a "transparent" mode, where it just sends the characters along without any interpretation. This is what you would do with some word processing programs. Unlike the Card/Print, there is no way to "lock" in any one mode.

Hardware-wise, the Connection has a 6502 microprocessor with RAM, ROM, and a printer port. How strange to buy another computer for your 64. It has enough RAM to serve as a 2K printer buffer. With a buffer, characters coming from the computer are stored until the printer can "catch up." If you sent something less than 2K long, it would be instantly printed from the computer's



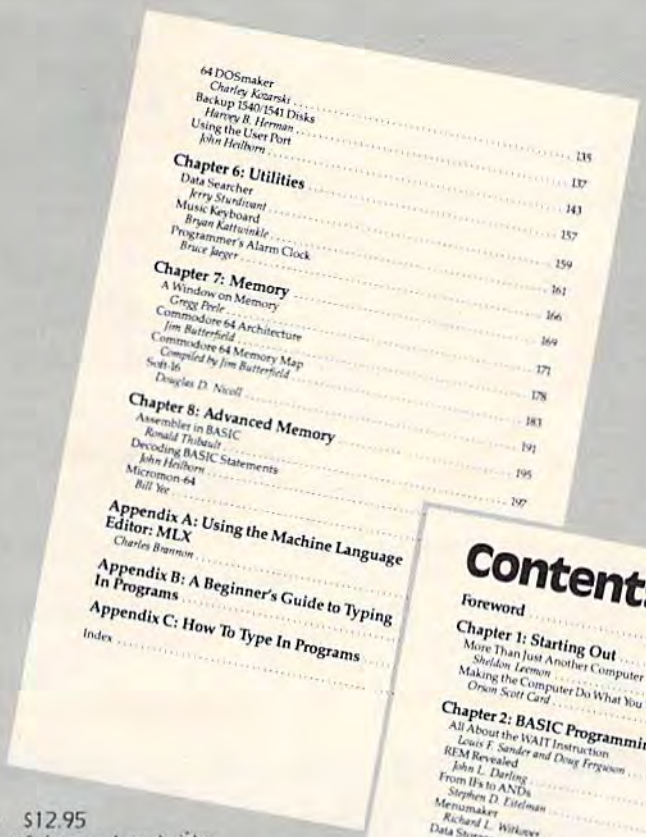
COMPUTE!'s First Book Of Commodore 64

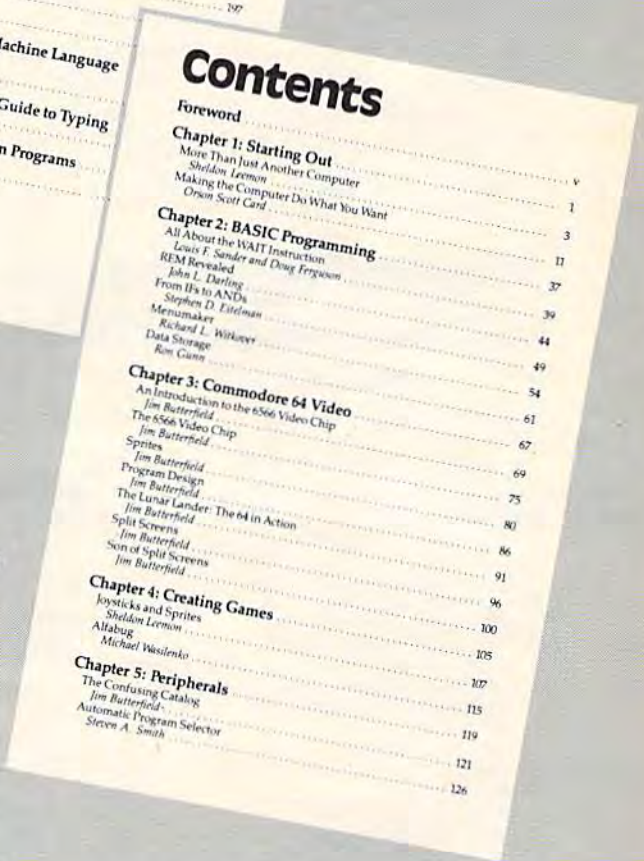
$12.95

The Commodore 64 is a powerful home computer. Its state-of-the-art features include sprite graphics and an advanced musical synthesizer chip. *COMPUTE!'s First Book of Commodore 64* is the definitive resource guide for owners and users of the 64. It includes some of the best articles and programs from *COMPUTE!* Magazine and *COMPUTE!'s GAZETTE*, plus many more that have never before appeared in print.

First Book Of Commmodore 64 contains something for everyone: BASIC programming techniques; data storage programs; examples of sprite graphics and musical synthesis; information about writing games, using peripherals, and working with many utilities (including a machine language assembler and a monitor); a complete memory map; and advanced programming techniques. Also includes many ready-to-type-in programs and games.

If you are a Commodore 64 owner who wants to learn more about using your computer to its full potential, this is the book for you.





Now Available From COMPUTE! Books

- *COMPUTE!'s First Book Of 64 Sound & Graphics* $12.95
Clear explanations to help you use all the 64's powerful sound and video features. Plus great programs for music synthesis, hi-res art, and sprite and character design.

- *COMPUTE!'s First Book Of Commodore 64 Games* $12.95
Packed full of games: Snake Escape, Oil Tycoon, Laser Gunner, Zuider Zee. Arcade-action machine language games for fast hands; strategy games for sharp minds.

- *COMPUTE!'s Reference Guide To 64 Graphics* $12.95
A complete, step-by-step tutorial for programming graphics. You'll like the clear writing, the example programs, and the full-featured sprite, character, and screen editors.

- *Creating Arcade Games On The 64* $12.95
The principles and techniques of fast-action game design, including custom characters, movement, animation, joysticks, sprites, and sound. With complete example game programs.

- *Commodore 64 Games For Kids* $12.95
Dozens of games for kids of all ages, making this an instant library of educational software. Learning, creativity, and excitement.

How To Order

COMPUTE! Books are available at bookstores, newsstands, and computer stores nationwide. If your local store has run out of **COMPUTE! Books** or does not carry them, you can order them directly:

By Mail. Send a check or money order in U.S. funds (no cash please) to: Order Department, COMPUTE! Books, P.O. Box 5406, Greensboro, NC 27403. Please include $2 shipping/handling for each book you order. All orders must be prepaid. (C.O.D. orders are not accepted.) Allow 4–6 weeks for delivery.

By Phone. If you have VISA, MasterCard, or American Express, call our **toll-free number 800-334-0868** (919-275-9809 in NC) between 8:30 AM and 4:30 PM, EST. Your order will be processed promptly; we will bill your credit card (plus $2 shipping/handling). Allow 4–6 weeks for delivery.

COMPUTE!™ Publications, Inc.
One of the ABC Publishing Companies

point of view. The buffer would meanwhile be feeding characters to the printer at the printer's speed. My printer has a 2K buffer, so the combination of the printer and the Connection gives me an effective 4K of buffer space.

The Connection (Tymac)
Distributed by Microware
1342 B Rt. 23
Butler, NJ 07405
$119

Strange Lock-Up Bug

It may have happened to you. It's extremely frustrating, and totally unexpected. Fortunately, it's predictable, hence preventable. Go down to the bottom of the screen (the very bottom, the last line, scroll if you have to). Now start typing. Anything. You could just type a bunch of X's. Keep typing as you wrap around the right margin on the first line, then stop right after you type past the right margin of the second 40-column line (the screen will scroll). Now press the INST/DEL key (unSHIFTed) to erase the last character on the previous line.

Whoa! Suddenly, the command LOAD is printed, you get a ?SYNTAX ERROR, the word RUN appears, and if you had a program in memory, it starts. Who typed SHIFT-RUN/STOP?

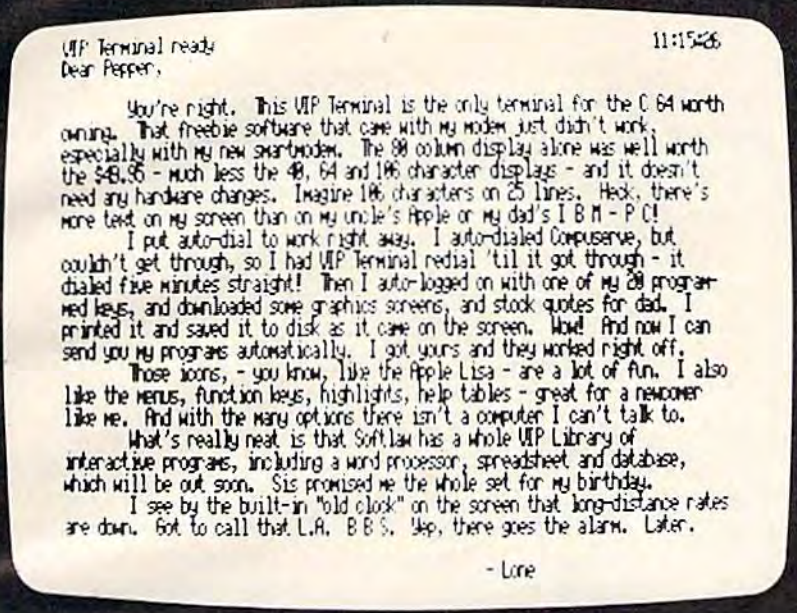
This is not a trivial error. If you had a program running, you can't stop it. If not, the cursor appears to be flashing merrily, and all seems well. Try typing. Worry begins to creep into your mind as no characters appear. You reach for the panacea of RUN/STOP-RESTORE. Panic sets in when this does nothing, and you press it over and over again, pounding and smashing at the keyboard in a frantic attempt to regain control. Too bad. You have to reach for the power switch and turn your computer off. RAM is wiped clean. Your program, if any, is gone.

I don't know what causes this error. One theory is that when the 64 tries to scroll color memory, it reaches one line too many past $DBE7 (end of color memory) and mangles the registers of the CIA chip, which controls all interrupts. There is no way around it, other than the emergency reset I covered last month. Just keep it in mind.

Hope you enjoy the word processor in this issue. Since it is a complex software product, we'll use this column in the future as a forum for answering your questions about it, as well as tips for using it.

I'll leave you with something to play with: extended background color mode. Enter it with POKE 53265,PEEK(53265)OR64 and try typing the letter "A", SHIFT-A, then CTRL-9 (reverse on), and inverse video "A" and SHIFTED-A again. See what conclusions you come up with, and try changing memory locations 53282 and 53283.

80 Column Smart Terminal
For Your C64 Without Any Hardware Change!



VIP TERMINAL™

They're right! To start with the best you've got to have the **VIP Terminal!**

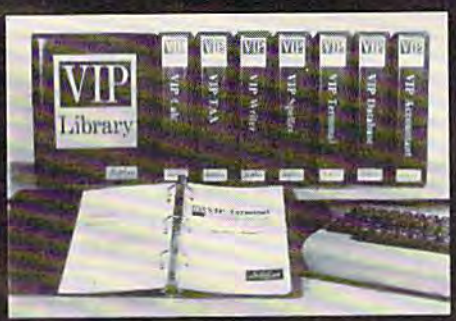
- **Built-in 40, 64, 80 and 106 columns**
- **Word wrap for a formatted display**
- **Talk to any computer**
- **Use any modem and printer**
- **Written in fast machine code**
- **15 entry phone directory**
- **20 programmable keys**
- **Automatically dial, redial, upload, download and log-on**
- **Professional 96 character ASCII display**
- **128 character ASCII keyboard**
- **Simultaneous on line printing and saving of files to tape and disk**
- **Use and save files as big as your disk!**
- **E Mail & E-COM Compatible**

Get yours NOW! **$49.95**

Introducing The VIP Library

The Library Concept

The **VIP Terminal** is only the first in a whole series of elegant software for your Commodore 64 called the **VIP Library.** This complete collection of easy-to-use, serious, high quality, totally interactive productivity software includes **VIP Writer, VIP Speller, VIP Calc, VIP Database, VIP Disk-ZAP, VIP Accountant and VIP Tax.** All are equal in quality to much more expensive software for the IBM PC, and all are very affordable!



Virtual Memory

VIP Library programs are not limited by the size of your computer's memory. All programs use virtual memory techniques to allow creation and use of files larger than your computer's available work area. You're only limited by the space on your disk!

Icons Make Learning Easy

Hi-res technology and sprites allow **VIP Library** programs to bring you task Icons, made famous by the Apple Lisa™ and the Xerox Star™. With these advanced sprite representations of the task options open to the user, even the total novice can, at a glance, perform every task with ease. Just look at the icon and press a key! No programs are easier or more fun to learn and use!

Total Compatibility

All **VIP Library** programs are compatible with each other and other computers for easy file transfer. Each uses ASCII, the universal language of computer communications so that files can be sent to and received from other computers without modification! The Library also gives you the benefit of a consistent icon and command structure. Once you have learned one program, the others will come easily.

For Orders ONLY — Call Toll Free —

1-800-328-2737

Order Status and Software Support call (612) 881-2777

Available at Dealers everywhere. If your Dealer is out of stock ORDER DIRECT!

MAIL ORDERS: $3.00 U.S. Shipping ($5.00 CANADA; $10.00 OVERSEAS. Personal checks allow 3 weeks.

Professional Displays

The 40-characters-per-line display of the Commodore 64 is inadequate for serious computing. An 80-column display is the industry standard. **VIP Library** programs bring this standard to your Commodore 64 with state-of-the-art graphics, without need for costly hardware modifications. With **VIP Library** programs you can freely choose from four displays: the standard 40 column display, plus a 64, 80 and even a 106 column by 25 line display. With these programs you can have more text on your screen than on an IBM PC or an Apple IIe with an 80-column board! Welcome to the professional world!

Who Is Softlaw?

Softlaw Corporation has years of software experience in micros. We currently offer the full-line **VIP Library** for other micros in the U.S. and in Europe. Now we are bringing this experience to the Commodore 64 so you get ultra-high quality software at very affordable prices.

Softlaw

9072 Lyndale Ave. So., Mpls., MN 55420

AUTHOR'S SUBMISSIONS ARE ENCOURAGED.

DAN CARMICHAEL, ASSISTANT EDITOR

Using The Dynamic Keyboard

The *dynamic keyboard* is a useful programming technique that can be used with both the VIC-20 and the 64. This technique enables you to POKE values into the *keyboard buffer* and "fool" the computer into thinking you typed the command from the keyboard. It can be used to do everything from simply running a program to chain-loading programs from tape or disk.

The Keyboard Buffer

The keyboard buffer is a block of memory ten bytes (characters) long that runs from memory addresses 631 to 640. The buffer is a temporary holding area that is used to store data input from the keyboard. If you could type faster than the VIC could read the keyboard (it does this 60 times a second), you could fill up the keyboard buffer. Obviously you can't type this fast, but there are other times the computer's operating system does use the keyboard buffer to temporarily store data. If a BASIC program is running and is at a stage where it is not ready to accept data input from the keyboard (a FOR/NEXT time-delay loop, for example), the keys that are pressed will be temporarily stored in the buffer until the program is ready to process the data. To see how this happens, enter the following line in the direct mode, press RETURN, and while the FOR/NEXT loop is running type the following ten keys: 1 2 3 4 5 6 7 8 9 0.

```
FOR A=1 TO 10000: NEXT
```

As you can see, while the time-delay loop is running, the keys you pressed are not displayed on the screen. As soon as the loop is finished, the computer reads what is stored in the keyboard buffer and processes the data accordingly and prints it on the screen. Now enter the above FOR/NEXT statement again, and while it is running type the following 12 keys: 1 2 3 4 5 6 7 8 9 0 1 2. When the loop is finished, you will see the same ten keys displayed on the screen as before. But what happened to the two extra keys that were pressed, the 1 and 2? Those keys were lost because the keyboard buffer, which runs from 631 to 640, can hold a maximum of ten characters. Any keys that are pressed after the buffer is filled are lost.

The Other Keyboard Buffer Byte

You may wonder how the operating system knows that there is data in the keyboard buffer waiting to be processed. Memory address 198 tells the computer how many characters are in the keyboard buffer ready to be processed. Each time you pressed a key while the computer was executing the above time-delay loop, memory location 198 was incremented by 1. After the loop finished running, the operating system took a look at location 198 (which had a value of ten) and knew that there were ten characters in the buffer waiting to be processed. Enter the following commands in the immediate mode, then while the FOR/NEXT loop is running, type 1 2 3 4 5 6 7 8 9 0.

```
FOR A=1TO10000:NEXTA:POKE 198,0
```

After the loop is finished, you'll notice that our ten characters were not printed on the screen. The reason is that after we came out of the loop we POKEd 198 with a 0. Even though our ten characters *were* in the cassette buffer, the operating system didn't print them as it did before. When we POKEd a value of zero into 198, we told the computer no characters were waiting in the buffer.

Using The Dynamic Keyboard

The keyboard buffer can be a very useful tool when properly used. For example, did you know that when you "chain-load" programs (the first program automatically loading and running the second, the second loading the third, and so on) the

THE VIC LIBRARY FROM COMPUTE!

For over two years, COMPUTE! Magazine has been publishing articles and ready-to-type-in programs for VIC-20 owners. Many VIC programming techniques first appeared in the pages of COMPUTE!.

COMPUTE!'s editors have gathered the best articles (revising and rewriting some) and added previously unpublished material. The result is a series of books for VIC-20 owners. Each contains articles and programs which educate, explain, and entertain.

Whether you're brand-new to computing or a seasoned programmer, you'll find fun and useful information, written in COMPUTE!'s easy-to-understand style, in all of these books. And if you know other VIC owners, COMPUTE! books make wonderful Christmas gifts.

COMPUTE!'s First Book of VIC $12.95

COMPUTE!'s First Book of VIC has been a number one best seller on the computer book lists. It has been called "useful for both novice and experienced VIC users."

If you like computer games, you will enjoy "STARFIGHT3," "ZAPII," "Meteor Maze," and the other games included. Chapters on joysticks and paddles explain how to use them in programs you write.

First Book of VIC shows you a simple way to alphabetize lists. And there are advanced programming techniques for saving memory space.

The custom character program allows you to design your own alphabet (if you want your VIC to print Hebrew, Greek, or other special characters). Custom characters can be developed into spaceships, flowers, and other unique shapes.

You will discover how to automatically renumber BASIC programs, combine two programs, start and stop the screen when it is LISTing, and dump the screen to a printer.

If you write programs, you will appreciate the discussion of VIC memory and how to create special effects with graphics and sound. The machine language monitor, TINYMON, is a valuable aid to machine language programmers.

COMPUTE!'s First Book of VIC was one of the first books published for VIC owners and it is still one of the best. VIC owners of all ages and abilities will find it an excellent resource.

COMPUTE!'s Second Book of VIC $12.95

Your VIC library is not complete without *COMPUTE!'s Second Book of VIC*.

Some of the outstanding programs include: a datassette filing system that lets you use your VIC as a filing cabinet; a mini word processor; a loan analyzer that figures out interest, monthly payments, and loan values; a machine language assembler (and a very fast machine language game); a program for printing poster-sized banners; and much more.

Second Book of VIC features a detailed memory map (including Kernal routines) that not only names the memory locations, but also gives you ideas for using them in your programs.

Chapters on graphics and sound include a character editor that redefines character shapes, a sound effects program that does not slow down BASIC programs, and a utility for programming the eight function keys.

And, of course, there are ready-to-type-in games: "UXB," "A Day At The Races," and others. If you write your own games, you will benefit from the techniques and examples throughout the book.

People who are curious about the many things you can do with a VIC need *COMPUTE!'s Second Book of VIC*. The variety of programs will be a welcome source of new software. Why not order your copy today?

COMPUTE!'s First Book of VIC Games $12.95

Would you like a lot of games at a very low price? *COMPUTE!'s First Book of VIC Games* contains two dozen (for only $12.95). Each game has been tested and debugged, and is ready to type into your VIC-20.

You get fast-action, commercial quality, arcade-style games that require quick reflexes in addition to strategy games which test your logical skills.

In "Time Bomb" you hear the ticking of a bomb about to blow up. You have to find and defuse it by making your way through the maze that extends beyond the borders of the screen.

"Closeout" takes you to a department store. You try to snatch as many sale items as possible, while avoiding the horde of nasty bargain hunters.

"Cryptic Numbers" combines logic and luck. Thinking logically, you have to decode the pattern picked by the VIC. After each turn you learn how close you were.

"Deflector" requires quick reactions and a sense of strategy. You control the bouncing ball by placing deflectors that direct it to the targets.

You are captain of a small space fortress in "Outpost." You have lasers and torpedos, but your energy is running low and the computer is on the blink. Will the supply ship arrive in time to save you?

First Book of VIC Games is more than just a book full of program listings. Each game is annotated. You can change the games if you like, or use the many programming techniques in games you write. And you can use the maze-drawing program in your own games.

Introductory chapters explain how to develop ideas for game programs, with hints and tips about the VIC's sound, color, and graphics capabilities.

No serious game player should be without *COMPUTE!'s First Book of VIC Games*.

NEW RELEASES

(Coming Soon from COMPUTE! Books)

Creating Arcade Games on the VIC
VIC Games for Kids
Things to Do in 4K or Less

HOW TO ORDER

COMPUTE! Books are available at bookstores, newsstands, and computer stores nationwide. If your local store has run out of COMPUTE! Books or does not carry them, you can order them directly:

– By Mail

Send a check or money order in U.S. funds (no cash please) to:

Order Department
COMPUTE! Books
P.O. Box 5406
Greensboro, NC 27403

Please include $2 shipping/handling for each book you order. All orders must be prepaid (C.O.D. orders are not accepted). Allow 4–6 weeks for delivery.

– By Phone

If you have VISA, MasterCard, or American Express, call our toll-free number 800-334-0868 (919-275-9809 in NC) between 8:30 AM and 4:30 PM, EST. Your order will be processed promptly; we will bill your credit card (plus $2 shipping/handling). Allow 4–6 weeks for delivery.

COMPUTE! Publications, Inc. abc
A Subsidiary Of American Broadcasting Companies, Inc.

first program has to be longer than the second? That's because of the variables. The start of BASIC variables always stays just past the end of your BASIC program. When you chain-load programs, any variables that are used in the first program are stored a few bytes past the end of BASIC. If the second program in the chain is longer, it will write over these variables, and all their values will be lost. To illustrate this, enter the following two short programs and save them to tape with the filenames specified below. Be sure to save the second program with the filename "TEST.LOAD.2" right after the first program (use the filename TEST.LOAD.1).

TEST.LOAD.1

```
10 A=10:B=20:C=30                    :rem 120
20 PRINT"{CLR}{DOWN} THIS IS PROGRAM 1"
                                     :rem 244
30 FORT=1TO500:NEXT                  :rem 189
40 LOAD"TEST.LOAD.2"                 :rem 182
```

TEST.LOAD.2

```
10 PRINT" THIS IS A DUMMY LINE"   :rem 251
15 PRINT" THIS IS A DUMMY LINE"     :rem 0
20 PRINT" THIS IS A DUMMY LINE"   :rem 252
25 PRINT" THIS IS A DUMMY LINE"     :rem 1
30 PRINT" THIS IS A DUMMY LINE"   :rem 253
35 PRINT" THIS IS A DUMMY LINE"     :rem 2
40 PRINT" THIS IS A DUMMY LINE"   :rem 254
50 PRINT"{CLR}{DOWN} THIS IS PROGRAM 2"
                                     :rem 248
60 PRINT"A=";A;"B=";B;"C=";C         :rem 41
```

As you can see, the first program sets three variables (A, B, and C), PRINTs a quick message to the screen, then LOADs in the second program, which prints a few messages to the screen (we made it longer here for our demonstration). It then prints the variables A, B, and C that were set during the first program. Note that the printed variable values are zeros, even though we initially set them at 10, 20, and 30, respectively.

What's the answer? The dynamic keyboard, of course. Change the last line (line 40) in the first program to read:

```
40 POKE 631,131 : POKE 198,1
```

Then reSAVE it using the same filename, and RUN it again. For our demonstration, this modification will work only with cassette. Save this modified version to tape, and be sure to save another copy of the second program right after it.

If you changed line 40 and saved both programs correctly, you'll notice that the first program successfully called in and ran the second, even though we removed the LOAD command in line 40.

This is accomplished by the two POKEs we used in line 40. POKE 631,131 places the token for SHIFT-RUN into the keyboard buffer. When you press SHIFT and RUN on the keyboard to LOAD and RUN a tape program, this character (131) is placed in the buffer. Jot this down; you probably won't find it in your *VIC-20 Programmer's Reference Guide*. We POKEd the SHIFT-RUN into location 631, the first byte of the keyboard buffer, because the buffer is of the FIFO (first in—first out) type. That is, when you press a key on the keyboard, the operating system places it in the first byte of the buffer; when ready to be processed, this will be the first character pulled out.

The other POKE we made in line 40 was POKE 198,1. This tells the operating system that there is one character in the buffer waiting to be processed. As soon as the 1 was POKEd into 198, the operating system was fooled into thinking you had typed SHIFT-RUN from the keyboard, and the computer LOADed and ran the next program.

The obvious drawback here is that it will only LOAD and RUN the next program *on tape*. This is because the POKEs we used did not specify a filename. So when using this method of chain-loading, be sure the programs are saved one after the other.

You'll notice that we still have a problem passing variables because of the longer length of the second program. There is another technique which can easily solve that.

Using The Buffer And The Screen

Delete line 40 from the first program and add the following lines:

```
32 PRINT"{CLR}{2 DOWN}LOAD"          :rem 11
40 PRINT"{6 DOWN}5 A=";A;":B=";B;":C=";C;
                                     :rem 113
45 PRINT"RUN"                        :rem 47
50 POKE631,19:FORA=632TO636:POKEA,13:NEXT
   :POKE198,6                        :rem 96
```

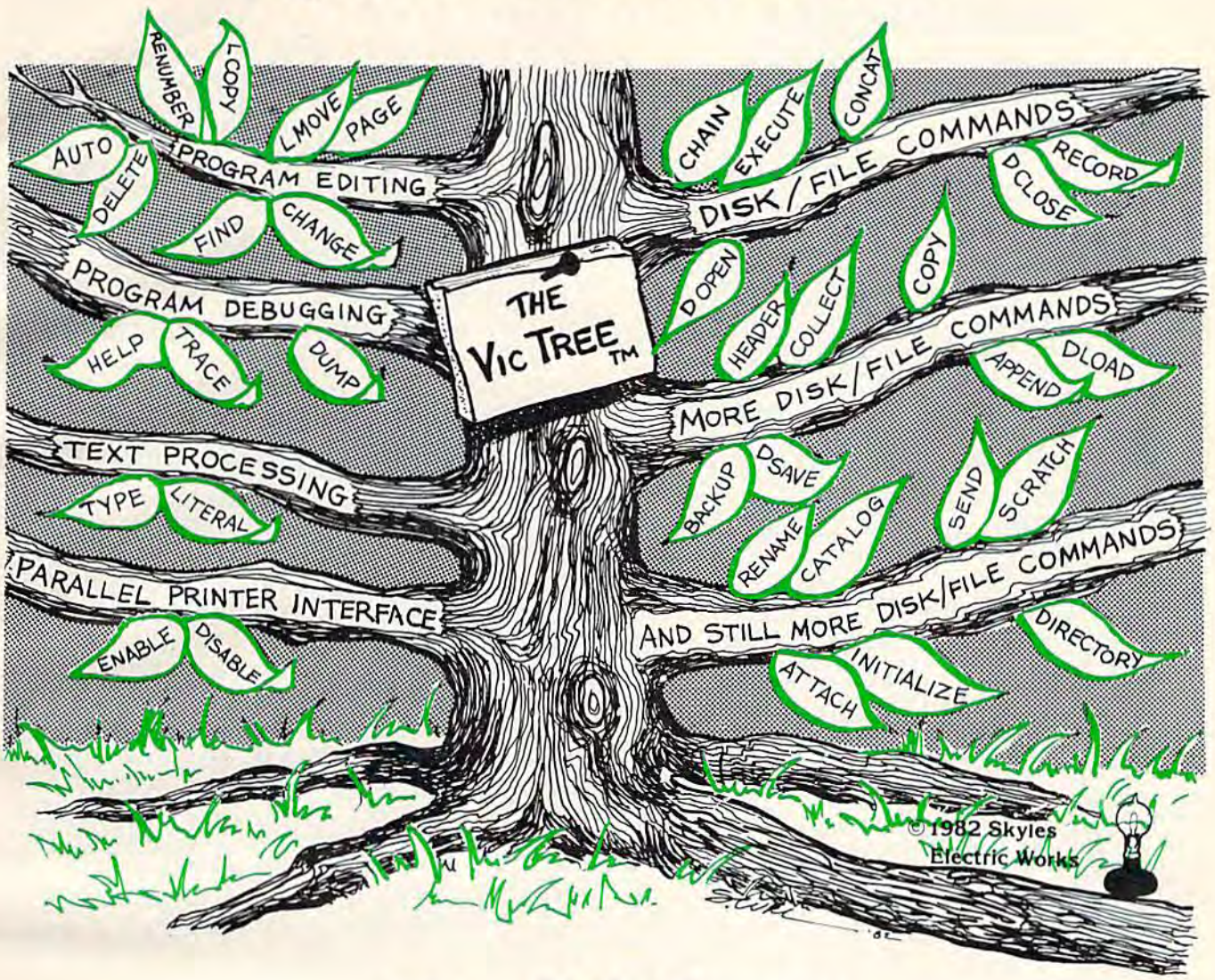
As before, SAVE it to tape and reSAVE the second program right after it. Now RUN the first program and see what happens. Be sure to leave the cassette PLAY button pressed down after the first program is loaded.

This time we've successfully passed our variable values from the first program to the second. Here's how we did it.

Line 32 clears the screen (which also moves the cursor to the HOME position). It then moves the cursor down two lines, and prints the word LOAD. Line 40 shows how we pass our variables from one program to another. It prints a line on the screen that looks like (and is) a BASIC statement: the number 5 (the BASIC line number) and the variables A=, B=, and C=. It then prints the current value of these variables, in this case 10, 20, and 30, respectively. Line 45 prints the word RUN and starts our second program.

Line 50 is the key to it all. The first POKE (POKE 631,19) places the value 19 into the first byte of the keyboard buffer. In this case, the ASCII

Skyles Electric Works Presents



The VicTree™

...Leaves your new Commodore 64 (or Vic 20) with 42 additional commands.
...Branches out to most BASIC 4.0 programs.
...Roots into most printers.

New from Skyles: the VicTree, a coordinated hardware and software cartridge that allows your Commodore 64 to branch out in unbelievable directions and makes it easier than ever to do BASIC programming, debugging and to access your disk. And the new VicTree provides routines to interface the Commodore 64 to all Centronics Parallel printers with an inexpensive cable. 8kb of ROM—4kb for the BASIC commands, 4kb for disk commands. Perfect not only for the new Commodore 64 but also for the Vic 20. Unbelievably simple to use with a 100+ page manual with many many program examples. Easy to install, plugs into cartridge port, the VicTree gives you all the additional BASIC 4.0 commands to allow most BASIC 4.0 programs to work on your new Commodore 64 and Vic 20.

And now with the BASIC command Commodore forgot!

Now only $89.95 or **$109.95** complete with Centronics standard printer cable. (Cable alone $29.95.)
Available now from your local dealer or order through your Visa or MasterCard toll free:
(800) 227-9998 (California, Canada, Alaska, Hawaii: (415) 965-1735) or send check or money order directly to:

Skyles Electric Works

231E South Whisman Road
Mountain View, CA 94041
(415) 965-1735

value of 19 is the control character which stands for HOME CURSOR. The cursor, then, is simply moved to the home position (the upper-left corner) without clearing the screen. The next five POKEs we make to the keyboard buffer are the same (a value of 13), and are done with a FOR/NEXT loop. The ASCII value of 13 is the control character which represents the RETURN key. POKE 198,6 fools the operating system into thinking that six keys have been pressed on the keyboard. The computer starts processing these commands, and away we go. Here's what happens.

The value of 19 moves the cursor to the HOME position, and then the operating system automatically moves it down one more line. The cursor is now sitting on top of the word LOAD we printed on the screen. The next character in the keyboard buffer is a 13, which stands for RETURN. We've fooled the computer into thinking that we typed the word LOAD and then pressed RETURN. The system now loads in the next program on the tape. While the computer is loading the program, the processing of data in the keyboard buffer is temporarily suspended.

After the LOAD has been completed (we now have the second program in BASIC memory), the operating system continues reading data from the keyboard buffer. The next few bytes in the buffer stand for RETURN. The cursor is now sitting on the BASIC line we printed on the screen in the first program. When the computer reads the next value of 13 from the keyboard buffer, it is again fooled into thinking we have just typed in a new BASIC line for the second program. It then enters that line. The cursor has now moved to the next line on the screen, which reads RUN. Reading another 13 from the buffer, it thinks we typed RUN from the keyboard and pressed RETURN, and away it goes with the second program.

Many Different Techniques

The above techniques could have been accomplished a number of different ways using the keyboard buffer and the screen. Just remember one important fact. In most cases, when the computer reads data from the buffer, it thinks it was entered directly from the keyboard. Almost anything you can do from the keyboard can be done from the keyboard buffer. As a matter of fact, you could probably create a computer program itself by POKEing various values into the buffer.

Experiment with this technique and I'm sure you'll find a useful application for it.

If you've got questions or ideas about subjects you'd like to see covered in this column, write to: VICreations, c/o COMPUTE!'s GAZETTE, P.O. Box 5406, Greensboro, NC 27403.

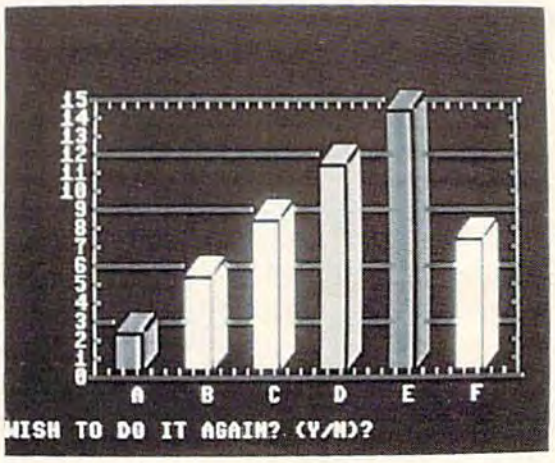
Graph Plotter

Ruth A. Hicks

Not only is "Graph Plotter" an interesting tool for drawing 3-D columnar charts, but the accompanying article takes you step by step through the program itself so you can learn how it was written. The program originally was designed for the Commodore 64, and we've added a version for the unexpanded VIC-20.

"Graph Plotter" is a good demonstration of what beginning programmers can accomplish in the way of graphics on the Commodore 64 and VIC-20. Different graphics techniques were used to create this program. By reading this article and following along with the program listing, you can increase your knowledge of graphics formatting. Of course, if you're not into learning programming, there's no reason why you can't just type in the program listing anyway.

Graph Plotter creates attractive bar graphs with three-dimensional columns. The graphs are particularly exciting in color. There are six columns, each a different color, to which you assign a value from 0 to 15 for the column height. You tell the computer what values each column has, and then you can interpret their meaning.

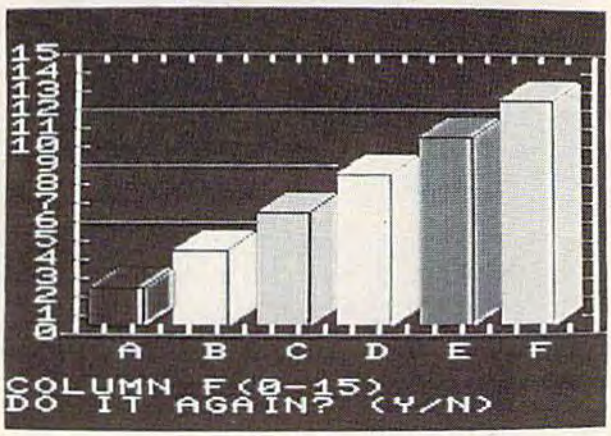


3-D bar graphs are a snap with "Graph Plotter" (64 version).

Modular Programming

Graph Plotter was written with a technique known as *modular* or *block programming*. This means a section at a time was written on the computer and then checked for eye appeal, function, and (of course) that familiar message, "?SYNTAX ERROR." There are five main blocks to this program.

When typing the program, I suggest that you omit unnecessary spaces except in any INPUT or PRINT statements between quotation marks. All



The VIC-20 version of "Graph Plotter."

other spaces are not needed by the computer and only consume more memory. Since this is an article to learn from, let's start some good habits right away by not typing those useless spaces.

Block One

Block one, lines 100-180, creates the graph, including the segments and the outlining border. Instead of using line after line of PRINT statements, we'll be POKEing the information directly into memory inside FOR/NEXT loops.

Line 100 clears the screen and sets the background color to black and the border blue. Line 110 starts the top border on the Commodore 64 at screen memory location 1230 and runs it across the screen to location 1261, drawing a continuous line (these addresses are 7726 to 7745 on the VIC). Refer to the manual which came with your computer for the "Screen and Color Memory Maps." Each time the FOR/NEXT loop is executed, it places the new value of I into the POKE statement with the symbol number 114 (refer to your manual, "Screen Display Codes"). The I value tells the computer *where* to put the symbol and the 114 tells *what* symbol to put in that spot.

The second POKE in line 110 colors the symbol green. Since the "Color Memory Map" (see manual) corresponds to the screen memory map, only with a different set of numbers, all we have to do is calculate the offset. The difference between 55296 and 1024 (the starting address of color and screen memory in the Commodore 64) is 54272, a simple subtraction problem. So, we POKE I+54272 with the color code for green (5) and presto, we have a green symbol at the correct location! The same thing works on the VIC-20, except the offset between screen and color memory is 30720 instead of 54272.

Line 120 draws the left border, beginning at screen memory location 1270 and ending at location 1790 on the Commodore 64 (7748 to 8034 on the VIC). The STEP 40 is used because a Commodore 64 has 40 characters per line across its screen (STEP 22 for the VIC). If you look at your manual and find screen location 1270, then add 40, you'll find that location 1310 is exactly one line below 1270. On the VIC, 7748+22 adds one screen line. The rest of line 120 and the next two lines are similar to lines 110 and 120, except for different screen symbol codes.

The last four lines (150-180) in this section were constructed in the same manner, using FOR-NEXT loops to POKE information directly into screen memory. These lines draw continuous lines on the graph, making it more readable.

Designations

Block two of the program prints a series of numbers on the left side of the graph and letter designations for each of the six columns. Line 190 positions the following PRINT statement at the right spot horizontally so the numbers can be displayed along the left side of the graph. We want the numbers to start at the fourth space right of the border, so we place a SPC(4) after the PRINT, and then place the number to be printed inside quotation marks.

So, lines 190 through 220 label the Y-axis with a sequence of numbers from 15 to 0. Notice that between each colon is a complete PRINT statement, and even though they are all crunched together in only four program lines with *no spaces*, they result in 16 lines of vertical display. Also, notice that with one-digit numbers the SPC() statement is increased from four to five in the Commodore 64 version for proper placement.

The last line of this section (230) puts letter designations along the bottom of the graph beneath the columns. Notice there is only one PRINT since this line is displayed horizontally. In the Commodore 64 version, the first letter is positioned with TAB(9) and the following letters are all equally spread with SPC(4) statements. Again, because of the VIC's smaller screen size, a TAB(4) and SPC(2) statement are used to position the letters properly.

READ-DATA Block

In the third block of the program (lines 240 to 300), DATA is READ that will be used in a later routine to position each vertical bar on the graph and decide its color. Line 240 prevents this DATA from being reREAD unnecessarily with any subsequent passes through the program.

The first statement that READs DATA in this section is in line 260. Here, a READ command is contained in a FOR-NEXT loop so it is executed six times. This causes six strings, representing the six column labels (A,B,C,D,E,F) to be READ and set equal to the string array variable, A$(I).

In line 280, a second set of DATA is READ and assigned to D(I). This string array variable denotes the color code for each vertical bar on the graph.

The last group of DATA in this block is READ from line 300. The values taken from line 290 are the screen memory addresses necessary to properly locate each bar on the graph.

The use of arrays in this section significantly shortens the length of the program. Instead of requiring six separate blocks of code to locate and draw each vertical bar, we will now be able to perform this in one routine.

Input Block

The fourth block of the program (lines 310-420) is the INPUT routine. Notice that that much of this routine is contained within a FOR/NEXT loop

Fold over a corner of this page so you find it again.

Just send us a postcard or a letter with your name, address and zip, and we'll send you a FREE copy of our new **Sunsoft**™ Catalog of Fantastic Software.

FREE CATALOG

Box 99
Alturas, FL
33820

sunsoft

Makers of the **Sunsoft**™ IQ™ program that tests IQ.

VIC-20
Software Sale
SAVE UP TO 30%

	List	Special	
ESCAPE MCP	27.95	**$19.95**	Cassette
APE CRAZE	27.95	**$19.95**	Cassette
APPLE PANIC	39.95	**$29.95**	Cartridge
ADVENTURE MANOR	25.00	**$19.95**	Cassette - req. 16K exp.
PRIMARY MATH TUTOR	27.95	**$19.95**	Cassette

Include $3.00 shipping & handling per order.
N.Y. State residents add applicable sales tax.
Send Check or Money Order to:
Jay Sun Enterprizes, Incorporated
2 Linda Court, Albany, NY 12208

Circle Reader Service for
FREE COMMODORE-64 CATALOGUE

COMSTAR AIR* SHIPPING WITHIN 2 DAYS
SEND $1.00 FOR COMPLETE LIST

COMMODORE

GX 100 PRINTER (80 COLUMN) (BANANA)	199
CARDCO PRINTER INTERFACE (PARALLEL)	59
ALIEN GROUP VOICE BOX (D,T)	85
NEWPORT PROSTICK	23
ELEPHANT DISKS (BOX OF 10)	21
RABBIT (EASTERN HOUSE) (VIC OR 64)	35
HES MODEM (WITH SOFTWARE) (VIC OR 64)	59
HES MON ASSEMBLER (C) (VIC OR 64)	29
DUST COVER	7
QUICK BROWN FOX (C) (VIC OR 64)	49

COM-64

WORDPRO 3 + (D)	59
VIDEOPAK 80 (80 COLUMN)	159
Z-80 VIDEOPAK (WITH CPM)	259
6502 PROF. DEV. SYSTEM (T)	23
ROBBERS OF LOST TOMB (D,T)	21
PROG. REF. GUIDE (BOOK)	18
ELEMENTARY 64 (BOOK)	14
64 FORTH (C)	46
JUMP MAN (D,T)	29
PRACTICALC (SPREAD SHEET) (T)	39
FORT APOCALYPSE (D,T)	26
GOTHMOG'S LAIR (D,T)	34
FROGGER (D,T)	26
ANNIHILATOR (T)	18
TEMPLE OF APSHAI (D,T)	29

VIC-20

16K RAM	59
CARDBOARD (3 SLOT EXP.)	33
DISPLAY MANAGER (40/80 COL.)	89
HESWRITER (C)	29
TURTLE GRAPHICS (C)	29
VIC FORTH (C)	46
CHOPLIFTER (C)	29
SHAMUS (C)	29
SCORPION (C)	29
SPIDER CITY (C)	29
SWORD OF FARGOAL (T) 21K	23
KIDS AND THE VIC (BOOK)	17
VIC GRAPHICS (BOOK)	12
SUBMARINE COMMANDER (C)	31
TYPE ATTACK (C)	29

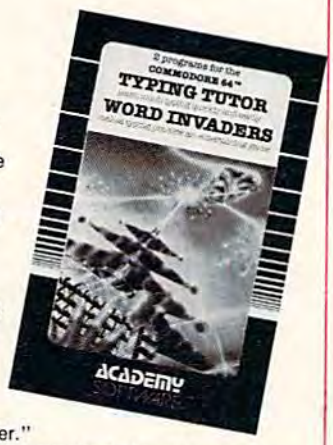
C = CARTRIDGE D = DISK T = CASSETTE * MOST ITEMS

COMSTAR
P.O. BOX 1730 GOLETA, CA 93116
(805) 964-4660

ORDERS ONLY: 800-558-8803
or send check or money order. VISA, MC add 3%. Shipping—$2 for software (call for hardware). Calif add 6% tax. COD add $2.50.

VIC 20™ COMMODORE 64™

Still the Best!



Rated **THE BEST** educational program for the VIC 20™ by ***Creative Computing*** magazine.

Commodore 64 version: "This is the best typing tutor we have seen yet; it can get your children touch typing in short order and bring an old hand up to speed. Includes excellent training modules and an arcade type mode to liven things up and put some pressure on; ★★★★+" **INFO-64**

Our customers continue to tell us of their success. . . .

". . . delighted with my son's progress . . . he is the only one in his second grade class who touch types at the computer."

(58 year old man writes) . . . "great, excellent. To me a source of great learning . . . I just can't express how much I have enjoyed it!"

In daily use by schools across the USA.

"Computer aided instruction at its best" ***Commander*** magazine

TYPING TUTOR + WORD INVADERS

The proven way to learn touch typing.

COMMODORE 64 Tape $21.95
COMMODORE 64 Disk $24.95
VIC 20 (unexpanded) Tape $21.95

NEW!

IFR
(FLIGHT SIMULATOR)

CARTRIDGE FOR THE VIC 20

COMMODORE 64
DISK OR TAPE

$39.95
JOYSTICK REQUIRED

Put yourself in the pilot's seat! A very challenging realistic simulation of instrument flying in a light plane. Take off, navigate over difficult terrain, and land at one of the 4 airports. Artificial horizon, ILS, and other working instruments on screen. Full aircraft features. Realistic aircraft performance — stalls/spins, etc. Transport yourself to a real-time adventure in the sky. Flight tested by professional pilots and judged "terrific"!

VISA Shipping and handling $1.00 per order. CA residents add 6% tax. MasterCard

ACADEMY SOFTWARE

P.O. Box 6277, San Rafael, CA 94903 (415) 499-0850

Programmers: Write to our New Program Manager concerning any exceptional VIC 20™ or Commodore 64™ game or other program you have developed.

(lines 310–370).

In this loop, you are asked what value you want for each column. The value that you INPUT determines the height of each vertical bar. Your response is checked in line 360 to make sure it is within the limits of 0 and 15.

After INPUTting the height of each column, the screen memory address (A) for the top of the column is determined by the first statement in line 370.

Here's how it works: A(J) was set as a starting screen location in the first line, then AA (the response) is multiplied by 40, because our screen is 40 characters across (22 for the VIC version). Then AA times 40 (or 22) is subtracted from A(J), because the columns are drawn upward. So, if the response is 10, the column rises 10 segments high. Then 80 (or 44 on the VIC) is added to A to bring it down two rows so we have room for our three-dimensional side. Program execution is then transferred to the subroutine at line 430, which actually draws each column on the graph.

In the process, the variables necessary to this subroutine are passed. The variable C defined in 310 is the offset between the screen memory map and the color memory map as explained above. The actual color of each column (variable D) and the starting screen location of each column, or variable X, are also transferred.

Once a column has been drawn, the user's previous INPUT is erased in line 320 by POKEing blank spaces into this area of screen memory. If you didn't do this, the prior answer, of course, would remain on the screen.

Line 320 enables you to position a PRINT statement exactly where you want vertically without disrupting any printing already on the screen. The cursor is first HOMEd, and then a blank PRINT statement is placed inside a FOR-NEXT loop. As the loop is executed, starting at the HOME position, it counts down vertically to the maximum number set by the FOR-NEXT loop.

The next line is the INPUT statement, now in the right position to be printed. At the end of the INPUT statement is the variable AA, which receives whatever value you enter between the limits of 0 and 15. If the response is less than 0 or greater than 15, the computer erases the answer and asks the same question again.

Once all six vertical bars have been drawn, you will be asked in line 390 if you wish to do another bar graph. If you do, the program will start again at line 100. Otherwise, it will END in line 420.

The Subroutine

The heart of this program is the subroutine beginning at line 430. This is the block which draws the columns by POKEing symbols onto the screen.

Let's start explaining this section with lines 430 and 440. These two lines check to see if the value AA from the INPUT block is a 1 or 0. If AA = 1, the program branches to line 530, which draws the top of a column one segment high on the graph. When AA = 0, it is a null entry, and the program gets another INPUT.

Lines 470 and 480 begin to actually draw the columns, which are three characters wide. Reflecting back to the INPUT block, you'll recall that variables A and X were set for the starting point and top part of the column. So, by POKEing the screen memory locations with the desired character symbols in a FOR-NEXT loop, we can draw the columns to any height we've chosen. Notice there are three POKEs, I, I + 1, I + 2. Each addition to I moves its location over one spot to the right, yielding a three-character-wide column. The different screen display codes create a three-dimensional appearance with reversed characters. The program reverses the character codes by adding 128 to the symbol code (32 + 128 = 160, 101 + 128 = 229). Line 480 follows up line 470 with the color information by adding the color variables C and D to the same locations from line 470.

Lines 510-540 follow the same format as lines 470 and 480. They draw the three-dimensional top segments of the columns. Six character symbols and six color locations are POKEd into the appropriate locations with the variables A, C, and D. By adding or subtracting numbers from A, we can position the symbols on the row above or to the right.

Once this subroutine is completed, line 550 RETURNs to the INPUT block.

Formatting

By now, you should have "Graph Plotter" typed in and SAVEd on tape or disk. The difference between this program and others you have typed is that you now know exactly how it was programmed. Remember the techniques of using PRINT statements for displaying characters vertically and horizontally; of blank PRINT statements and SPC() commands for positioning INPUT or PRINT statements exactly where you want them; of directly placing symbols and colors onto the screen with POKEs and variables. In planning your own programs, use these techniques for your screen displays and see how handy and time-saving they can be for you.

See program listings on page 202.

Happy Holiday Savings

Commodore 64 $219

Avalon Hill Game Company

B-1 Nuclear Bomber (C) ..... $ 12
Midway Campaign (C) ..... $ 12
North Atlantic Convoy Raider (C) ..... $ 12
Nukewar (C) ..... $ 12
Planet Miners (C) ..... $ 12
Computer Stocks & Bonds (C) ..... $ 15
Andromeda Conquest (C) ..... $ 14
Computer Football Strategy (C) ..... $ 12
Telengard (C) $16

Automated Simulations

Crunch, Crumble, Chomp ..... $ 20
Temple of Apshai ..... $ 27
Jumpman ..... $ 27

Broderbund

David's Midnight Magic ..... $ 23
Choplifter (CT) ..... $ 34
Serpentine (CT) ..... $ 27
Sea Fox (CT) ..... $ 27
Bank Street Writer ..... $ 45

Creative Software

Home Inventory ..... $ 12
Crisis Mountain ..... $ 20

Datamost

Swashbuckler (D) ..... $ 23
Aztec (D) ..... $ 27
Paint Wizard ..... $ 34

Human Engineered Software (HES)

6502 Professional Dev. System (C) ..... $ 23
Retro Ball (CRT) ..... $ 27
Hesmon (CRT) ..... $ 27
Turtle Graphics II (CRT) ..... $ 45
Heswriter 64 (CRT) ..... $ 35
Turtle Tutor ..... $ 27
Paint Brush ..... $ 20

Infocom

Zork I, II, III ..... Ea. $ 27
Deadline (D) ..... $ 34
Starcross (D) ..... $ 27
Witness ..... $ 34
Planetfall ..... $ 34

Sierra On-Line

Frogger (D) ..... $ 25
Crossfire ..... $ 15
Jaw Breaker ..... $ 20
Threshold (CRT) ..... $ 27
Sammy Light Foot (CRT) ..... $ 25

Sirius Software

Blade of Blackpoole (D) ..... $ 27
Type Attack (CRT) ..... $ 27
Repton (D) ..... $ 27
Critical Mass (D) ..... $ 27
Snake Byte (D) ..... $ 23
Bandits ..... $ 27
Squish 'em (CRT) ..... $ 23
Final Orbit (CRT) ..... $ 23

Spinnaker

Snooper Troops #1 (D) ..... $ 30
Facemaker (D) ..... $ 23
Kindercomp (D) ..... $ 20
Hey Diddle Diddle ..... $ 20
In Search of the Most Amazing Thing ..... $ 27
Fraction Fever (CRT) ..... $ 20
Alphabet Zoo (CRT) ..... $ 20
Delta Drawing (CRT) ..... $ 20

Synapse Software

Ft. Apocalypse ..... (D) $ 23, (C) $ 23
Dreibs ..... (D) $ 23, (C) $ 23
Survivor ..... (D) $ 23, (C) $ -23
Pharoah's Curse ..... (D) $ 23, (C) $ 23

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
VT 106A Recreation Pack ..... $ 45
VT 107A Home Calculation Pack ..... $ 45
VIC 1600 Vicmodem ..... $ 89
VM Programmer's Reference Guide ..... $ 14
Commodore Programmer's Ref. Guide $ 18

★★ *Stocking Stuffers* ★★

Koala Touch Tablet (Apple, IBM, Commodore, Atari) $85
Wico "BOSS" Joystick $15
Elephant Disk S/S $18
Flip & File Diskette Box $21

Educational

Books

Kids and the VIC ..... $ 18
Programmer's Reference Guide-VIC ..... $ 14
Programmer's Reference Guide-64 ..... $ 18

Reading & Language Arts

Super Hangman (C) ..... $ 14
Simon/Hess (C) ..... $ 13
Concentration (C) ..... $ 13
Home Babysitter ..... $ 23
Word Search ..... $ 15
Facemaker ..... 64-$ 23 VIC-$ 27
Kindercomp/64 ..... $ 20
Snooper Troops I/64 ..... $ 27

Mathematics

Sky Math (C) ..... $ 12
Space Division ..... $ 12
Bingo Speed Math (CT) ..... $ 23
Number Crunch (CT) ..... $ 23
Number Chaser ..... $ 17
Number Gulper ..... $ 17

Music

Fun with Music ..... $ 27
VIC Music Composer (CT) ..... $ 42
HES Synthesound (CT) ..... $ 49

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
Kids on Keys/64 ..... $ 23

Social Studies/Science

Visible Solar System ..... $ 23
Reagonomics (CT) ..... $ 27

Commodore Software

Avenger ..... $ 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 ..... $ 56

1-800-634-6766

Call Toll Free — Order Line Only — Ask for Operator CO64

Information & Order Inquiries
(702) 369-5523

We Accept VISA & MasterCard

COMPUTER OUTLET

1095 E. Twain, Las Vegas, Nevada
(702) 796-0296

VIC 20 $99

Avalon Hill

Tank Arcade ..... $ 12
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
Sky Blazer (CT) ..... $ 27
Sea Fox (CT) ..... $ 27
A.E. (CT) ..... $ 27

Creative Software

Black Hole (CRT) ..... $ 36
Astroblitz (CRT) ..... $ 36
Apple Panic (CRT) ..... $ 36
Serpentine (CRT) ..... $ 36
Choplifter (CRT) ..... $ 36
Terraguard (CRT) ..... $ 36
Household Finance ..... $ 20

HES Software

HES Mon (CT) ..... $ 29
HES Writer (CT) ..... $ 29
Synthesound Music Synthesizer (CT) ..... $ 49
Turtle Graphics (CT) ..... $ 29
VIC Forth (CT) ..... $ 45

Sierra On-Line

Apple Cider Spider (CT) ..... $ 27
Sammy Lightfoot (CT) ..... $ 27
Frogger ..... $ 27

Sirius

Type Attack ..... $ 27
Snake Byte ..... $ 27

Thorn EMI

River Rescue (CT) ..... $ 27
Submarine Commander ..... $ 27

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) $ 23
Alien Blitz ..... (C) $ 17 (CT) $ 27
Super Hangman (C) ..... $ 14
The Alien (C) ..... $ 17
Kosmic Kamikaze (C) ..... $ 17
Sub Chase (C) ..... $ 15
Renaissance ..... $ 20
Cloud Burst (CT) ..... $ 27
Satellites & Meteorites (CT) ..... $ 34
Outworld (CT) ..... $ 34
Wordcraft ..... $ 65

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. Charges for C.O.D. orders are $3.00 minimum or 1% for orders over $300. School purchase orders welcomed. Prices reflect a cash discount only and are subject to change without notice. Please enclose your phone number with any orders.* ***SHIPPING — Software:*** *$2.50 minimum.* ***SHIPPING — Hardware:*** *(Please call)* ***SHIPPING — Foreign Orders:*** *APO & FPO orders: $10 minimum and 15% of all orders over $100. Nevada residents add 5 3/4% sales tax. All goods are new and include factory warranty. Due to our low prices, all sales are final. All returns must be accompanied by a return authorization number. Call 702-369-5523 to obtain one before returning goods for replacement.*

MACHINE LANGUAGE FOR BEGINNERS

RICHARD MANSFIELD, SENIOR EDITOR

Addressing

You'll soon notice that most of your ML programming involves sending bytes around in the computer's memory. It's quite similar to PEEKing and POKEing, but you've got more options on how you *address* these bytes before you send them somewhere. Addressing bytes is like addressing a letter—you want it to get to its destination so you must write the destination on the letter. There are even ways to send the byte c/o another address, but we'll get to that in a minute. First, we need to review our all-ML game in terms of some addressing options we can use when writing an ML program.

A Cumbersome List

So far, our ML game can be divided into three sections (like subroutines). Let's use the 64 version, Program 2, as our example this month; it's essentially the same as the VIC version. The first section (from address 49152 through 49169) puts the number 8 into all the addresses of Color RAM memory. We usually have a choice of which ML *addressing mode* we want to use. We could have used the simplest mode, absolute addressing, and just listed every address we wanted to POKE the 8 into. It would have looked like this:

```
STA 55296
STA 55297
STA 55298
STA 55299 and so on
```

but that's pretty inefficient. We would have had to list a thousand addresses. Instead, we chose to do our POKEing within a loop. The Y register is quite useful for addressing things because it can be used as an *offset*. That is, you can address something so that the actual address you give is *added to whatever Y equals at the time*. This is a special form of absolute addressing called *absolute indexed* (you can use the X register this way too).

How does it work? First we set Y to equal zero (at 49152). Then we load the A register with our color value, 8. Then we have four STAs lined up, using the absolute indexed addressing mode. The first time the computer comes across this list, it will put 8 into 55296, 55552, 55808, and 56064. It will add Y to these addresses, but Y equals 0 this first pass through the loop. Then, we INY (raise it by one). The three registers (A,X, and Y) can only count up to 255; after that, they reset themselves to 0. So, when Y = 1 after the INY, the BNE instruction will "fail" and we'll branch back to 49156 for the second pass through the loop. We can't get past BNE until Y resets itself to zero—BNE means Branch Not Equal (to 0). And Y isn't yet reset to 0.

But, notice what happens the second time through the loop. Since Y now equals 1, we'll be putting our 8 into 55297, 55553, and so on. This storage of 8's continues until all the locations between 55296 and 56319 have been filled.

The Most Common ML Bug

The second section of our game (lines 49171 through 49184) is quite similar and makes use of the same addressing mode. But here we're drawing a horizontal line across the top and bottom of the screen. So, since the screen is only 40 bytes wide, we'll have to test Y (line 49182) to see if it's equal to 40. If not, we BNE back and continue the loop. There are several "B" instructions; all of them begin with the letter B and branch somewhere (if

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.

But the problem is, most software's been designed for toys. Not for real computers. Until now.

Commodore 64 and VIC-20. Not just for games...

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).

See us at CES Booth #6940

conditions pass their test). BEQ means Branch if EQual to 0. We'll get to the others in the future. BEQ and BNE, though, are by far the most commonly used ones.

The other instructions here are also the most frequently used ML commands. STA (STore the A register), LDA (LoaD the A register), STY, LDY, CPY (ComPare Y), INY (raise Y register by 1; literally INcrement Y), DEY (reduce Y by 1, literally DEcrement Y), and their companion instructions (DEX, INX, LDX, STX, CPX) all operate according to the same rules (and set up flags for the "B" instructions to test and then decide whether or not to branch).

One other thing to notice here: The computer will always assume that you are loading *from an address* unless you specify otherwise. If you write LDA 15, the computer copies whatever is in address 15 into the A register. (Whatever was at address 15 remains there; only a copy of it is placed into the A register.) So, if you want to actually put the *number 15* itself into the A register, you must put a number sign in front of it: LDA #15. As you can see, we do this frequently in our program. But beware—the single most common source of ML bugs is forgetting to put in that # when you mean a number as such, or putting it in when you mean to get a copy of a number from some memory location in the computer.

Sending Something C/O

Now on to the new portion of our ML game (lines 49186 to the end). Here we are going to draw vertical lines down the sides of the screen, completing the frame for our game. To do this, we'll need to POKE the first address of screen RAM memory, then POKE the 39th (the top of the right-hand margin), then the 40th (the second space down on the left margin), then the 79th, and so on down. The easiest way to do this in ML is to send a character to the right side of the frame, add 1 to Y (INY) and send the character again, this time to the left side of the screen. Then we must add 40 to our address (22 on the VIC) to get over to the right side again.

As before, we'll set up a loop, but we first have to prepare two bytes in zero page (the first 250 memory cells of the computer). These two bytes will hold our addresses in a special way. We'll change the contents of these bytes as we go along, adding 40 (or 22). You can visualize these special bytes in zero page as a Ping-Pong paddle and, by shifting its angle, you can send the balls low or high or anywhere in between.

Working In The Real World

Our 6502 chip can send things to 65536 memory cells, but how can we store a number that large when each cell will only hold numbers up to 255? It's simple enough: We gang two bytes together to hold large numbers. Take a number like 1024, the start of the 64's screen memory RAM. Divide it by 256 and you get 4. So put that in one of the two bytes holding our number (call it the *most significant byte*, or MSB). Then put the remainder of the division (0 in this case) into the other byte, the *least significant byte*, or LSB. Line them up in memory as LSB/MSB (it's backwards to us, but the computer likes them put in this way). There you have it. Notice that the Assembler program performs this whole task for you automatically when you type in a large address (as in line 49156).

How does this work in the real world? We do need to set up just such a double-byte address for our routine which draws vertical lines. We're going to use the two bytes at 71–72 (space that's not being used by the computer during an ML program run). Notice that we must use zero page for setting up our special c/o addressing method. We start off (line 49186) by putting 39 into the LSB, address 71. Then we put a 4 into 72, the MSB. Since the real address (the target) is 256 × LSB + MSB, we'll get 1063 as the target when we land on these two bytes. 1063 is the first space on the right-hand side of our vertical screen line. We can start there because the first line is already filled in anyway with our horizontal line, built earlier in the program.

At this point we can formally introduce one of the most significant and useful of the ML addressing modes—*Indirect Y*. (It's usually called *indirect indexed*. Who can remember that? Let's call it Indirect Y.) It takes a minute to get it straight, but it's a minute well spent. You'll find many uses for this handy method of sending bytes anywhere in the computer. When you address something this way, it "bounces off" the number you prepare in zero page, it's indirect, it's like sending a letter c/o someone. In effect, it gets readdressed once the computer lands down in zero page.

TELSTAR 64

Sophisticated Terminal Communications Cartridge for the 64.

PFO 10D 00D CP D1 D2 BELL 12:30:00 10:14:36
(TELSTAR's Status Line)

Don't settle for less than the best!

- Upload/Download to/from disk or tape.
- Automatic File Translation.
- Communicates in Industry Standard ASCII.
- Real-Time Clock plus Alarm Clock.
- Line editing capability allows correcting and resending long command lines.
- 9 Quick Read functions.
- Menu-driven.
- Similar to our famous STCP Terminal package.
- Works with Commodore Modems and supports auto-dialing.

The best feature is the price — **only $49.95** (Cartridge and Manual)

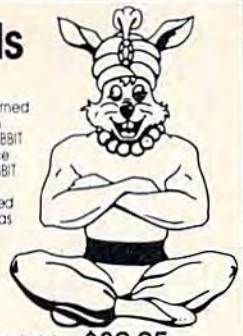
Machine Language Monitor Cartridge
for the CBM 64

More than 20 commands allow you to access the CBM 64's Microprocessors Registers and Memory Contents. Commands include assemble, disassemble, registers, memory, transfer, compare, plus many more.

Someday every CBM 64 owner will need a monitor such as this.

Cartridge and Manual — **$24.95**

8K in 30 Seconds
for your VIC 20 or CBM 64



If you own a VIC 20 or a CBM 64 and have been concerned about the high cost of a disk to store your programs on worry yourself no longer. Now there's the RABBIT. The RABBIT comes in a cartridge, and at a much, much lower price than the average disk. And speed ... this is one fast RABBIT. With the RABBIT you can load and store on your CBM datasette an 8K program in almost 30 seconds, compared to the current 3 minutes of a VIC 20 or CBM 64, almost as fast as the 1541 disk drive.

The RABBIT is easy to install, allows one to Append Basic Programs, works with or without Expansion Memory, and provides two data file modes. The RABBIT is not only fast but reliable.

(The Rabbit for the VIC 20 contains an expansion connector so you can simultaneously use your memory board, etc.)

$39.95

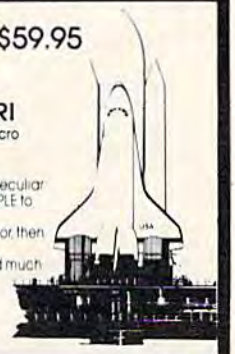
MAE NOW THE BEST FOR LESS!
$59.95

For CBM 64, PET, APPLE, and ATARI

Now, you can have the same professionally designed Macro Assembler/Editor as used on Space Shuttle projects.

- Designed to improve Programmer Productivity
- Similar syntax and commands – No need to relearn peculiar syntaxes and commands when you go from PET to APPLE to ATARI
- Coresident Assembler/Editor – No need to load the Editor, then the Assembler, then the Editor, etc.
- Also includes Word Processor, Relocating Loader, and much more
- Powerful Editor: Macros, Conditional and Interactive Assembly, and Auto – zero page addressing

Still not convinced, send for our free spec sheet!



Eastern House

3239 Linda Dr.
Winston-Salem, N.C. 27106
(919) 924-2889 (919) 748-8446
Send for free catalog!

VISA MasterCard

Software Gallery MAIL ORDER

WE'RE THE ONLY SOFTWARE AND COMPUTER STORE YOU'LL EVER NEED!

P.O. BOX 3354, CHERRY HILL, N.J. 08034

TOLL FREE (800) 992-3300 FOR ORDER ONLY

FOR INFORMATION & N.J. CALL (609) 424-7106

Disk Drive 1541

For the Vic 20 and CMD64. Fast high capacity storage and retrieval of data

$249

CMD64 Computer	$229
VIC 20 Computer	$99
1525 Printer	$215
1526 Printer	$319
1530 Datasette	$64
1600 Modem	$59
1650 Modem	$88

1110	8K Memory Expander	$52
1111	16K Memory Expander	$68
1011	RS 232 Terminal Interface	$42
1211	Super Expander	$53
1212	Programmer's Aid Cartridge	$39
1213	Machine Language Monitor	$39
1311	Joystick	$7
1312	Game Paddles	$12

Color Monitor 1701

14" screen, outstanding resolution, special Commodore computer circuit

Lower Price **$249**

PRINTERS

EPSON

RX-80	CALL
MX-80 FT	CALL
FX-80 & FX-100	Call

SMITH CORONA

TP-1	$525

OKIDATA

82 A	$395	92	$495
83 A	$634	93	$809
84 P	$949		

STAR

GEMINI 10	$314
GEMINI 15	$449

C-ITOH

PROWRITER	$360
PROWRITER 2	$645
STARWRITER	$1199
PRINTMASTER	$1545
GORRILA	$209

CARDCO

Parallel Printer Interface	$57
16K Expander	$59
6 Slot Expander Interface	$72
3 Slot Expansion Interface	$31
Video Pak 80	$150

FLOPPY DISK

Elephant SS/DD(10)	$25
Elephant SS/SD(10)	$18

Surge Protector

4 OUTLET	$37.85
6 OUTLET	$62.85

DATA 20

Z80 VIDEO & CPM	$249

MONITORS

AMDEK (300)	$154.95
COLOR I	$299.95
COLOR II	$589.95

WICO

The Boss	$15.95	Joystick	$22.50
Famous Red Ball			$25.95
Track Ball			$49.95

COMMODORE 64 TOP 20 SOFTWARE

Frogger	$25	Motor Mania	$22	Hes Mon	$28
Zork I	$27	Zork 3	$27	Heswriter	$34
Jumpman	$29	Grid Runner	$27	Facemaker	$26
Temple of Apshai	$29	Jawbreaker	$22	Wordpro 64	$77
Fort Apocalypse	$26	Starcross	$28	Shamus	$26
Deadline	$35	Suspended	$35	Quick Brown Fox	$48
Zork 2	$27	Repton	$29		

VIC 20

Shamus	$26
Choplifter	$29
Gortek	$21
Grid Runner	$27
Gold Fever	$27
Spider City	$28
Side Winder	$22

COMPUTER COVERS $6.99 ea. Reg. $15.95

Features heavy duty canvas with vinyl interior ... waterproof

CMD64	$6.99
Vic 20	$6.99
Disk Drive	$6.99
	$7.99
	$7.99
	$7.99

We Carry a Large Selection of Commodore Software, call for FREE catalog with order ... or send $1 (refundable with order) UPS shipping 3% for hardware and $3 for software. N.J. resident add 6% sales tax. Note: Atari, Commodore, Epson, IBM, and Okidata are Registered trademarks

VISA MasterCard

PRICES SUBJECT TO CHANGE WITHOUT NOTICE — MASTER CARD OR VISA ADD 3%

Program 2: 64 Version

```
49152  LDY   # Ø
49154  LDA   # 8
49156  STA   55296 ,Y
49159  STA   55552 ,Y
49162  STA   55808 ,Y
49165  STA   56064 ,Y
49168  INY
49169  BNE         49156
49171  LDY   # Ø
49173  LDA   # 224
49175  STA   1024 ,Y
49178  STA   1984 ,Y
49181  INY
49182  CPY   # 40
49184  BNE         49175
49186  LDA   # 39
49188  STA   71
49190  LDA   # 4
49192  STA   72
49194  LDX   # 24
49196  LDY   # Ø
49198  LDA   # 224
49200  STA ( 71 ),Y
49202  INY
49203  STA ( 71 ),Y
49205  DEX
49206  BEQ         49224
49208  CLC
49209  LDA   71
49211  ADC   # 40
49213  STA   71
49215  LDA   72
49217  ADC   # Ø
49219  STA   72
49221  JMP    --->   49196
49224  RTS
```

After we load Y (our offset) with 0 again and load A with the framing character (224), we can store the 224 into the address which the computer finds by looking at the double-byte number we put into addresses 71–72. When it sees STA (71),Y—the computer knows what to do. It first calculates the correct target formed by multiplying whatever it finds in cell 72 × 256 and then adding whatever's in cell 71. Then, it also adds the value of the Y register.

So, we can manipulate the number in Y here the same way that we used it with Absolute Indexed above (line 49156), but have the added advantage of being able to manipulate the double-byte address at 71–72 as well. The first time through this loop, the framing character will be sent to 1063. Then we INY and send another framing character to 1064 (the second space down the left side of the screen). Then we DEX. X is counting down from 24 because there are 24 spaces down each side of the screen that we need to fill. If the DEX causes X to equal 0, then the BEQ takes effect and sends us back to BASIC mode via the RTS (ReTurn from Subroutine) at line 49224. If X is not yet zero (and thus we want to continue the looping), we will add 40 to the double-byte number at 71–72.

The adding is done by first clearing the carry, CLC, and then putting the number from 71 into the A register, adding 40 (ADC means ADd with Carry) and then storing the result back into cell 71. Likewise, we get the number from 72, add it to 0, and put it back. Why add to 0? Because there might be a carry from the operation on the number in 71. If so, we need to reflect that in the overall number by adding it to the MSB (in cell 72). After we've added 40 to this special double-byte number, we just jump (JMP) back to the line where we start our loop that prints the framing characters to the screen.

We've covered a good bit of ground this month. You should try out these routines with your Assembler and run them after they've been placed into memory: SYS 12288 (VIC) or SYS 49152 (64). Then, change some of the numbers and see the effects. Try using a different character for the frame. Pay particular attention to the way that Indirect Y addressing accomplishes its effects—we'll be using it frequently from here on.

See program listings on page 201.

For VIC - 20

16K MEMORY EXPANSION

ONLY $49.95

- ★ 14 Day Money Back Guarantee
- ★ Boosts VIC to 21K RAM
- ★ Top Quality, Fully Tested
- ★ 90 Day Warranty

EPYX 16K ADVENTURE GAMES	
Temple of Apshai (T)	27.50
Sword of Fargoal (T)	21.80
MICRODIGITAL ARACADE GAMES	
Snakman (T) (Excellent PacMan Style)	10.75
Gridder (T) (fast chase game)	10.75
Skramble (T) (Thrilling Arcade Action)	10.75

for IMMEDIATE SHIPMENT on Credit Card Orders

Call: (303) 245-9012

10 AM - 9 PM MST Every Day

ASSEMBLY TECHNOLOGY

2692 Hwy 50 Suite 210

Grand Junction, CO 81503

Personal checks allow 3 weeks
Shipping & handling $2.50
Colorado Residents add Sales Tax
COD add $2.00

HUNDREDS OF PROGRAMS AVAILABLE FOR THE COMMODORE 64 & VIC 20

Commodore 64

All Prices up to 35% OFF RETAIL

VIC 20

HOME/BUSINESS	
Practicalc (D)	41.50
Bank St. Writer (D)	52.50
Smart 64 Terminal	31.95
Quick Brown Fox (CT)	42.95
Paper Clip (D)	96.25
WordPro SpellRight (D)	72.95
CalcResult Advanced (D)	135.00
Master Type (D)	29.95
Home Acct (Continental) (D)	49.95
EDUCATIONAL	
Facemaker (D/CT)	22.95
Kindercomp (D/CT)	19.95
Delta Drawing (CT)	26.95
The Most Amazing Thing (D)	26.95
Early Games (D/CASS)	22.50
Electronic Party (VIC 20/CASS)	22.50
Square Pairs (VIC 20/CASS)	22.50
Turtle Trax (VIC 20/CASS)	22.50
Hey Diddle Diddle (D)	19.95
Alphabet Zoo (CT)	22.95
Type Attack (D)	29.95

GAMES	
Zork I, II, or III (D)	29.00
Deadline (D)	37.00
Starcross (D)	29.00
Jumpman (D/CASS)	29.95
Spritemaster (D/CASS)	26.95
Neutral Zone (D/CASS)	27.50
Temple of Apshai (D/CASS)	29.95
HARDWARE AND ACCESSORIES	
Cardco Printer Interface	62.95
Printer Utility (Cardco)	15.95
The Connection Parallel Int.	89.95
Cardco 3 Slot Exp. Board	31.95
Cardco 5 Slot Exp. (C-64)	55.95
Data 20 Video Pak 80 (C-64)	143.95
Data 20 8K Display Mgr. (40/80 col. plus Word Proc.)	119.95
0 K Display Mgr. (40/80)	79.95
Zenith 12" Green Monitor	109.00
Brother/Dynax DX-15 (Letter Quality)	485.00
C. Itoh Prowriter 8510AP	375.00
Control Data Diskettes (10)	23.00

Write or call for FREE CATALOG. TO ORDER: CALL 1-714-643-1056

8:00 A.M.-6:00 P.M. PST Mon.-Sat. or send check or credit card number, signature and expiration date. Please include phone number.

CENTURY MICRO PRODUCTS

P.O. Box 2520, Mission Viejo, CA 92690

Visa/Mastercard add 3%. Personal checks allow 2 weeks to clear. CA residents add sales tax. Shipping and handling add $3.00 (hardware extra). Prices subject to change.

80 COLUMNS! 25 LINES

A FULL PROFESSIONAL DISPLAY FOR

Commodore 64 with Screenmaker™

Screenmaker is a video display generator module that plugs into the expansion connector of the Commodore 64.

- ✓ WORD PROCESSING
- ✓ CALCULATIONS
- ✓ BASIC PROGRAMS

Screenmaker provides a B & W video signal that connects to your video monitor to provide a full 80 characters on each line. With Screenmaker, Screen displays will appear the same as the printer output. Trial printouts can be eliminated. Word processing is easier. Forms and reports can be set up faster. Screenmaker features a bank switched memory, 40/80 video switch, and a full character set including graphics.

SCREENMAKER $159.95
Copy-Writer Word Processor... $ 79.95
SCREENMAKER/
Copy-Writer Package $199.95

MICROTECH P.O. Box 102 Langhorne, Pa. 19047

215-757-0284

TAX AID TAX AID

FOR COMMODORE 64™ and VIC 20™

USE TAX AID™ TO PREPARE YOUR INCOME TAX RETURN

Developed by an experienced accounting firm, TaxAid is accurate, easy to use, and comes with a detailed manual. Your tax data is permanently stored on tape or disk. The cost is tax deductible and yearly updates are available.

TAX AID I	TAX AID II	TAX AID III
For Unexpanded Vic 20	For Vic 20 with 16K	For Commodore 64
$19.95	$24.95	$24.95

Prices above are for cassette; add $5.00 for disk version.

NORTHLAND ACCOUNTING, INC.

606-A Second Ave.
Two Harbors, MN 55616

(218) 834-5012

VISA MasterCard

TAX AID TAX AID TAX AID

TAXAID IS A TRADEMARK OF NORTHLAND ACCOUNTING, INC.
VIC 20 & COMMODORE 64 ARE TRADEMARKS OF COMMODORE ELECTRONICS, LTD.

64 BASIC Aid

Harold D. Vanderpool

This extremely useful utility program adds four commands to BASIC and belongs in every programmer's toolbox. The utility itself is written in machine language. To type it in, use the MLX entry program found elsewhere in this issue.

No version of BASIC has everything. No matter what computer you look at, there are things that could be added to customize it for your particular needs. The VIC and 64 have an excellent version of BASIC, Microsoft, which has been popular for years on microcomputers. But if you do a bit of programming, you might find that "64 BASIC Aid" will be among the most valuable utility programs in your library—it adds four extremely useful commands to the 64's BASIC.

It uses up very little of your RAM memory (about 1000 bytes) and after you've typed in and saved a copy, you can use 64 BASIC Aid anytime you want those extra four commands. You LOAD it and RUN it as you would any other program, but it hides itself high up in memory and becomes invisible. You can then program as always, but you've got those four extra commands available to you.

Since these commands are useful for writing and debugging programs, they are available to you only in *direct* mode. (You can't include them in a program itself, but you'd never have reason to use them that way.)

Four Programming Aids

NUMBER 100,10. With this command, you can renumber any program that's in your computer. Just type the command and press RETURN. The new version of the program will start with line 100 and go up from there in steps of ten. You can use any numbers you want as the starting line number and any number from 1 to 255 as the step size. This can be useful in several ways. For example, you might have used up all the line numbers somewhere in your program: you've got lines 25, 26, 27, 28, 29, and so forth. No room for new numbers to insert a line? Just use NUMBER and they'll be spread apart instantly for you.

Within your program, there probably will be GOSUBs or GOTOs or other references to existing line numbers. 64 BASIC Aid takes care of that, adjusting the references automatically. However, if it finds a GOTO that's targeted to a line that doesn't exist in the program, it will print the number 65535 on the screen. This is helpful when you debug your programs. Also, all adjusted lines will be printed on the screen.

DELETE 100–200. When you type this, all the lines between 100 and 200 (inclusive) will disappear from your program. It works the same way that the LIST command works, using the same format. But be careful with this one. If you just type DELETE without any line numbers after it and then hit RETURN, it will delete the whole program.

FIND/GOTO/,500–900. This would print a list on screen of each line between 500 and 900 which contained a GOTO command. Again, you can indicate how you want the line numbers handled in the same way, using the same options, as with the ordinary LIST command. If you want a

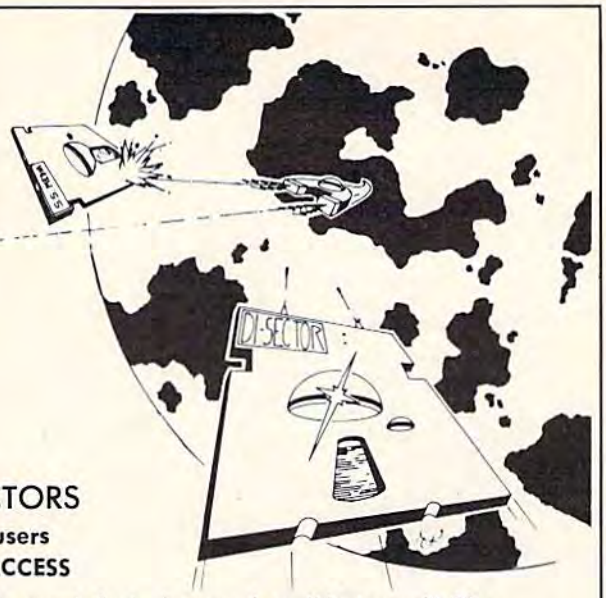
DI-SECTOR

"To The Rescue"

REPAIR - CUSTOMIZE - LEARN

from

STARPOINT SOFTWARE



AT LAST!! DIRECT INTERACTION with SECTORS

an absolute necessity for disk-based Commodore 64™* users

EXAMINE, MODIFY, and RESTORE by DIRECT SECTOR ACCESS

- ★ simultaneous HEX and ASCII display of sectors
- ★ dual cursor editing
- ★ modify in HEX, ASCII or DECIMAL
- ★ restore scratched/deleted files
- ★ repair lost, damaged, or "destroyed" files
- ★ complete operating instructions
- ★ many useful data recovery and modification tips
- ★ source code fully commented by author

Dealer Inquiries Invited

. . . WRITE OR PHONE . . .

STARPOINT SOFTWARE

Star Route 10 Gazelle, CA 96034 (916) 435-2386

$39.95
on 5¼" diskette

VISA/MASTERCARD accepted

C.O.D. add $2

Program by Bruce Q. Hammond
Ad Art by Steven M. Hettema

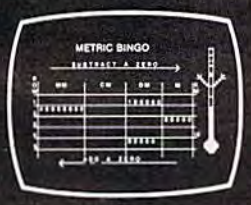
*Commodore 64 is a registered trademark of Commodore Business Machines

---

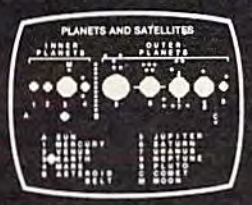
ONLY $18.95!

GIVE YOUR CHILD A HEAD START!

NEW! FOR COMMODORE 64™



METRIC FEVER
Ages 8-13



SEARCHER OF THE SOLAR SYSTEM
Ages 7-13

RACEWAY U.S.A.
Ages 8-12

MATH MANIA
Ages 6-10

Sungem® Co.

EDUCATIONAL GAMES

- ***Designed by Experienced Educators***
- ***Programmed in Machine Language***
- ***Fast & Smooth Action, Maximum Graphics & Sound***

Each of our educational programs targets a specific learning need, begins with step-by-step interactive learning activities, and ends with a reinforcing game. Our educational software advances the gifted child and motivates the underachiever.

☐ **METRIC FEVER**—Makes it fun to learn the metric conversions of linear, capacity and weight or mass measurements.

☐ **SEARCHER OF THE SOLAR SYSTEM**—Teaches meanings of astronomical terms, order of the inner and outer planets and satellites, many planetary facts.

☐ **RACEWAY U.S.A.**—Identifies each state by shape & name, locates capitals, teaches state nicknames and important points of interest.

☐ **MATH MANIA**—Designed with different levels of mastery for addition, subtraction, multiplication & division; many unique games (drills).

SUNGEM
Educational Games
Dept. CG 84, 900 W. Los Vallecitos
San Marcos, CA 92069 (619) 744-2011

START MY CHILD LEARNING TODAY!

☐ **Order by mail or phone.** Each educational game comes in either Diskette ($18.95) or Cassette ($15.95). VISA and MASTERCARD welcome. Include $1.50 postage & handling; CA Residents add 6% Sales Tax.

☐ **Send me your free Color Brochure.**

NAME _____

ADDRESS _____

CITY _____ STATE _____ ZIP _____

report on the whole program, just leave off the comma and the line numbers. You can search for anything: variables, strings, commands, numbers. One note, however, about looking for things in quotes. The computer won't know if you want the *word* "to" or the *BASIC command* TO unless you use quotation marks instead of the slashes when you're looking for words as literal words. So, to find the word "to" you'd type:

```
FIND"TO",500-900
```

CHANGE@PRINT@PRINT#4 @,300–400. Similar to FIND, CHANGE will both locate and *replace* all occurrences of something within your program. All the rules for FIND apply the same way to CHANGE. The example here could be very useful if you have a printer. As written, your program is designed to PRINT everything to the screen. But you can make everything go to the printer instead by adding a line at the start of the program, OPEN 4,4 which alerts the computer that a channel has been opened to the printer. Then this CHANGE will make all printing go to channel 4 (Commodore printers are always Device #4, by convention) instead of the screen.

Another use for CHANGE would become apparent if you'd written a large program and used an illegal variable name like TI (reserved for the clock) or TO (a command name). Instead of hunting through the program, trying to find each illegal variable, just SAVE the program, LOAD and RUN 64 BASIC Aid, LOAD the program back in, and type: CHANGE/TI/TR/ and it's fixed in a flash. Like FIND, the whole program is changed if you leave off the line number information.

KILL turns off 64 BASIC Aid. If you want the computer to be returned to its original state, just type KILL and everything will be as if you'd just turned it on.

Here's the information you need to type in 64 BASIC Aid with the special MLX machine language entry program found elsewhere in this issue:

Start address = 39852
End address = 40961

To activate 64 BASIC Aid once it's loaded, enter SYS 39852.

If you don't want to type this program, send $3, a blank cassette or 1541/4040 disk, and a self-addressed, stamped mailer to:

Harold D. Vanderpool
3705 Enon Xenia Pike
Enon, OH 45323

See program listing on page 203.

Bug-Swatter:
Modifications And Corrections

● Two changes are necessary in the 64 version of "Oil Tycoon" (October). Add the following lines to pick a difficulty level or to press E to end the game:

```
230 PRINT"{DOWN} DIFFICULTY LEVEL? 123456
    78E(END){GRN}":T=1                :rem 76
235 POKE56194+T,0:T1=T:T=T+(PEEK(JS)AND4)
    /4-(PEEK(JS)AND8)/8:IF T>9 THEN T=1
                                      :rem 157
237 IF T<1 THENT=9                    :rem 229
263 IF T=9 THEN SYS2048               :rem 240
```

When converting the VIC version to the 64, we also neglected to include the feature which allows you to replace the oil rig by pressing the fire button. Add these lines:

```
435 Q=R-1:IF Q=0 THEN 360             :rem 41
437 J=Z:FORA=0TO21:A%(A)=0:NEXT:B=Q*40+W+
    X:A=PEEK(B):IF A<>7 THEN 480 :rem 255
```

● The text-adventure game "Martian Prisoner" (November) does not respond properly when you attempt to hit a Martian guard. This bug escaped our testing because we found it safer to fool the guards rather than provoke them (hint). Nevertheless, if you want to hit the guards, insert a GOSUB command in line 125:

```
125 IFV=14 THEN GOSUB 3900        :rem 154
```

● Many of you who typed in the VIC version of "Aardvark Attack" (October) encountered a syntax error in line 55. The reason is that the programmer used a keyword abbreviation when he wrote the line, and it works fine when typed in with the abbreviation. That's why our testing detected no errors. Unfortunately, listings do not show abbreviations. Here is the line:

```
55 M=(TANDFNM(J))/4 J....          :rem 77
```

The computer interprets T AND FNM(J) as TAN (tangent) without a value. There are four ways to fix it: (1) insert a space between T and AND; (2) abbreviate AND with A-SHIFT-N; (3) put the T in parentheses; (4) reverse the order—FNM(J)ANDT.

● The 64 version of "Munchmath" (November) does not accept answers beginning with 9. To fix this, remove ORAN$>"9" from line 200.

● In November's "Machine Language For Beginners" column, the VIC version of "The Assembler" program requires an 8K memory expander.

LIST Freezer

Doug Ferguson

This very short routine will prove indispensable to BASIC programmers—it allows you to pause or freeze a LISTing of the program on the screen. The routine is a machine language program presented in the form of a BASIC loader, so you need to know nothing about machine language to use it. What's more, it works equally well on the Commodore 64 or VIC-20.

The VIC-20 and Commodore 64 cry out for a pause feature during a LIST. When you're writing or debugging a program, especially if you lack a printer, you can waste a lot of time typing LIST again and again just to get a look at your BASIC code.

"LIST Freezer" is an elegant solution to the problem. It patches directly into the LIST routine in ROM (Read Only Memory) without interfering with anything else. Once it's activated, there is never any need to turn it off. It also eliminates the screen ripple effect of some other LIST pause routines, including one I published in COMPUTE! Magazine in 1982.

The LIST Freezer

The program below activates the pause feature for either the VIC or 64. Type it in exactly, SAVE it, RUN it, LOAD something in BASIC, and give it a try. (Because it destroys the BASIC loader part of itself in line 80, be sure to SAVE it before typing RUN for the first time.)

To use LIST Freezer, LIST any BASIC program and hold down the SHIFT key. The listing will pause. To freeze it entirely while freeing your hands, press SHIFT LOCK. You can restart the listing at any time by releasing SHIFT or SHIFT LOCK.

Technical Details

For the curious, here's how it works. Line 20 sets the low-byte/high-byte address of a machine language "patch" at the top of RAM (Random Access Memory). The patch consumes 23 bytes of memory on either the VIC or 64.

Line 30 redefines the computer's memory size to protect the patch. It also moves the LIST vector at memory addresses 774–775 (hexadecimal $0306–$0307) to reroute the indirect jump to ROM (address $A717 in the 64 or $C717 in the VIC).

The remaining lines create the patch routine at the top of RAM. Line 50 adjusts the patch to work on either the VIC or 64.

Notice that the program assumes the normal LIST vector at power-up; line 20 thus prevents you from accidentally trying to activate the routine more than once while the power is on.

Also note that the routine clears out the keyboard buffer when activated. Actually this was necessary only for the VIC, but it causes no harm on the 64 and was left in to make the routine universal.

LIST Freezer

```
20 L=232:H=PEEK(56)-1:Q=PEEK(775):IF Q<16
   7 THEN 80                          :rem 236
30 POKE 55,L:POKE 51,L:POKE 56,H:POKE 52,
   H:POKE 774,L:POKE 775,H            :rem 74
40 FOR X=L+H*256 TO X+21:READ D:POKE X,D:
   NEXT                               :rem 51
50 POKE X,Q                           :rem 105
60 DATA 72,152,72,32,159,255,169,1,44,141
   ,2,208,246                         :rem 209
70 DATA 169,0,133,198,104,168,104,76,26
                                      :rem 136
80 NEW                                :rem 82
```

"$u¢h-A-Deal!

Others Claim The Lowest Prices
We'll Prove It. By Beating Anyone's Price!*

COMMODORE 64 SOFTWARE

	List Price	But for You!
EPYX		
DRAGONRIDERS OF PERN (D&C)	$40	$25
Jumpman D&C	$40	$25
Temple of Apshai (D&C)	$40	$25
Upper Reaches Apshai (D&C)	$20	$13
Curse of Ra (D&C)	$20	$13
Starfire & Fire One (D&C)	$40	$29
Jumpman Junior (CT)	$40	$29
Lunar Outpost (D&CT)	$40	$25
Gateway to Apshai (D&CT)	$40	$25
Pitstop (D&CT)	$40	$25
Crush, Crumble, Chomp (C&D)	$30	$19
Fun With Art (CT)	$40	$27
Fun With Music (CT)	$40	$27
Facts (D&C)	$30	$19
Swat Rescue (D&CT)	$40	$25
Silicon Warrior (D&CT)	$40	$25
Swat Rescue (D&CT)	$40	$25
SEGA		
CONGO BONGO (CT)		
Buck Rogers (CT)	$40	$25
Star Trek (CT)		
HESWARE		
MULTIPLAN (D)	$100	$79
Coco (CT)	$45	$29
Maze Master (CT)	$40	$25
Turtle Tutor (CT)	$40	$26
Turtle Trainer (CT)	$40	$26
Paintbrush (CT)	$30	$17
Home Manager (D)	$50	$33
Time & Money Manager (D)	$70	$45
Hesman 64 (CT)	$40	$26
Gridrunner (CT)	$40	$26
64 Forth (CT)	$60	$39
Turtle Graphics II (CT)	$60	$40
Omnicalc (D)	$50	$37
Attack Mutant Camel (CT)	$30	$19
INFOCOM		
ENCHANTER (D)		
Planet Fall (D)		
Witness (D)	$50	$33
Infidel (D)		
SYNAPSE		
BLUE MAX (D&C)		
Protector (D&C)		
Shamus (D&C)		
Sentinel (D&C)		
Morgol (D&C)		
Drelbs (D&C)	$35	$22
Fort Apocalypse (D&C)		
Pharoah's Curse (D&C)		
Protector II (D&C)		
Survivor (D&C)		
Shamus II (D&C)		
BRODERBUND		
CHOPLIFTER (CT)	$45	$29
Seafox (CT)	$40	$26
Serpentine (CT)	$40	$26
David Midnight Magic (D)	$35	$23
Bank St. Writer (D)	$70	$45
Lode Runner (D)	$35	$23
Spare Change (D)	$35	$23
Drol (D)	$35	$23

COMMODORE 64 SOFTWARE Cont'd.

SPINNAKER		
DELTA MUSIC (CT)	$40	$26
Alphabet Zoo (CT)	$35	$23
Cosmic Life (CT)	$35	$23
Delta Drawing (CT)	$40	$26
Facemaker (CT)	$35	$23
Fraction Fever (CT)	$35	$23
Kids on Keys (CT)	$35	$23
Kindercomp (CT)	$30	$19
Story Machine (CT)	$40	$26
Up For Grabs (CT)	$40	$26

And Incredible Savings On:

DATASOFT ZAXXON (D&C)	Just $25
Data 20 Word Manager (C)	$27
Data 20 Business Manager (D)	$89
Continental Home Accountant (D)	$49
Lightning Mastertype (D)	$27
Sublogic Pinball (C)	$20
Acess Neutral Zone (D&C)	$23
Practicalc (D&C)	$35
Programmable Spreadsheet	$55
Datamost Kids & The 64 (book)	$14
Commodore 64 Ref. Guide (book)	$17
Access Beachead (D&C)	$23
Datasoft Pooyan (D&C)	$19
Datamost Swashbuckler (D)	$25
Datamost Aztec (D)	$25
Sierra OnLine Threshhold (CT)	$27
MicroProse Hellcat Ace (D)	$19
MicroProse Spitfire Ace (D)	$19
MicroProse Floyd of Jungle (D)	$19
Sirius Wayout (D)	$27
Sirius Gruds in Space (D)	$23
Sirius Bandits (D)	$27
Sirius Type Attack (D)	$27
Sirius Blade Blackpool (D)	$27
Sirius Wavy Navy (D)	$22
Timeworks Data Manager (D&C)	$17
Timeworks Money Manager (D&C)	$17
Timeworks Programmer Kit (D&C)	$17
Timeworks Programmer Kit II (D&C)	$17
Timeworks Star Battle (D&C)	$17
Timeworks Programming Kit 3 (D&C)	$17
Timeworks Robbers Lost Tomb (D&C)	$17
Timeworks Spellbound (D&C)	$17

"$u¢h-A-Deal! $oftware."
903 S. Rural Rd. #102
Tempe, AZ 85281

CALL TOLL FREE:
1-800-431-8697
For Information Call (602) 968 9128
We sell mail order only!
Dealer Inquiries Invited

COMMODORE VIC 20 SOFTWARE

HESWARE		
ATTACK MUTANT CAMEL (CT)	$30	$19
Predator (CT)	$30	$19
Coco II (C&D)	$20	$14
Lazer Zone (CT)	$30	$19
Necromancer (CT)	$30	$19
Pharoah's Curse (CT)	$30	$19
Kindercomp (CT)	$40	$26
Facemaker (CT)	$40	$26
Story Machine (CT)	$40	$26

COMMODORE VIC 20 SOFTWARE Cont'd.

	List Price	But For You!
SEGA		
BUCK ROGERS (CT)	$40	$25
Congo Bongo (CT)		
Star Trek (CT)		
BRODERBUND		
Lode Runner (CT)		
A.E. (CT)		
Seafox (CT)	$40	$24
Mastertype (CT)		
EPYX		
TEMPLE OF APSHAI (C&D)	$40	$24
Sword of Fargoal (C)	$30	$19
SIERRA ON LINE		
CANNONBALL BLITZ (CT)		
Lunar Leeper (CT)		
Crossfire (CT)	$30	$18
Threshhold (CT)		

Rock Bottom Prices on Peripherals!

MONITORS	
BMC 12" Green	Just $79
BMC 12" HiRes Green	$125
BMC 12" Amber	$89
BMC 12" HiRes Amber	$129
BMC 13" Composite Color	$249
Monitor Cable with Above	$10
DATA 20	
Vic 40-80 Display Manager	$79
C64 Video Pak 80 includes Word Manager Free	$139
Parallel Printer Interface	$45
CARDCO	
Vic 3 Slot Motherboard	$26
Vic 6 Slot Motherboard	$65
Printer Utility Software	$17
Numeric Keypad C64	$29
C64 5 Slot	$45
Graphic Printer Interface	$69
Economy Printer Interface	$39
WICO JOYSTICKS	
The Boss	$14
Bat Stick	$19
Red Ball	$21
Track Ball	$30
KOALAPAD	
Vic, C64, Atari	$79
HESWARE	
Hescard 5 Slot Vic	$39
HesModem Vic & C64	$49
PRINTERS	
Alphacom 40 Column	$119
Alphacom 80 Column includes Vic, C64, Atari Cable	$179
Cardco DM1 Two Color Impact	$119
Gemini 10X	$295

Okidata, Epson, Citoh
call 968 9128

*TERMS OF GUARANTEE: If you find a price for any software or peripheral in this issue that is lower than our advertised price, we'll guarantee to beat it! Valid only on products appearing in our ad. Valid only on product in similar in-stock conditions. Valid only on prices appearing in print, in this issue.

ORDERING & TERMS: Send cashier check, money order, personal/company checks allow 3 weeks bank clearance. VISA/MasterCard accepted. Provide phone number with order. SHIPPING: Software add $3.00. Hardware add $300 or 3% whichever is greater. Arizona residents add 6% sales tax. Returns must have authorization number (call 602-968-9128) for authorization number). All returned merchandise subject to restocking fee and **must** come with **all** original packaging. No returns allowed after 30 days from shipping dat[illegible] Prices subject to change without notice. All products subject to avaiability from manufacturers and/or suppliers.

Centering VIC Screens

Mary Conlin

If you've discovered a clever time-saving technique, or a brief but effective programming shortcut, send it in to "Hints & Tips," c/o COMPUTE!'s GAZETTE for Commodore. If we use it, we'll pay you $35.

Some VIC-20s and TV sets don't match up perfectly — the screen image appears to be a little off-center. This is called *overscan*. If you can't compensate by fiddling with the TV controls — or if you prefer to leave the TV controls as they are for regular TV viewing — there's a way to adjust the screen from the computer.

Two memory locations inside the VIC control the horizontal and vertical positions of the screen image. By using simple POKE statements in direct mode or within a program, you can quickly adjust the screen for any TV. This method works on a VIC of any memory size. You can restore the screen to its normal position at any time by holding down the RUN/STOP key and pressing RESTORE.

(The Commodore 64 lacks these adjustments, but has much less need for them because its screen image is smaller than the VIC's and is less subject to overscan.)

Horizontal Adjustments

The horizontal screen position is controlled by memory location 36880. Normally this location contains a 5. POKEing smaller numbers into this location moves the screen left, and POKEing larger numbers moves it right. For example, to move the screen one position left, type:

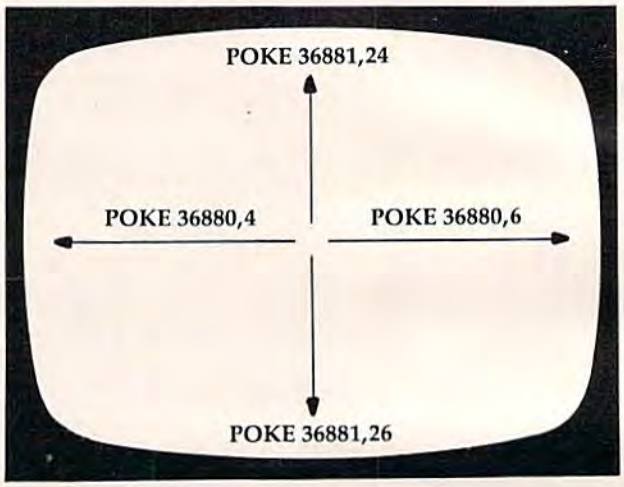
POKE 36880,4 [press RETURN]

Or, to adjust the screen one position right, type:

POKE 36880,6 [press RETURN]

If your screen is off-center by more than one position, try POKEing a 3 or a 7, etc. Once you find the correct value for your particular computer/TV combination, you can include the statement at the beginning of all your BASIC programs so the adjustment is made automatically whenever you type RUN.

Centering VIC Screens On A TV



Vertical Adjustments

The vertical screen position is controlled by memory location 36881. Normally this location contains a 25. POKEing smaller numbers into this location moves the screen up, and POKEing larger numbers moves it down. For example, to move the screen one position up, type:

POKE 36881,24 [press RETURN]

Or, to move the screen one position down, type:

POKE 36881,26 [press RETURN]

Using numbers too high can move the screen completely out of view. If this happens, simply POKE back the 25 or press RUN/STOP-RESTORE. Again, once you find the best number, you can include this statement in all your BASIC programs.

SJB DISTRIBUTORS. ONE STOP SHOPPING FOR COMMODORE SYSTEMS.

commodore

COMMODORE BUSINESS SERIES

SuperPet (5 languages, 2 processors)	$1059
CBM 8032 Computer, 80 Column	625
CBM Memory Expansion, 64K	259
CBM 8050, 1 mg. Dual Drive	995
CBM 8250, 2 mg. Dual Drive	1295
CBM D9060, 5 mg. Hard Disk	1995
CBM D9090, 7.5 mg. Hard Disk	2250
CBM 2031, 170K Single Drive (New)	295
DC Hayes Smart Modem	300

BUSINESS SOFTWARE-8032

Word Pro 4+ or 5+	$ 309
InfoPro	219
Administrator	489
VisiCalc (expanded)	199
BPI A/R, G/L, Job Cost, Inventory, Payroll	ea.325
Commodore 64	Call

INTERFACES & ACCESSORIES

80 Column Expander	$ 159
VIC 1600 Modem	95
VIC 1650 (auto answer, auto dial)	150
VIC 1525 Graphic Printer	225
VIC 1530 Datasette Recorder	65
VIC 1541 Disk Drive	249
VIC Switch (connect 8 64's or Vics to printer, dd)	149
PET-IEEE cable	33
IEEE-IEEE cable (2m)	49
5 Slot Expander for 64	65
Parallel Interface (Epson, Okidata, IDS, NEC)	70
Programmers Reference Guide	18
Verbatim Diskettes (10 per box)	26
Hes Modem	75
ADA 1450	149
ADA 1800 (New)	129
Numeric Keypad	35

NEW COMMODORE PRODUCTS

The Executive 64	$ Call
CBM B128-80	825
CBM B256-80	1095
CBM BX700	2990
B Series Software	Call
CBM 1520 Plotter	169
CBM 1526 Printer	349

SOFTWARE FOR CBM 64 BUSINESS

WordPro 3+/64 w/Spell Right Plus	$ 79
Spell Right Plus	55
Calc Result (Advanced)	125
Calc Result (Easy)	75
Busicalc II	95
Mirage Concepts (Powerful Data Base)	95
M File (merge with WordPro)	89
Home Utilities	49
64 Mailing List (Galactic)	28
The Manager	50
Home Accountant (continental)	75
Code Writer (Writes Basic Programs)	95
Stock (investment analysis)	80
Agricultural Management	Call
General Ledger, A/R, A/P, P/R, Inv.	Call

RECREATION

Assembler Package (cassette or disk, compiled, includes editor, loader, disassembler)	$ 39
Sprite Master (access)	30
Neutral Zone (access)	35
Space Belt	19
Coco II (build your own games)	40
Vic Tree (programmers utilities)	75
Micro-Term (save to printer, disk)	39
Hesmon	35
Gothmogs Lair	30
Road Toad	15
Commodore Games	Call

MONITORS - GREAT RESOLUTION (64 OR VIC)

CBM 1701 Color Monitor	$ 249
Amdek Color Plus	299
Panasonic TR-120 (w/speaker)	155
Panasonic CT-160	279
BMC (green screen)	95
Video/Audio Cable	15

PRINTERS - LETTER QUALITY

CBM 6400, 40 cps	$1450
Diablo 620, 25 cps	949
Transtar 140 (serial)	1395
Transtar 130, 16 cps (auto load, wp features!)	769
NEC 3500 Series	1600
NEC 7700 Series	2350
Transtar 120, 14 cps	500

PRINTERS - DOT MATRIX

CBM 8023, 150 cps/graphics	$ 545
CBM 4023 Printer	395
Epson FX Printer, 160 cps	549
Epson MX-80 FT w/graftrax	Call
Epson FX-100	859
Okidata 82A, 120 cps (serial and parallel)	429
NEC 8023A (parallel)	429
Okidata 92	559
Star Gemini, 10X	329
Star Gemini, 15	499
Transtar 315 (hi-res, color)	575

POLICY

VISA/MASTERCARD
C.O.D. FOR CASH
OR BANK CHECK

Minimum $50.00 Order
Add 3% Surcharge for Credit Cards.
Exact freight calculated.
Prices subject to change without notice.
Products shipped with manufacturer's warranty.
F.O.B. Dallas, Texas

*Customer's must call for return authorization before returning any product.

PRODUCT SELECTION ADVICE
CUSTOMER SERVICE
(214) 343-1328

SJB DISTRIBUTORS INC.
10520 Plano Road, Suite 206
Dallas, Texas 75238

TO ORDER
CALL TOLL FREE
800-527-4893
800-442-1048
(Within Texas)

Business Hours
Mon. - Fri. 8:30 to 5:30
Saturday 10 to 2

Write for free catalog.

NOTE: SJB HAS A FULL LINE OF COMPUTER MEDIA IN STOCK, CALL OR WRITE FOR MORE INFORMATION.

NEWS & PRODUCTS

VIC-20 Spreadsheet

Computer Software Associates has released a VIC-20 version of PractiCalc Plus, a combination spreadsheet-data base manager program.

The program, available on tape or disk, requires a 16K RAM card expander.

PractiCalc Plus allows full use of mathematical and trigonometric function, incorporates search and sort routines, and has a single-key histogram function. The program, which sells for $49.95 on tape and $54.95 on disk, can be used for budget and business projections, expense tracking, investments, and inventory.

Micro Software International, Inc.
The Silk Mill
44 Oak Street
Newton Upper Falls, MA 02164
(617) 527-7510

Space Game For VIC

Ridge Runner is a machine-language space game for the unexpanded VIC-20.

In the game, produced by Bytes and Bits, you maneuver your multicolor ship through a minefield and a volley from enemy ships.

The game, which requires a joystick and sells for $14.95, includes a horizontally scrolling playfield, multicolor graphics, sound, a high-score recorder, and a pause option.

Bytes and Bits
524 East Canterbury Lane
Phoenix, AZ 85022
(602) 942-1475

Commodore 64 Rescue, VIC-20 Educational Games

Zeppelin Rescue, a game of coordination and skill for the Commodore 64, and several educational games for the VIC-20 are available from Micro Software International.

In *Zeppelin Rescue*, you must overcome the forces of gravity and the slow, cumbersome controls of your airship to rescue the inhabitants of a threatened city. The game includes five cityscapes and four levels—daylight, dusk, night, and dawn.

The program is available on disk for $24.95, or on cassette for $19.95.

The VIC-20 programs are all available on cassette and require no memory expansion. They include *Math Duel*, a basic number skills program for students in grades 1 through 6; *Tiny Tutor*, a simple math problem tutorial for children ages 2 through 7; *VIC Sketch*, a drawing program with SAVE and PRINT features; and *Composer*, which teaches the rudiments of music.

Math Duel and *Tiny Tutor* sell for $19.95. *VIC Sketch* and *Composer* are $14.95 each.

Micro Software International
The Silk Mill
44 Oak Street
Newton Upper Falls, MA 02164
(617) 527-7510

One-Handed Bridge

Computer Management Corporation has released *BridgePro*, a program that will allow one person and a Commodore 64 to enjoy a game of bridge.

The program, which is written in machine language, takes care of the shuffling and dealing, and will bid and play three hands.

The program allows replaying hands, prevents illegal bids, and offers a help screen on bidding for bridge newcomers. A game for two players is among *BridgePro*'s other options.

BridgePro is available on disk for $35.

Computer Management Corporation
Customer Service Center
2424 Exbourne Court
Walnut Creek, CA 94596

Help For Programmers

A collection of worksheets, programming aids, grid-sheets, and logs, designed to help simplify VIC-20 and Commodore 64 programming, is available from PM Products.

Programmer's Aids and Logs includes a guide to all keys, POKEs, CHR$ codes, and characters; sound and music worksheets; a condensed BASIC dictionary; grids for screen layout, custom characters, and sprites; and documentation worksheets for variable, subroutine, and file use.

Also included are cutout function key templates and a BASIC-Aid reference card. The package, which sells for $9.95, contains 95 color-coded pages and is punched for use in three-ring binders.

PM Products
4455 Torrance Blvd., #177
Torrance, CA 90503

Programmer's Aids and Logs is a collection of worksheets, reference cards, and programming aids for the VIC or 64.

Tax Preparation Programs

Northland Accounting has produced three tax preparation programs for the Commodore 64 and VIC-20. The programs produce a line-by-line readout of IRS Form 1040 and related schedules. Updates for new tax years will be published annually.

Taxaid I is for the unexpanded VIC-20. The program directs its output to the monitor. *Taxaid II* is for a VIC-20 with 16K. Output can be directed to the monitor or a printer. *Taxaid III* is for the Commodore 64, with output directed to the monitor or printer.

Taxaid I is available on tape for $19.95 or on disk for $24.95. The other two versions are available on tape for $24.95 or on disk for $29.95.

Northland Accounting, Inc.
Software Department
606 Second Ave.
Two Harbors, MN 55616
(218) 834-5012

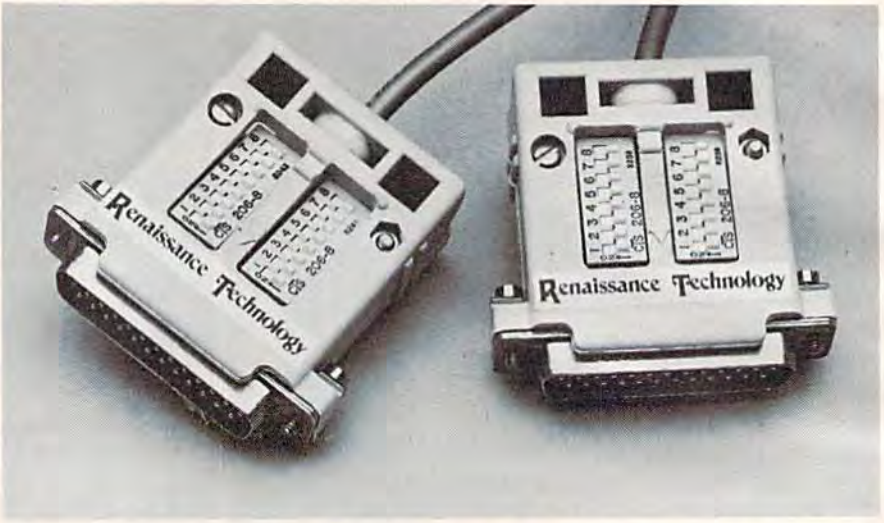
Universal Serial Cable

Renaissance Technology has produced the Universal Serial Cable, which simplifies connections between RS-232 serial computers and peripheral devices.

Built into the cable connectors are sets of DIP switches that can be set according to the requirements of the devices being connected. A cross-reference chart of switch settings is included with the cable.

The Universal Serial Cable sells for $62.

Renaissance Technology Corp.
1070 Shary Circle
Concord, CA 94518
(415) 676-5757



The Universal Serial Cable from Renaissance includes DIP switches in the connectors to eliminate the need for specially wired cables.

Activity-Planning Software

SEI Enterprises has produced a series of programs designed to help groups of users plan their activities.

The programs, which cover vacation, menu, and spending plans, allow up to ten participants to enter their preferences into the computer. Each person is then given a chance to vote on the suggestions made, and the program tallies the results and generates a printout.

The initial series is $24.95 and consists of four programs. They are *Acti-Trip*, for trip planning; *Acti-Spend*, for spending priorities; *Acti-Menu*, for meal planning; and *Acti-Play*, designed for youngsters to determine how to spend their free time.

The programs are available for the Commodore 64 and the VIC-20 with 8K expansion.

SEI Enterprises
17 Serpi Road
Highland Mills, NY 10930

Word Processor For Commodore 64

Easy Script 64, a word processing program from Commodore Software, is available for the Commodore 64.

Features of the program include selectable display colors; hunt and find; search and replace; function key editing; superscripts and subscripts; vertical and horizontal tabs; and the ability to transfer words, phrases and blocks within text. The program includes a form-letter command, and it offers optional sound effect prompts.

Easy Script 64, which sells for $49.95, also is compatible with *Easy Spell 64*, Commodore's spelling checker.

Commodore Software
1200 Wilson Drive
West Chester, PA 19380
(215) 431-9100

VIC Memory Poster

Kevco has produced *Inside the VIC-20*, a 27 by 21-inch color poster that shows the important memory locations in the VIC.

The chart includes information on the BASIC memory map, video screens, sound and color, as well as data on paddles, joysticks and light pens.

The poster, which makes often-used information avail-

able at a glance, sells for $6.95. A Commodore 64 version is forthcoming.

Kevco Electronic and Software Engineering
480 Georgia Court
Claremont, CA 91711
(714) 626-4148

Investment Manager

Portfolio Manager is an investment management program for the Commodore 64 or 16K VIC-20 computers.

The program, which is the first in a series of personal finance programs planned by Basic Byte, allows the user to easily record and compute stock transactions. *Portfolio Manager* sells for $29.95.

Basic Byte, Inc.
13108 Ludlow
Huntington Woods, MI 48070
(313) 545-6779

Property Management System

MicroSpec has introduced its *Rental Property Manager* program for the Commodore 64.

With the program, a landlord can keep track of up to 200 rental units per diskette. The program maintains 18 fields per record, including information on the property owner, the tenant, the rent payment record, and the availability of the unit.

The system, which requires one disk drive, can produce a variety of reports including tenant lists, overdue rent lists, expired lease lists, vacancy lists, and income and expense reports.

Rental Property Manager sells for $179.95.

MicroSpec, Inc.
Box 836085
Plano, TX 75086
(214) 867-1333

Educational Games For Commodore 64

Bertamax has reached agreement with Commodore Business Machines to convert 21 educational programs for use on the Commodore 64.

Among the programs being converted are several programs designed for children in kindergarten through third grade. The titles include *Number Match It, Addition Match, Subtraction Match, Multiplication Match, Division Match*, and six reading programs in the *Story Mix* series.

Other programs covered in the agreement are: *Spelling in Context*, a 308-lesson program available for grade levels one through eight; *Math Facts Games—Set 2*, a series of four two-player math games; and *Number Cruncher*, 30 lessons in math and problem-solving skills.

Bertamax, Inc.
3647 Stoneway North
Seattle, WA 98103
(206) 547-4056

Spreadsheet For VIC And 64

ESP›Calc is an electronic spreadsheet planning calculator for both the VIC and 64.

The same program runs on both computers, and the size of the spreadsheet is limited only by computer memory. The manual includes step-by-step instructions to help novice spreadsheet users become accustomed to the program.

ESP›Calc is designed to handle things like household budgets, heat and electric use, stock portfolios, and rental property analysis. The program also includes printer options.

The cassette version of *ESP›Calc* sells for $43.50; the disk version is $47.50.

New Leaf Inc.
120 Lynnhaven
Belleville, IL 62223

COMPUTE!'s GAZETTE *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.

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, ⟨⟩, 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:
{CLEAR}	SHIFT + CLR/HOME	
{HOME}	CLR/HOME	
{UP}	SHIFT + CRSR ↑	
{DOWN}	CRSR ↓	
{LEFT}	SHIFT + CRSR ←	
{RIGHT}	CRSR →	
{RVS}	CTRL + 9	
{OFF}	CTRL + 0	
{BLK}	CTRL + 1	
{WHT}	CTRL + 2	
{RED}	CTRL + 3	
{CYN}	CTRL + 4	
{PUR}	CTRL + 5	
{GRN}	CTRL + 6	
{BLU}	CTRL + 7	
{YEL}	CTRL + 8	
⟨1⟩	f1	
⟨2⟩	f2	
⟨3⟩	f3	
⟨4⟩	f4	
⟨5⟩	f5	
⟨6⟩	f6	
⟨7⟩	f7	
⟨8⟩	f8	
{F1}	f1	
{F2}	f2	
{F3}	f3	
{F4}	f4	
{F5}	f5	
{F6}	f6	
{F7}	f7	
{F8}	f8	

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.*

The Automatic Proofreader

"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. The same program works on both the VIC-20 and Commodore 64. Be very careful when entering the DATA statements — don't type an l instead of a 1, an O instead of a 0, extra commas, etc.

2. SAVE the Proofreader on tape or disk at least twice *before running it for the first time*. This is very important because the Proofreader erases this part of itself when you first type RUN.

3. After the Proofreader is SAVEd, type RUN. It will check itself for typing errors in the DATA statements and warn you if there's a mistake. Correct any errors and SAVE the corrected version. Keep a copy in a safe place — you'll need it again and again, every time you enter a program from COMPUTE!'s Gazette.

4. When a correct version of the Proofreader is RUN, it activates itself. You are now ready to enter a program listing. If you press RUN/STOP-RESTORE, the Proofreader is disabled. To reactivate it, just type 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 in 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.

Special Tape SAVE Instructions

When you're done typing a listing, you must disable the Proofreader before SAVEing the program on tape. Disable the Proofreader by pressing RUN/STOP-RESTORE (hold down the RUN/STOP key and sharply hit the RESTORE key). This procedure is not necessary for disk SAVEs, *but you must disable the Proofreader this way before a tape SAVE.*

SAVE to tape erases the Proofreader from memory, so you'll have to LOAD and RUN it again if you want to type another listing. SAVE to disk does not erase the Proofreader.

Replace Original Proofreader

If you typed in the original version of the Proofreader (October 1983 issue), you should replace it with the improved version below. We added a POKE to the original version to protect it from being erased when you LOAD another program from tape. The POKE does protect the Proofreader, and the Proofreader itself was not affected. However, a quirk in the VIC-20's operating system means that programs typed in with the Proofreader and SAVEd on 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 Commodore 64).

If you have a program typed in with the original Proofreader and SAVEd on tape, follow 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 fine in the future. We strongly recommend that you type in the new version of the Proofreader and discard the old one.

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
```

MLX Machine Language Entry Program

For Commodore 64 And VIC-20

Charles Brannon, Program Editor

MLX is a labor-saving utility that allows almost failsafe entry of machine language programs published in COMPUTE!'s GAZETTE. You need to know nothing about machine language to use MLX—it was designed for everyone. There are separate versions for the Commodore 64 and expanded VIC-20 (at least 8K). MLX was conceived and written by Program Editor Charles Brannon. Important: MLX is required to type in the machine language programs in this issue.

MLX is a new way to enter long machine language (ML) programs with a minimum of fuss. MLX lets you enter the numbers from a special list that looks similar to BASIC DATA statements. It checks your typing on a line-by-line basis. It won't let you enter illegal characters when you should be typing numbers. It won't let you enter numbers greater than 255 (forbidden in ML). It won't let you enter the wrong numbers on the wrong line. In addition, MLX creates a ready-to-use tape or disk file. You can then use the LOAD command to read the program into the computer, as with any program:

```
LOAD "filename",1,1   (for tape)
LOAD "filename",8,1   (for disk)
```

To start the program, you enter a SYS command that transfers control from BASIC to machine language. The starting SYS number always appears in the appropriate article.

Using MLX

Type in and save the correct version of MLX for your computer (you'll want to use it in the future). When you're ready to type in an ML program, run MLX. MLX asks you for two numbers: the starting address and the ending address. These numbers are given in the article accompanying the ML program.

You'll see a prompt corresponding to the starting address. The prompt is the current line you are entering from the listing. It increases by six each time you enter a line. That's because each line has seven numbers—six actual data numbers plus a *checksum number*. The checksum verifies that you typed the previous six numbers correctly. If you enter any of the six numbers wrong, or enter the checksum wrong, the computer rings a buzzer and prompts you to reenter the line. If you enter it correctly, a bell tone sounds and you continue to the next line.

MLX accepts only numbers as input. It you make a typing error, press the INST/DEL key; the entire number is deleted. You can press it as many times as necessary back to the start of the line. If you enter three-digit numbers as listed, the computer automatically prints the comma and goes on to accept the next number. If you enter less than three digits, you can press either the comma, SPACE bar, or RETURN key to advance to the next number. The checksum automatically appears in inverse video for emphasis.

MLX Commands

When you finish typing an ML listing (assuming you type it all in one session) you can then save the completed program on tape or disk. Follow the screen instructions. If you get any errors while saving, you probably have a bad disk, or the disk is full, or you've made a typo when entering the MLX program itself.

You don't have to enter the whole ML program in one sitting. MLX lets you enter as much as you want, save it, and then reload the file from tape or disk later. MLX recognizes these commands:

SHIFT-S: Save	**SHIFT-N: New Address**
SHIFT-L: Load	**SHIFT-D: Display**

When you enter a command, MLX jumps out of the line you've been typing, so we recommend you do it at a new prompt. Use the Save command to save what you've been working on. It will save on tape or disk as if you've finished, but the tape or disk won't work, of course, until you finish the typing. Remember what address you stop at. The next time you run MLX, answer all the prompts as you did before, then insert the disk or tape. When you get to the entry prompt, press SHIFT-L to reload the partly completed file into memory. Then use the New Address command to resume typing.

To use the New Address command, press SHIFT-N and enter the address where you previously stopped. The prompt will change, and you can then continue typing. Always enter a New Address that matches up with one of the line numbers in the special listing, or else the checksum won't work. The Display command lets you display a section of your typing. After you press SHIFT-D, enter two addresses within the line number range of the listing. You can abort the listing by pressing any key.

The special MLX commands may seem a bit confusing, but as you work with MLX, they will become valuable. For example, what if you forgot where you stopped typing? Use the Display command to scan memory from the beginning to the end of the program. When you reach the end of your typing, the lines will contain a random pattern of numbers. When you see the end of your typing, press any key to stop the listing. Use the New Address command to continue typing from the proper location.

You can use the Save and Load commands to make copies of the completed program. Use Load to reload the tape or disk, then insert a new tape or disk and use Save to make a new copy.

Be sure to save MLX; it will be used for future ML programs in COMPUTE!'s GAZETTE.

See program listings on page 184.

SpeedScript

(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:

SpeedScript—Commodore 64 Version

```
2049 :011,008,010,000,158,050,238
2055 :048,054,049,000,000,000,158
2061 :032,103,009,076,193,009,179
2067 :165,251,141,051,008,165,032
2073 :252,141,052,008,165,253,128
2079 :141,054,008,165,254,141,026
2085 :055,008,166,181,240,032,207
2091 :169,000,141,186,026,160,213
2097 :000,185,000,000,153,000,131
2103 :000,200,204,186,026,208,111
2109 :244,238,052,008,238,055,128
2115 :008,224,000,240,007,202,236
2121 :208,224,165,180,208,222,000
2127 :096,165,181,170,005,180,108
2133 :208,001,096,024,138,101,141
2139 :252,141,123,008,165,251,007
2145 :141,122,008,024,138,101,119
2151 :254,141,126,008,165,253,026
2157 :141,125,008,232,164,180,191
2163 :208,004,240,013,160,255,227
2169 :185,000,000,153,000,000,203
2175 :136,192,255,208,245,206,089
2181 :123,008,206,126,008,202,038
2187 :208,234,096,169,040,133,251
2193 :195,133,020,169,004,133,031
2199 :196,169,216,133,021,173,035
2205 :182,026,133,155,173,183,241
2211 :026,133,156,162,001,173,046
2217 :185,026,133,012,173,195,125
2223 :026,141,032,208,160,000,230
2229 :173,194,026,145,020,177,148
2235 :155,153,196,026,200,041,190
2241 :127,201,031,240,019,192,235
2247 :040,208,235,136,177,155,126
2253 :041,127,201,032,240,005,083
2259 :136,208,245,160,039,200,175
2265 :132,167,136,185,196,026,035
2271 :145,195,136,016,248,164,103
2277 :167,024,152,101,155,133,193
2283 :155,165,156,105,000,133,181
2289 :156,152,157,060,003,192,193
2295 :040,240,008,169,032,145,113
2301 :195,200,076,246,008,024,234
2307 :165,195,105,040,133,195,068
2313 :133,020,144,004,230,196,224
2319 :230,021,232,224,025,240,219
2325 :003,076,179,008,165,155,095
2331 :141,192,026,165,156,141,080
2337 :193,026,096,169,000,133,138
2343 :155,141,182,026,141,188,104
2349 :026,133,038,169,028,133,060
2355 :156,141,183,026,141,189,119
2361 :026,133,039,169,032,162,106
2367 :179,160,255,198,156,145,132
2373 :155,200,230,156,145,155,086
2379 :200,208,251,230,156,202,042
2385 :208,246,145,155,096,133,040
2391 :167,132,168,160,000,177,123
2397 :167,240,006,032,210,255,235
2403 :200,208,246,096,169,012,006
2409 :141,195,026,169,038,133,039
2415 :001,169,011,141,194,026,141
2421 :032,036,009,169,000,141,248
2427 :185,026,032,115,015,169,153
2433 :255,141,138,002,032,245,174
2439 :012,032,150,009,169,100,095
2445 :160,025,032,086,009,238,179
2451 :184,026,096,032,166,009,148
2457 :169,085,160,025,032,086,198
2463 :009,169,000,141,184,026,176
2469 :096,162,039,169,032,157,052
2475 :000,004,202,016,250,169,044
2481 :019,076,210,255,072,041,082
2487 :128,074,133,167,104,041,062
2493 :063,005,167,096,160,000,168
2499 :177,038,133,002,160,000,193
2505 :177,038,073,128,145,038,032
2511 :032,142,008,032,228,255,136
2517 :208,013,165,162,041,016,050
2523 :240,245,169,000,133,162,144
2529 :076,199,009,170,160,000,071
2535 :165,002,145,038,224,095,132
2541 :208,012,032,160,011,169,061
2547 :032,160,000,145,038,076,182
2553 :193,009,173,184,026,240,050
2559 :007,138,072,032,150,009,151
2565 :104,170,138,201,013,208,071
2571 :002,162,095,138,041,127,064
2577 :201,032,144,070,224,160,080
2583 :208,002,162,032,138,072,125
2589 :173,185,026,240,003,032,176
2595 :140,014,104,032,181,009,003
2601 :160,000,145,038,032,142,046
2607 :008,056,165,038,237,188,227
2613 :026,133,167,165,039,237,052
2619 :189,026,005,167,144,014,092
2625 :165,038,105,000,141,188,190
2631 :026,165,039,105,000,141,035
2637 :189,026,230,038,208,002,002
2643 :230,039,032,231,010,076,189
2649 :193,009,138,174,125,010,226
2655 :221,125,010,240,006,202,131
2661 :208,248,076,193,009,202,013
2667 :138,010,170,169,009,072,163
2673 :169,192,072,189,162,010,139
2679 :072,189,161,010,072,096,207
2685 :035,029,157,137,133,002,106
2691 :012,138,134,020,148,004,075
2697 :019,009,147,135,139,005,079
2703 :136,140,022,145,017,159,250
2709 :018,024,026,016,028,030,035
2715 :006,001,011,008,031,003,215
2721 :150,011,159,011,170,011,161
2727 :227,011,054,012,066,012,037
2733 :080,012,179,012,231,013,188
2739 :139,014,014,014,083,014,201
2745 :201,014,225,014,253,014,138
2751 :024,015,185,015,222,017,157
2757 :205,016,043,018,080,012,059
2763 :179,012,111,018,118,019,148
2769 :023,020,028,012,108,020,164
2775 :186,017,107,023,002,014,052
2781 :039,020,244,012,210,023,001
```

```
2787 :052,025,122,014,032,071,031
2793 :011,056.165,038,237,182,154
2799 :026,133,167,165,039,237,238
2805 :183,026,005,167,176,030,064
2811 :056,173,182,026,233,000,153
2817 :133,167,173,183,026,233,148
2823 :028,005,167,240,013,165,113
2829 :038,141,182,026,165,039,092
2835 :141,183,026,032,142,008,039
2841 :056,173,192,026,229,038,227
2847 :133,155,173,193,026,229,172
2853 :039,133,156,005,155,240,253
2859 :002,176,024,024,173,182,112
2865 :026,109,061,003,141,182,059
2871 :026,173,183,026,105,000,056
2877 :141,183,026,032,142,008,081
2883 :076,025,011,096,056,173,248
2889 :188,026,233,000,133,167,052
2895 :173,189,026,233,207,005,144
2901 :167,144,010,169,000,141,204
2907 :188,026,169,207,141,189,243
2913 :026,056,165,038,233,000,103
2919 :133,167,165,039,233,028,100
2925 :005,167,176,009,169,000,123
2931 :133,038,169,028,133,039,143
2937 :096,056,165,038,237,188,133
2943 :026,133,167,165,039,237,126
2949 :189,026,005,167,176,001,185
2955 :096,173,188,026,133,038,025
2961 :173,189,026,133,039,096,033
2967 :230,038,208,002,230,039,130
2973 :076,231,010,165,038,208,117
2979 :002,198,039,198,038,076,202
2985 :231,010,165,038,133,155,133
2991 :165,039,133,156,198,156,254
2997 :160,255,177,155,201,032,137
3003 :240,004,201,031,208,003,106
3009 :136,208,243,177,155,201,033
3015 :032,240,008,201,031,240,183
3021 :004,136,208,243,096,132,000
3027 :167,056,165,155,101,167,254
3033 :133,038,165,156,105,000,046
3039 :133,039,076,231,010,160,104
3045 :000,177,038,201,032,240,149
3051 :008,201,031,240,004,200,151
3057 :208,243,096,200,240,025,229
3063 :177,038,201,032,240,247,158
3069 :201,031,240,243,024,152,120
3075 :101,038,133,038,165,039,005
3081 :105,000,133,039,076,231,081
3087 :010,173,188,026,133,038,071
3093 :173,189,026,133,039,076,145
3099 :013,012,169,000,141,182,032
3105 :026,173,189,026,056,233,224
3111 :004,201,028,176,002,169,107
3117 :028,141,183,026,032,142,085
3123 :008,076,016,012,238,195,084
3129 :026,173,195,026,041,015,021
3135 :141,195,026,096,238,194,185
3141 :026,173,194,026,041,015,032
3147 :141,194,026,076,142,008,150
3153 :165,038,133,155,165,039,008
3159 :133,156,198,156,160,255,121
3165 :177,155,201,046,240,012,156
3171 :201,033,240,008,201,063,077
3177 :240,004,201,031,208,004,025
3183 :136,208,235,096,177,155,094
3189 :201,046,240,026,201,033,096
3195 :240,022,201,063,240,018,139
3201 :201,031,240,014,136,208,191
3207 :235,198,156,165,156,201,222
3213 :000,176,227,076,169,012,033
3219 :132,167,198,167,200,240,227
3225 :010,177,155,201,032,240,200
3231 :247,136,076,210,011,164,235
3237 :167,076,115,012,169,000,192
3243 :133,038,169,028,133,039,199
3249 :076,231,010,160,000,177,063
3255 :038,201,046,240,029,201,170
3261 :033,240,025,201,063,240,223
3267 :021,201,031,240,017,200,137
3273 :208,235,230,039,165,039,093
3279 :205,189,026,240,226,144,213
3285 :224,076,016,012,200,240,213
3291 :250,177,038,201,032,240,133
3297 :247,201,046,240,243,201,123
3303 :033,240,239,201,063,240,223
3309 :235,201,031,240,231,076,227
3315 :001,012,169,000,141,130,184
3321 :027,169,208,141,131,027,184
3327 :032,166,009,169,120,160,143
3333 :025,032,086,009,169,001,071
3339 :141,184,026,096,056,165,167
3345 :038,233,000,133,167,165,241
3351 :039,233,028,005,167,208,191
3357 :003,104,104,096,165,038,027
3363 :133,251,165,039,133,252,240
3369 :096,056,165,038,133,253,014
3375 :073,255,101,251,141,134,234
3381 :027,165,039,133,254,073,232
3387 :255,101,252,141,135,027,202
3393 :165,251,141,136,027,165,182
3399 :252,141,137,027,165,253,022
3405 :141,138,027,133,251,165,164
3411 :254,141,139,027,133,252,005
3417 :024,173,135,027,109,131,176
3423 :027,201,255,144,020,032,006
3429 :166,009,169,135,160,025,253
3435 :032,086,009,169,001,141,033
3441 :184,026,169,000,133,198,055
3447 :096,173,130,027,133,253,163
3453 :173,131,027,133,254,173,248
3459 :134,027,133,180,024,109,226
3465 :130,027,141,130,027,173,253
3471 :135,027,133,181,109,131,091
3477 :027,141,131,027,169,000,132
3483 :141,026,208,169,032,133,096
3489 :001,032,019,008,169,038,172
3495 :133,001,169,001,141,026,126
3501 :208,173,136,027,133,251,077
3507 :173,137,027,133,252,173,050
3513 :138,027,133,253,173,139,024
3519 :027,133,254,056,173,188,254
3525 :026,229,253,133,180,173,167
3531 :189,026,229,254,133,181,191
3537 :032,019,008,056,173,188,173
3543 :026,237,134,027,141,188,200
3549 :026,173,189,026,237,135,239
3555 :027,141,189,026,096,032,226
3561 :015,013,032,160,011,032,240
3567 :042,013,056,173,130,027,168
3573 :233,001,141,130,027,173,182
3579 :131,027,233,000,141,131,146
3585 :027,096,032,151,011,032,094
```

```
3591 :015,013,032,160,011,076,058
3597 :042,013,032,245,012,169,014
3603 :002,133,012,032,166,009,117
3609 :169,147,160,025,032,086,132
3615 :009,032,228,255,240,251,022
3621 :072,032,150,009,104,041,189
3627 :191,201,023,208,009,032,195
3633 :015,013,032,171,011,076,111
3639 :042,013,201,019,208,009,035
3645 :032,015,013,032,081,012,246
3651 :076,042,013,201,016,208,111
3657 :009,032,015,013,032,025,199
3663 :015,076,042,013,096,056,121
3669 :165,038,237,182,026,133,098
3675 :167,165,039,237,183,026,140
3681 :005,167,240,011,173,182,107
3687 :026,133,038,173,183,026,170
3693 :133,039,096,169,000,133,167
3699 :038,169,028,133,039,076,086
3705 :231,010,160,005,140,156,055
3711 :027,032,140,014,172,156,156
3717 :027,136,208,244,076,228,028
3723 :011,024,165,038,133,251,249
3729 :105,001,133,253,165,039,073
3735 :133,252,105,000,133,254,004
3741 :056,173,188,026,229,253,058
3747 :133,180,173,189,026,229,069
3753 :254,133,181,201,255,208,121
3759 :006,169,001,133,180,230,126
3765 :181,032,080,008,160,000,130
3771 :169,032,145,038,238,188,229
3777 :026,208,003,238,189,026,115
3783 :076,013,012,173,185,026,172
3789 :073,014,141,185,026,096,228
3795 :169,162,160,025,032,086,077
3801 :009,032,228,255,240,251,208
3807 :201,089,096,169,002,133,145
3813 :012,032,166,009,169,185,034
3819 :160,025,032,086,009,032,067
3825 :211,014,240,003,076,150,167
3831 :009,162,255,154,076,013,148
3837 :008,160,000,177,038,201,069
3843 :031,240,015,200,208,247,176
3849 :230,039,165,039,205,189,108
3855 :026,144,238,076,016,012,015
3861 :200,076,001,012,165,038,001
3867 :133,155,165,039,133,156,040
3873 :198,156,160,255,177,155,110
3879 :201,031,240,016,136,192,087
3885 :255,208,245,198,156,165,248
3891 :156,201,028,176,237,076,157
3897 :169,012,056,152,101,155,190
3903 :133,155,169,000,101,156,009
3909 :133,156,056,165,155,229,195
3915 :038,133,167,165,156,229,195
3921 :039,005,167,208,018,132,138
3927 :167,024,165,155,229,167,226
3933 :133,155,165,156,233,000,167
3939 :133,156,076,043,015,165,175
3945 :155,133,038,165,156,133,117
3951 :039,076,231,010,120,169,244
3957 :127,141,013,220,169,027,046
3963 :141,017,208,169,146,141,177
3969 :020,003,169,015,141,021,242
3975 :003,169,001,141,026,208,171
3981 :141,018,208,088,096,169,093
3987 :058,164,012,205,018,208,044
3993 :208,005,169,001,172,195,135
3999 :026,140,033,208,141,018,213
4005 :208,201,001,240,008,169,224
4011 :001,141,025,208,076,188,042
4017 :254,169,001,141,025,208,207
4023 :076,049,234,173,141,002,090
4029 :041,001,208,003,032,245,207
4035 :012,032,166,009,169,200,015
4041 :160,025,032,086,009,160,161
4047 :000,177,038,073,128,145,000
4053 :038,032,142,008,160,000,081
4059 :177,038,073,128,145,038,050
4065 :169,002,133,012,032,228,033
4071 :255,240,251,009,064,201,227
4077 :087,208,009,032,022,016,099
4083 :032,228,011,076,037,016,131
4089 :201,083,208,009,032,022,036
4095 :016,032,180,012,076,037,096
4101 :016,201,080,208,009,032,039
4107 :022,016,032,254,014,076,169
4113 :037,016,076,150,009,165,214
4119 :038,133,253,141,045,027,148
4125 :165,039,133,254,141,046,039
4131 :027,096,056,165,038,133,038
4137 :251,237,045,027,141,134,108
4143 :027,165,039,133,252,237,132
4149 :046,027,141,135,027,032,205
4155 :065,013,173,045,027,133,003
4161 :038,173,046,027,133,039,009
4167 :032,142,008,076,206,015,038
4173 :169,038,229,211,141,190,031
4179 :026,169,000,141,158,027,092
4185 :160,000,169,156,032,210,048
4191 :255,169,018,032,210,255,010
4197 :169,032,032,210,255,169,200
4203 :157,032,210,255,140,191,068
4209 :026,032,228,255,240,251,121
4215 :172,191,026,133,167,169,209
4221 :146,032,210,255,169,032,201
4227 :032,210,255,169,157,032,218
4233 :210,255,169,155,032,210,144
4239 :255,165,167,201,013,240,160
4245 :046,201,020,208,015,136,007
4251 :016,004,200,076,091,016,046
4257 :169,157,032,210,255,076,036
4263 :091,016,041,127,201,032,163
4269 :144,172,204,190,026,240,125
4275 :167,165,167,153,236,026,069
4281 :032,210,255,169,000,133,216
4287 :212,200,076,091,016,032,050
4293 :210,255,169,000,153,236,196
4299 :026,152,096,032,166,009,172
4305 :169,237,160,025,032,086,150
4311 :009,032,051,017,176,031,019
4317 :169,000,133,155,169,028,107
4323 :133,156,174,188,026,172,052
4329 :189,026,169,155,032,216,252
4335 :255,176,010,032,183,255,126
4341 :041,191,208,003,076,028,024
4347 :018,240,039,173,050,017,020
4353 :201,008,144,006,032,169,049
4359 :023,076,028,017,173,050,118
4365 :017,201,001,240,249,032,241
4371 :166,009,169,243,160,025,023
4377 :032,086,009,032,115,015,058
4383 :169,001,141,184,026,096,136
4389 :032,166,009,169,254,160,059
4395 :025,032,086,009,076,028,043
4401 :017,000,032,077,016,240,175
```

```
4407 :024,169,029,160,026,032,239
4413 :086,009,032,228,255,240,143
4419 :251,162,008,201,068,240,229
4425 :012,162,001,201,084,240,005
4431 :006,032,150,009,104,104,228
4437 :096,142,050,017,169,001,048
4443 :160,000,032,186,255,160,116
4449 :000,224,001,240,042,185,021
4455 :236,026,201,064,208,007,077
4461 :185,237,026,201,058,240,032
4467 :028,169,048,141,020,027,036
4473 :169,058,141,021,027,185,210
4479 :236,026,153,022,027,200,023
4485 :204,191,026,144,244,240,158
4491 :242,200,076,156,017,185,247
4497 :236,026,153,020,027,200,039
4503 :204,191,026,208,244,140,140
4509 :044,027,032,166,009,169,092
4515 :236,160,026,032,086,009,200
4521 :173,044,027,162,020,160,243
4527 :027,032,189,255,169,013,092
4533 :032,210,255,076,086,018,090
4539 :032,166,009,169,232,160,187
4545 :025,032,086,009,032,228,093
4551 :255,240,251,032,181,009,143
4557 :009,128,072,173,185,026,030
4563 :240,003,032,140,014,032,160
4569 :150,009,104,076,041,010,095
4575 :056,165,038,233,000,133,080
4581 :167,165,039,233,028,005,098
4587 :167,240,004,169,005,133,185
4593 :012,032,166,009,169,049,166
4599 :160,026,032,086,009,032,080
4605 :051,017,165,012,201,005,192
4611 :240,003,032,036,009,169,236
4617 :000,166,038,164,039,032,192
4623 :213,255,144,003,076,252,190
4629 :016,142,188,026,140,189,210
4635 :026,032,231,255,032,166,001
4641 :009,169,019,160,026,032,192
4647 :086,009,076,028,017,032,031
4653 :166,009,169,055,160,026,118
4659 :032,086,009,032,051,017,022
4665 :169,001,162,000,160,028,065
4671 :032,213,255,032,183,255,009
4677 :041,191,240,211,032,166,182
4683 :009,169,006,160,026,032,221
4689 :086,009,076,028,017,120,161
4695 :169,000,141,026,208,141,004
4701 :033,208,169,049,141,020,201
4707 :003,169,234,141,021,003,158
4713 :169,255,141,013,220,088,223
4719 :096,169,147,032,210,255,252
4725 :169,013,032,210,255,032,060
4731 :086,018,032,162,018,169,096
4737 :013,032,210,255,169,063,103
4743 :160,026,032,086,009,032,224
4749 :228,255,201,013,208,249,015
4755 :032,115,015,076,150,009,032
4761 :032,204,255,169,001,032,078
4767 :195,255,096,032,231,255,199
4773 :169,001,162,008,160,000,153
4779 :032,186,255,169,002,162,209
4785 :078,160,026,032,189,255,149
4791 :032,192,255,176,221,162,197
4797 :001,032,198,255,032,207,146
4803 :255,032,207,255,032,207,159
4809 :255,032,183,255,208,202,056
4815 :032,207,255,240,197,032,146
4821 :204,255,032,228,255,201,108
4827 :032,208,005,032,228,255,211
4833 :240,251,162,001,032,198,085
4839 :255,032,207,255,072,032,060
4845 :207,255,168,104,170,152,013
4851 :160,055,132,001,032,205,060
4857 :189,160,054,132,001,169,186
4863 :032,032,210,255,032,207,255
4869 :255,240,006,032,210,255,235
4875 :076,003,019,169,013,032,067
4881 :210,255,076,199,018,162,169
4887 :000,142,047,027,142,048,173
4893 :027,142,049,027,056,177,251
4899 :155,233,048,144,042,201,090
4905 :010,176,038,014,047,027,097
4911 :046,048,027,014,047,027,000
4917 :046,048,027,014,047,027,006
4923 :046,048,027,014,047,027,012
4929 :046,048,027,013,047,027,017
4935 :141,047,027,200,208,212,138
4941 :230,156,076,033,019,248,071
4947 :173,047,027,013,048,027,162
4953 :240,023,056,173,047,027,143
4959 :233,001,141,047,027,173,205
4965 :048,027,233,000,141,048,086
4971 :027,238,049,027,076,083,095
4977 :019,173,049,027,216,096,181
4983 :056,173,130,027,233,000,226
4989 :141,132,027,173,131,027,244
4995 :233,208,141,133,027,013,118
5001 :132,027,208,016,032,166,206
5007 :009,169,088,160,026,032,115
5013 :086,009,169,001,141,184,227
5019 :026,096,024,165,038,133,125
5025 :251,109,132,027,133,253,042
5031 :165,039,133,252,109,133,230
5037 :027,133,254,056,173,188,236
5043 :026,229,251,133,180,173,147
5049 :189,026,229,252,133,181,171
5055 :024,101,254,201,207,144,098
5061 :016,032,166,009,169,080,157
5067 :160,026,032,086,009,169,173
5073 :001,141,184,026,096,032,177
5079 :080,008,024,173,132,027,147
5085 :133,180,109,188,026,141,230
5091 :188,026,173,133,027,133,139
5097 :181,109,189,026,141,189,044
5103 :026,165,038,133,253,165,251
5109 :039,133,254,169,000,133,205
5115 :251,169,208,133,252,169,153
5121 :000,141,026,208,169,032,065
5127 :133,001,032,019,008,169,113
5133 :038,133,001,169,001,141,240
5139 :026,208,076,231,010,160,218
5145 :000,177,038,170,200,177,019
5151 :038,136,145,038,200,138,214
5157 :145,038,096,160,000,177,141
5163 :038,041,063,240,010,201,124
5169 :027,176,006,177,038,073,034
5175 :064,145,038,076,151,011,028
5181 :133,167,041,064,010,005,225
5187 :167,041,191,133,167,041,039
5193 :032,073,032,010,005,167,136
5199 :096,005,075,066,005,058,128
5205 :002,001,027,014,015,018,162
5211 :000,000,000,000,000,032,123
5217 :166,009,169,132,160,026,247
```

```
5223 :076,086,009,076,129,021,244
5229 :169,004,141,159,027,160,001
5235 :007,173,141,002,041,001,224
5241 :240,054,032,166,009,169,023
5247 :107,160,026,032,086,009,035
5253 :032,228,255,240,251,056,171
5259 :233,048,201,003,144,217,217
5265 :201,008,176,213,141,159,019
5271 :027,032,166,009,169,116,158
5277 :160,026,032,086,009,032,246
5283 :228,255,240,251,056,233,146
5289 :048,048,190,201,010,176,074
5295 :186,168,169,001,174,159,008
5301 :027,032,186,255,169,000,082
5307 :032,189,255,032,096,020,043
5313 :032,192,255,162,001,032,099
5319 :201,255,144,003,076,129,239
5325 :021,169,000,133,155,169,084
5331 :028,133,156,162,000,142,064
5337 :141,027,142,140,027,142,068
5343 :160,027,142,161,027,142,114
5349 :154,027,189,080,020,157,088
5355 :146,027,232,224,007,208,055
5361 :245,169,255,141,155,027,209
5367 :160,000,177,155,016,003,246
5373 :076,073,022,201,031,240,128
5379 :034,153,050,027,200,238,193
5385 :153,027,173,153,027,205,235
5391 :147,027,144,230,136,140,071
5397 :187,026,177,155,201,032,031
5403 :240,009,206,153,027,136,030
5409 :208,244,172,187,026,140,242
5415 :187,026,152,056,101,155,204
5421 :133,155,165,156,105,000,247
5427 :133,156,160,000,173,155,060
5433 :027,201,255,208,003,032,015
5439 :241,021,032,038,022,173,078
5445 :187,026,141,186,026,169,036
5451 :050,133,169,169,027,133,244
5457 :170,032,104,024,032,053,240
5463 :022,173,155,027,205,150,051
5469 :027,144,003,032,153,021,217
5475 :056,165,155,237,188,026,158
5481 :133,167,165,156,237,189,128
5487 :026,005,167,240,034,144,215
5493 :032,169,000,141,140,027,114
5499 :141,149,027,032,153,021,134
5505 :032,225,255,240,251,169,021
5511 :013,032,210,255,032,204,113
5517 :255,169,001,032,195,255,024
5523 :076,150,009,076,247,020,213
5529 :056,173,148,027,237,155,181
5535 :027,168,136,136,240,010,108
5541 :048,008,169,013,032,210,133
5547 :255,136,208,248,173,141,052
5553 :027,240,019,141,186,026,048
5559 :173,144,027,133,169,173,234
5565 :145,027,133,170,032,038,222
5571 :022,032,104,024,169,013,047
5577 :032,210,255,032,210,255,171
5583 :032,210,255,173,152,027,032
5589 :208,026,032,204,255,032,202
5595 :166,009,169,141,160,026,122
5601 :032,086,009,032,228,255,099
5607 :240,251,032,096,020,162,008
5613 :001,032,201,255,238,154,094
5619 :027,173,140,027,240,019,101
5625 :141,186,026,173,142,027,176
5631 :133,169,173,143,027,133,009
5637 :170,032,038,022,032,104,147
5643 :024,169,013,032,210,255,202
5649 :172,149,027,140,155,027,175
5655 :136,136,240,010,048,008,089
5661 :169,013,032,210,255,136,076
5667 :208,248,096,169,032,172,192
5673 :146,027,140,153,027,032,054
5679 :210,255,136,208,250,096,178
5685 :172,151,027,024,152,109,176
5691 :155,027,141,155,027,169,221
5697 :013,032,210,255,136,208,151
5703 :250,096,141,157,027,041,015
5709 :127,032,061,020,201,049,055
5715 :144,007,201,058,176,003,160
5721 :076,075,023,174,162,022,109
5727 :221,162,022,240,012,202,186
5733 :208,248,206,153,027,173,092
5739 :157,027,076,000,021,202,078
5745 :138,010,170,140,156,027,242
5751 :169,022,072,169,134,072,245
5757 :189,174,022,072,189,173,176
5763 :022,072,096,056,173,156,194
5769 :027,101,155,133,155,165,105
5775 :156,105,000,133,156,076,001
5781 :247,020,177,155,201,031,212
5787 :240,001,136,140,156,027,087
5793 :096,010,087,065,076,082,065
5799 :084,066,083,078,072,070,108
5805 :192,022,201,022,210,022,074
5811 :220,022,230,022,240,022,167
5817 :250,022,004,023,007,023,002
5823 :045,023,169,000,141,152,209
5829 :027,200,076,151,022,200,105
5835 :169,001,141,160,027,076,009
5841 :151,022,200,032,022,019,143
5847 :141,146,027,076,151,022,010
5853 :200,032,022,019,141,147,014
5859 :027,076,151,022,200,032,223
5865 :022,019,141,149,027,076,155
5871 :151,022,200,032,022,019,173
5877 :141,150,027,076,151,022,044
5883 :200,032,022,019,141,151,048
5889 :027,076,151,022,076,153,250
5895 :021,056,152,101,155,141,121
5901 :142,027,165,156,105,000,096
5907 :141,143,027,032,037,023,166
5913 :056,152,237,156,027,141,026
5919 :140,027,200,076,151,022,135
5925 :200,177,155,201,031,208,241
5931 :249,136,096,056,152,101,065
5937 :155,141,144,027,165,156,069
5943 :105,000,141,145,027,032,249
5949 :037,023,056,152,237,156,210
5955 :027,141,141,027,200,076,167
5961 :151,022,200,177,155,201,211
5967 :061,240,004,136,076,106,190
5973 :022,200,032,022,019,072,196
5979 :173,157,027,041,015,170,162
5985 :202,104,157,087,020,032,187
5991 :151,022,076,134,022,032,028
5997 :231,255,169,000,032,189,217
6003 :255,169,015,162,008,160,116
6009 :015,032,186,255,032,192,065
6015 :255,144,001,096,032,166,053
6021 :009,169,062,032,210,255,102
```

```
6027 :032,077,016,240,025,162,179
6033 :015,032,201,255,176,012,068
6039 :169,236,160,026,032,086,092
6045 :009,169,013,032,210,255,077
6051 :032,231,255,076,150,009,148
6057 :032,231,255,169,000,032,120
6063 :189,255,169,015,162,008,205
6069 :160,015,032,186,255,032,093
6075 :192,255,176,228,032,166,212
6081 :009,162,015,032,198,255,096
6087 :032,077,016,032,231,255,074
6093 :169,001,141,184,026,096,054
6099 :173,141,002,201,005,240,205
6105 :005,173,158,027,208,037,057
6111 :032,166,009,169,162,160,153
6117 :026,032,086,009,032,077,235
6123 :016,208,003,076,150,009,185
6129 :169,001,141,158,027,141,110
6135 :184,026,169,000,133,155,146
6141 :169,028,133,156,076,017,064
6147 :024,165,038,133,155,165,171
6153 :039,133,156,160,001,076,062
6159 :019,024,160,000,162,000,124
6165 :189,236,026,032,181,009,182
6171 :209,155,240,002,162,255,026
6177 :200,208,011,230,156,165,235
6183 :156,205,189,026,240,002,089
6189 :176,035,232,236,191,026,173
6195 :208,224,024,152,101,155,147
6201 :133,038,165,156,105,000,142
6207 :133,039,056,165,038,237,219
6213 :191,026,133,038,165,039,149
6219 :233,000,133,039,076,231,019
6225 :010,032,166,009,169,172,127
6231 :160,026,032,086,009,169,057
6237 :001,141,184,026,169,000,102
6243 :141,158,027,096,096,160,009
6249 :000,204,186,026,240,248,241
6255 :177,169,048,038,032,061,124
6261 :020,032,021,025,032,210,201
6267 :255,173,161,027,240,010,221
6273 :169,008,032,210,255,169,204
6279 :095,032,210,255,032,225,216
6285 :255,208,005,104,104,076,125
6291 :129,021,200,076,106,024,191
6297 :140,156,027,041,127,032,164
6303 :061,020,201,049,144,017,139
6309 :201,058,176,013,041,015,157
6315 :170,202,189,087,020,032,103
6321 :210,255,076,149,024,201,068
6327 :067,208,026,056,169,080,021
6333 :237,186,026,074,056,237,237
6339 :146,027,168,169,032,032,001
6345 :210,255,136,208,250,172,152
6351 :156,027,076,149,024,201,072
6357 :069,208,017,056,173,147,115
6363 :027,237,186,026,056,237,220
6369 :146,027,168,169,032,076,075
6375 :200,024,201,085,208,008,189
6381 :173,161,027,073,001,141,045
6387 :161,027,201,035,240,003,142
6393 :076,149,024,140,156,027,053
6399 :174,154,027,169,000,160,171
6405 :055,132,001,032,205,189,107
6411 :160,054,132,001,172,156,174
6417 :027,076,149,024,174,160,115
6423 :027,240,026,133,167,041,145
6429 :127,201,065,144,018,201,017
6435 :091,176,014,170,165,167,050
6441 :041,128,073,128,074,074,047
6447 :133,167,138,005,167,096,241
6453 :032,166,009,056,169,000,229
6459 :237,188,026,170,169,207,032
6465 :237,189,026,160,055,132,096
6471 :001,032,205,189,160,054,200
6477 :132,001,169,001,141,184,193
6483 :026,096,014,008,155,211,081
6489 :080,069,069,068,211,067,141
6495 :082,073,080,084,000,032,190
6501 :066,089,032,195,072,065,108
6507 :082,076,069,083,032,194,131
6513 :082,065,078,078,079,078,061
6519 :000,194,085,070,070,069,095
6525 :082,032,195,076,069,065,132
6531 :082,069,068,000,194,085,117
6537 :070,070,069,082,032,198,146
6543 :085,076,076,000,196,069,133
6549 :076,069,084,069,032,040,007
6555 :211,044,215,044,208,041,150
6561 :000,058,032,193,082,069,083
6567 :032,089,079,085,032,083,055
6573 :085,082,069,063,032,040,032
6579 :217,047,206,041,058,000,236
6585 :197,210,193,211,197,032,201
6591 :193,204,204,032,212,197,209
6597 :216,212,000,197,082,065,201
6603 :083,069,032,040,211,044,170
6609 :215,044,208,041,058,032,039
6615 :018,210,197,212,213,210,251
6621 :206,146,032,084,079,032,032
6627 :069,088,073,084,000,203,232
6633 :069,089,058,000,211,065,213
6639 :086,069,058,000,212,065,217
6645 :080,069,032,197,210,210,019
6651 :207,210,000,211,084,079,018
6657 :080,080,069,068,000,214,000
6663 :069,082,073,070,089,032,166
6669 :197,082,082,079,082,000,023
6675 :206,079,032,069,082,082,057
6681 :079,082,083,000,147,032,192
6687 :018,212,146,065,080,069,109
6693 :032,079,082,032,018,196,220
6699 :146,073,083,075,063,000,227
6705 :204,079,065,068,058,000,011
6711 :214,069,082,073,070,089,140
6717 :058,000,208,082,069,083,049
6723 :083,032,018,210,197,212,051
6729 :213,210,206,146,000,036,116
6735 :048,206,079,032,210,079,221
6741 :079,077,000,206,079,032,046
6747 :084,069,088,084,032,073,009
6753 :078,032,066,085,070,070,242
6759 :069,082,046,000,196,069,053
6765 :086,073,067,069,032,035,215
6771 :000,211,069,067,079,078,107
6777 :068,046,032,193,068,068,084
6783 :082,046,032,035,000,208,018
6789 :082,073,078,084,073,078,089
6795 :071,000,206,069,088,084,145
6801 :032,083,072,069,069,084,042
6807 :044,032,018,210,197,212,096
6813 :213,210,206,146,000,200,108
6819 :085,078,084,032,070,079,079
6825 :082,058,000,206,079,084,166
```

```
6831 :032,198,079,085,078,068,203
6837 :000,000,000,000,000,000,181
```

Program 2:

SpeedScript—VIC-20 Version

```
4609 :011,018,010,000,158,052,250
4615 :054,050,049,000,000,000,160
4621 :032,114,019,076,247,019,008
4627 :000,000,000,000,000,000,019
4633 :000,000,165,251,141,059,129
4639 :018,165,252,141,060,018,173
4645 :165,253,141,062,018,165,073
4651 :254,141,063,018,166,181,098
4657 :240,032,169,000,141,129,248
4663 :036,160,000,185,000,000,180
4669 :153,000,000,200,204,129,235
4675 :036,208,244,238,060,018,103
4681 :238,063,018,224,000,240,088
4687 :007,202,208,224,165,180,041
4693 :208,222,096,165,181,170,103
4699 :005,180,208,001,096,024,093
4705 :138,101,252,141,131,018,110
4711 :165,251,141,130,018,024,064
4717 :138,101,254,141,134,018,127
4723 :165,253,141,133,018,232,033
4729 :164,180,208,004,240,013,162
4735 :160,255,185,000,000,153,112
4741 :000,000,136,192,255,208,156
4747 :245,206,131,018,206,134,055
4753 :018,202,208,234,096,169,048
4759 :022,133,195,133,020,169,055
4765 :016,133,196,169,148,133,184
4771 :021,173,125,036,133,155,038
4777 :173,126,036,133,156,173,198
4783 :128,036,032,223,019,162,007
4789 :001,160,000,173,137,036,176
4795 :145,020,177,155,153,139,208
4801 :036,200,041,127,201,031,061
4807 :240,019,192,022,208,235,091
4813 :136,177,155,041,127,201,018
4819 :032,240,005,136,208,245,053
4825 :160,021,200,132,167,136,009
4831 :185,139,036,145,195,136,035
4837 :016,248,164,167,024,152,232
4843 :101,155,133,155,165,156,076
4849 :105,000,133,156,152,157,176
4855 :060,003,192,022,240,008,004
4861 :169,032,145,195,200,076,046
4867 :249,018,024,165,195,105,247
4873 :022,133,195,133,020,144,144
4879 :004,230,196,230,021,232,160
4885 :224,023,240,003,076,182,001
4891 :018,165,155,141,135,036,165
4897 :165,156,141,136,036,096,251
4903 :173,019,018,133,155,141,166
4909 :125,036,141,131,036,133,135
4915 :038,173,020,018,133,156,077
4921 :141,126,036,141,132,036,157
4927 :133,039,056,173,022,018,248
4933 :237,020,018,170,169,032,203
4939 :160,255,198,156,145,155,120
4945 :200,230,156,145,155,200,143
4951 :208,251,230,156,202,208,062
4957 :246,145,155,096,133,167,011
4963 :132,168,160,000,177,167,135
4969 :240,006,032,210,255,200,024
4975 :208,246,096,169,001,141,204
4981 :138,036,032,174,022,169,176
4987 :000,141,019,018,141,021,207
4993 :018,141,023,018,141,025,239
4999 :018,024,173,130,002,105,075
5005 :020,141,020,018,056,173,057
5011 :132,002,233,001,141,026,170
5017 :018,056,233,004,141,024,117
5023 :018,056,233,001,141,022,118
5029 :018,169,000,141,137,036,154
5035 :032,039,019,169,000,141,059
5041 :128,036,169,255,141,138,020
5047 :002,032,121,023,032,203,084
5053 :019,169,073,160,035,032,165
5059 :097,019,169,000,141,127,236
5065 :036,096,162,021,169,160,077
5071 :157,000,016,202,016,250,080
5077 :169,019,032,210,255,169,043
5083 :018,076,210,255,141,134,029
5089 :002,162,021,157,000,148,203
5095 :202,016,250,096,072,041,140
5101 :128,074,133,167,104,041,116
5107 :063,005,167,096,160,000,222
5113 :177,038,133,002,160,000,247
5119 :177,038,073,128,145,038,086
5125 :032,150,018,173,141,002,009
5131 :041,004,240,009,165,197,155
5137 :201,064,240,003,076,161,250
5143 :020,032,228,255,208,013,011
5149 :165,162,041,016,240,229,114
5155 :169,000,133,162,076,253,060
5161 :019,170,160,000,165,002,045
5167 :145,038,224,095,208,012,001
5173 :032,007,022,169,032,160,219
5179 :000,145,038,076,247,019,072
5185 :173,127,036,240,007,138,018
5191 :072,032,187,019,104,170,143
5197 :138,201,013,208,002,162,033
5203 :095,138,041,127,201,032,205
5209 :144,092,224,160,208,002,151
5215 :162,032,138,072,173,128,032
5221 :036,240,003,032,007,025,188
5227 :104,032,235,019,160,000,145
5233 :145,038,032,150,018,056,040
5239 :165,038,237,131,036,133,091
5245 :167,165,039,237,132,036,133
5251 :005,167,144,014,165,038,152
5257 :105,000,141,131,036,165,203
5263 :039,105,000,141,132,036,084
5269 :230,038,208,002,230,039,128
5275 :032,067,021,076,247,019,105
5281 :160,000,165,002,145,038,159
5287 :024,165,197,105,064,170,124
5293 :132,162,165,162,201,006,233
5299 :208,250,132,198,138,174,255
5305 :217,020,221,217,020,240,096
5311 :006,202,208,248,076,247,154
5317 :019,202,138,010,170,169,137
5323 :019,072,169,246,072,189,202
5329 :254,020,072,189,253,020,249
5335 :072,096,035,029,157,137,229
5341 :133,099,085,138,134,020,062
5347 :148,082,019,076,147,135,066
5353 :139,113,136,140,091,145,229
5359 :017,121,074,090,097,077,203
5365 :070,118,072,081,108,107,033
5371 :110,003,252,021,006,022,153
5377 :018,022,076,022,162,022,067
```

```
5383 :193,022,208,022,055,023,018
5389 :094,024,006,025,133,024,063
5395 :203,024,068,025,092,025,200
5401 :122,025,149,025,241,025,100
5407 :255,027,242,026,083,028,180
5413 :208,022,055,023,127,028,244
5419 :120,029,013,030,134,022,135
5425 :098,030,219,027,105,033,049
5431 :121,024,029,030,120,023,146
5437 :208,033,046,035,245,024,140
5443 :032,165,021,056,165,038,032
5449 :237,125,036,133,167,165,168
5455 :039,237,126,036,005,167,177
5461 :176,032,056,173,125,036,171
5467 :237,019,018,133,167,173,070
5473 :126,036,237,020,018,005,027
5479 :167,240,013,165,038,141,099
5485 :125,036,165,039,141,126,229
5491 :036,032,150,018,056,173,068
5497 :135,036,229,038,133,155,079
5503 :173,136,036,229,039,133,105
5509 :156,005,155,240,002,176,099
5515 :024,024,173,125,036,109,118
5521 :061,003,141,125,036,173,172
5527 :126,036,105,000,141,126,173
5533 :036,032,150,018,076,119,076
5539 :021,096,056,173,131,036,164
5545 :237,021,018,133,167,173,150
5551 :132,036,237,022,018,005,113
5557 :167,144,012,173,021,018,204
5563 :141,131,036,173,022,018,196
5569 :141,132,036,056,165,038,249
5575 :237,019,018,133,167,165,170
5581 :039,237,020,018,005,167,179
5587 :176,011,173,019,018,133,229
5593 :038,173,020,018,133,039,126
5599 :096,056,165,038,237,131,178
5605 :036,133,167,165,039,237,238
5611 :132,036,005,167,176,001,240
5617 :096,173,131,036,133,038,080
5623 :173,132,036,133,039,096,088
5629 :230,038,208,002,230,039,232
5635 :032,067,021,096,165,038,166
5641 :208,002,198,039,198,038,180
5647 :032,067,021,096,165,038,178
5653 :133,155,165,039,133,156,034
5659 :198,156,160,255,177,155,104
5665 :201,032,240,004,201,031,230
5671 :208,003,136,208,243,177,246
5677 :155,201,032,240,008,201,114
5683 :031,240,004,136,208,243,145
5689 :096,132,167,056,165,155,060
5695 :101,167,133,038,165,156,055
5701 :105,000,133,039,032,067,189
5707 :021,096,160,000,177,038,055
5713 :201,032,240,008,201,031,026
5719 :240,004,200,208,243,096,054
5725 :200,240,026,177,038,201,207
5731 :032,240,247,201,031,240,066
5737 :243,024,152,101,038,133,028
5743 :038,165,039,105,000,133,079
5749 :039,032,067,021,096,173,033
5755 :131,036,133,038,173,132,254
5761 :036,133,039,076,118,022,041
5767 :169,000,141,125,036,173,011
5773 :132,036,056,233,004,205,039
5779 :020,018,176,003,173,020,045
5785 :018,141,126,036,032,150,144
5791 :018,076,122,022,238,138,005
5797 :036,173,138,036,041,015,092
5803 :141,138,036,010,010,010,004
5809 :010,133,167,173,138,036,066
5815 :041,007,024,105,008,101,213
5821 :167,141,015,144,096,238,222
5827 :137,036,173,137,036,041,243
5833 :007,141,137,036,032,150,192
5839 :018,096,165,038,133,155,044
5845 :165,039,133,156,198,156,036
5851 :160,255,177,155,201,046,189
5857 :240,012,201,033,240,008,191
5863 :201,063,240,004,201,031,203
5869 :208,004,136,208,235,096,100
5875 :177,155,201,046,240,027,065
5881 :201,033,240,023,201,063,242
5887 :240,019,201,031,240,015,233
5893 :136,208,235,198,156,165,079
5899 :156,205,019,018,176,226,043
5905 :076,042,023,132,167,198,143
5911 :167,200,240,010,177,155,204
5917 :201,032,240,247,136,076,193
5923 :058,022,164,167,076,243,253
5929 :022,173,019,018,133,038,188
5935 :173,020,018,133,039,032,206
5941 :067,021,096,160,000,177,062
5947 :038,201,046,240,029,201,046
5953 :033,240,025,201,063,240,099
5959 :021,201,031,240,017,200,013
5965 :208,235,230,039,165,039,225
5971 :205,132,036,240,226,144,042
5977 :224,076,122,022,200,240,205
5983 :250,177,038,201,032,240,009
5989 :247,201,046,240,243,201,255
5995 :033,240,239,201,063,240,099
6001 :235,201,031,240,231,076,103
6007 :106,022,173,023,018,141,090
6013 :073,037,173,024,018,141,079
6019 :074,037,032,203,019,169,153
6025 :088,160,035,032,097,019,056
6031 :169,001,141,127,036,096,201
6037 :056,165,038,237,019,018,170
6043 :133,167,165,039,237,020,148
6049 :018,005,167,208,003,104,154
6055 :104,096,165,038,133,251,186
6061 :165,039,133,252,096,056,146
6067 :165,038,133,253,073,255,072
6073 :101,251,141,077,037,165,189
6079 :039,133,254,073,255,101,022
6085 :252,141,078,037,165,251,097
6091 :141,079,037,165,252,141,250
6097 :080,037,165,253,141,081,198
6103 :037,133,251,165,254,141,172
6109 :082,037,133,252,056,173,186
6115 :078,037,109,074,037,205,255
6121 :026,018,144,020,032,203,164
6127 :019,169,103,160,035,032,245
6133 :097,019,169,001,141,127,031
6139 :036,169,000,133,198,096,115
6145 :173,073,037,133,253,173,075
6151 :074,037,133,254,173,077,243
6157 :037,133,180,024,109,073,057
6163 :037,141,073,037,173,078,046
6169 :037,133,181,109,074,037,084
6175 :141,074,037,032,027,018,104
```

```
6181 :173,079,037,133,251,173,115
6187 :080,037,133,252,173,081,031
6193 :037,133,253,173,082,037,252
6199 :133,254,056,173,131,036,070
6205 :229,253,133,180,173,132,137
6211 :036,229,254,133,181,032,164
6217 :027,018,056,173,131,036,002
6223 :237,077,037,141,131,036,226
6229 :173,132,036,237,078,037,010
6235 :141,132,036,096,032,149,165
6241 :023,032,007,022,032,178,135
6247 :023,056,173,073,037,233,186
6253 :001,141,073,037,173,074,096
6259 :037,233,000,141,074,037,125
6265 :096,032,253,021,032,149,192
6271 :023,032,007,022,076,178,209
6277 :023,032,121,023,169,002,247
6283 :032,223,019,032,203,019,155
6289 :169,115,160,035,032,097,241
6295 :019,032,228,255,240,251,152
6301 :072,032,187,019,104,041,100
6307 :191,201,023,208,009,032,059
6313 :149,023,032,019,022,076,234
6319 :178,023,201,019,208,009,045
6325 :032,149,023,032,209,022,136
6331 :076,178,023,201,016,208,121
6337 :009,032,149,023,032,150,076
6343 :025,076,178,023,096,056,141
6349 :165,038,237,125,036,133,171
6355 :167,165,039,237,126,036,213
6361 :005,167,240,011,173,125,170
6367 :036,133,038,173,126,036,253
6373 :133,039,096,173,019,018,195
6379 :133,038,173,020,018,133,238

6385 :039,032,067,021,096,160,144
6391 :005,140,099,037,032,007,055
6397 :025,172,099,037,136,208,162
6403 :244,076,077,022,024,165,099
6409 :038,133,251,105,001,133,158
6415 :253,165,039,133,252,105,194
6421 :000,133,254,056,173,131,000
6427 :036,229,253,133,180,173,007
6433 :132,036,229,254,133,181,230
6439 :201,255,208,006,169,001,111
6445 :133,180,230,181,032,088,121
6451 :018,160,000,169,032,145,063
6457 :038,238,131,036,208,003,199
6463 :238,132,036,076,118,022,173
6469 :173,128,036,073,006,141,114
6475 :128,036,096,169,130,160,026
6481 :035,032,097,019,032,228,012
6487 :255,240,251,201,089,096,195
6493 :032,203,019,169,141,160,049
6499 :035,032,097,019,169,002,197
6505 :032,223,019,032,078,025,002
6511 :240,004,032,187,019,096,177
6517 :162,255,154,076,013,018,027
6523 :160,000,177,038,201,031,218
6529 :240,015,200,208,247,230,245
6535 :039,165,039,205,132,036,239
6541 :144,238,076,122,022,200,175
6547 :076,106,022,165,038,133,175
6553 :155,165,039,133,156,198,231
6559 :156,160,255,177,155,201,239
6565 :031,240,017,136,192,255,012
6571 :208,245,198,156,165,156,019
6577 :205,020,018,176,236,076,140

6583 :042,023,056,152,101,155,200
6589 :133,155,169,000,101,156,135
6595 :133,156,056,165,155,229,065
6601 :038,133,167,165,156,229,065
6607 :039,005,167,208,018,132,008
6613 :167,024,165,155,229,167,096
6619 :133,155,165,156,233,000,037
6625 :133,156,076,168,025,165,180
6631 :155,133,038,165,156,133,243
6637 :039,032,067,021,096,173,153
6643 :141,002,041,001,208,003,127
6649 :032,121,023,032,203,019,167
6655 :169,152,160,035,032,097,132
6661 :019,160,000,177,038,073,216
6667 :128,145,038,032,150,018,010
6673 :160,000,177,038,073,128,081
6679 :145,038,169,002,032,223,120
6685 :019,032,228,255,240,251,030
6691 :009,064,201,087,208,009,101
6697 :032,079,026,032,077,022,053
6703 :076,094,026,201,083,208,223
6709 :009,032,079,026,032,056,031
6715 :023,076,094,026,201,080,047
6721 :208,009,032,079,026,032,195
6727 :123,025,076,094,026,076,235
6733 :187,019,165,038,133,253,104
6739 :141,244,036,165,039,133,073
6745 :254,141,245,036,096,056,149
6751 :165,038,133,251,237,244,139
6757 :036,141,077,037,165,039,084
6763 :133,252,237,245,036,141,127
6769 :078,037,032,201,023,173,145
6775 :244,036,133,038,173,245,220
6781 :036,133,039,032,150,018,021

6787 :076,006,026,169,022,229,147
6793 :211,141,133,036,169,000,059
6799 :141,101,037,160,000,169,239
6805 :166,032,210,255,169,157,114
6811 :032,210,255,140,134,036,194
6817 :032,228,255,240,251,172,059
6823 :134,036,133,167,169,032,070
6829 :032,210,255,169,157,032,004
6835 :210,255,165,167,201,013,166
6841 :240,046,201,020,208,015,147
6847 :136,016,004,200,076,148,003
6853 :026,169,157,032,210,255,022
6859 :076,148,026,041,127,201,054
6865 :032,144,192,204,133,036,182
6871 :240,187,165,167,153,179,026
6877 :036,032,210,255,169,000,155
6883 :133,212,200,076,148,026,254
6889 :032,210,255,169,000,153,028
6895 :179,036,152,096,032,203,169
6901 :019,169,180,160,035,032,072
6907 :097,019,032,087,027,176,177
6913 :033,173,019,018,133,155,020
6919 :173,020,018,133,156,174,169
6925 :131,036,172,132,036,169,177
6931 :155,032,216,255,176,010,095
6937 :032,183,255,041,191,208,167
6943 :003,076,068,028,240,036,226
6949 :173,086,027,201,008,144,164
6955 :006,032,167,033,076,067,168
6961 :027,173,086,027,201,001,052
6967 :240,249,032,203,019,169,199
6973 :186,160,035,032,097,019,078
6979 :169,001,141,127,036,096,125
```

```
6985 :032,203,019,169,197,160,085
6991 :035,032,097,019,076,067,149
6997 :027,000,032,134,026,240,032
7003 :024,169,228,160,035,032,227
7009 :097,019,032,228,255,240,200
7015 :251,162,008,201,068,240,009
7021 :012,162,001,201,084,240,041
7027 :006,032,187,019,104,104,055
7033 :096,142,086,027,169,001,130
7039 :160,000,032,186,255,160,152
7045 :000,224,001,240,042,185,057
7051 :179,036,201,064,208,007,066
7057 :185,180,036,201,058,240,021
7063 :028,169,048,141,219,036,024
7069 :169,058,141,220,036,185,198
7075 :179,036,153,221,036,200,220
7081 :204,134,036,144,244,240,147
7087 :242,200,076,192,027,185,073
7093 :179,036,153,219,036,200,236
7099 :204,134,036,208,244,140,129
7105 :243,036,032,203,019,169,127
7111 :179,160,036,032,097,019,210
7117 :173,243,036,162,219,160,174
7123 :036,032,189,255,169,013,137
7129 :076,210,255,032,203,019,244
7135 :169,175,160,035,032,097,123
7141 :019,032,228,255,240,251,230
7147 :032,235,019,009,128,072,218
7153 :173,128,036,240,003,032,085
7159 :007,025,032,187,019,104,109
7165 :076,111,020,056,165,038,207
7171 :237,019,018,133,155,165,218
7177 :039,237,020,018,005,155,227
7183 :133,155,240,004,169,006,210

7189 :133,155,032,203,019,169,220
7195 :248,160,035,032,097,019,106
7201 :165,155,032,223,019,032,147
7207 :087,027,165,155,208,003,172
7213 :032,039,019,169,000,166,214
7219 :038,164,039,032,213,255,024
7225 :144,003,076,035,027,142,228
7231 :131,036,140,132,036,032,058
7237 :231,255,032,203,019,169,210
7243 :218,160,035,032,097,019,124
7249 :076,067,027,032,203,019,249
7255 :169,254,160,035,032,097,066
7261 :019,032,087,027,169,001,172
7267 :174,019,018,172,020,018,008
7273 :032,213,255,032,183,255,051
7279 :041,191,240,209,032,203,003
7285 :019,169,205,160,035,032,225
7291 :097,019,076,067,027,169,066
7297 :147,032,210,255,169,013,187
7303 :032,210,255,032,172,028,096
7309 :169,013,032,210,255,169,221
7315 :006,160,036,032,097,019,241
7321 :032,228,255,201,013,208,066
7327 :249,076,187,019,032,204,158
7333 :255,169,001,032,195,255,048
7339 :096,032,231,255,169,001,187
7345 :162,008,160,000,032,186,213
7351 :255,169,002,162,021,160,184
7357 :036,032,189,255,032,192,157
7363 :255,176,221,162,001,032,018
7369 :198,255,032,207,255,032,156
7375 :207,255,032,207,255,032,171
7381 :183,255,208,202,032,207,020

7387 :255,240,197,032,204,255,122
7393 :032,228,255,201,032,208,157
7399 :005,032,228,255,240,251,218
7405 :162,001,032,198,255,032,149
7411 :207,255,072,032,207,255,247
7417 :168,104,170,152,032,205,056
7423 :221,169,032,032,210,255,150
7429 :032,207,255,240,006,032,009
7435 :210,255,076,005,029,169,243
7441 :013,032,210,255,076,209,044
7447 :028,162,000,142,246,036,125
7453 :142,247,036,142,248,036,112
7459 :056,177,155,233,048,144,080
7465 :042,201,010,176,038,014,010
7471 :246,036,046,247,036,014,160
7477 :246,036,046,247,036,014,166
7483 :246,036,046,247,036,014,172
7489 :246,036,046,247,036,013,177
7495 :246,036,141,246,036,200,208
7501 :208,212,230,156,076,035,226
7507 :029,248,173,246,036,013,060
7513 :247,036,240,023,056,173,096
7519 :246,036,233,001,141,246,230
7525 :036,173,247,036,233,000,058
7531 :141,247,036,238,248,036,029
7537 :076,085,029,173,248,036,248
7543 :216,096,056,173,073,037,002
7549 :237,023,018,141,075,037,144
7555 :173,074,037,237,024,018,182
7561 :141,076,037,013,075,037,004
7567 :208,016,032,203,019,169,022
7573 :031,160,036,032,097,019,012
7579 :169,001,141,127,036,096,213
7585 :024,165,038,133,251,109,113

7591 :075,037,133,253,165,039,101
7597 :133,252,109,076,037,133,145
7603 :254,056,173,131,036,229,034
7609 :251,133,180,173,132,036,066
7615 :229,252,133,181,024,101,087
7621 :254,205,022,018,144,016,088
7627 :032,203,019,169,023,160,041
7633 :036,032,097,019,169,001,051
7639 :141,127,036,096,032,088,223
7645 :018,024,173,075,037,133,169
7651 :180,109,131,036,141,131,187
7657 :036,173,076,037,133,181,101
7663 :109,132,036,141,132,036,057
7669 :165,038,133,253,165,039,014
7675 :133,254,173,023,018,133,217
7681 :251,173,024,018,133,252,084
7687 :032,027,018,032,067,021,204
7693 :096,160,000,177,038,170,142
7699 :200,177,038,136,145,038,241
7705 :200,138,145,038,096,160,034
7711 :000,177,038,041,063,240,078
7717 :010,201,027,176,006,177,122
7723 :038,073,064,145,038,076,221
7729 :253,021,133,167,041,064,216
7735 :010,005,167,041,191,133,090
7741 :167,041,032,073,032,010,160
7747 :005,167,096,005,075,066,225
7753 :005,058,002,001,027,014,180
7759 :015,018,000,000,000,000,112
7765 :000,032,203,019,169,075,071
7771 :160,036,076,097,019,076,043
7777 :121,031,169,004,141,102,153
7783 :037,160,007,173,141,002,111
```

```
7789 :041,001,240,054,032,203,168
7795 :019,169,050,160,036,032,069
7801 :097,019,032,228,255,240,224
7807 :251,056,233,048,201,003,151
7813 :144,217,201,008,176,213,068
7819 :141,102,037,032,203,019,161
7825 :169,059,160,036,032,097,186
7831 :019,032,228,255,240,251,152
7837 :056,233,048,048,190,201,165
7843 :010,176,186,168,169,001,105
7849 :174,102,037,032,186,255,187
7855 :169,000,032,189,255,032,084
7861 :086,030,032,192,255,162,170
7867 :001,032,201,255,144,003,055
7873 :076,121,031,173,019,018,119
7879 :133,155,173,020,018,133,063
7885 :156,162,000,142,084,037,018
7891 :142,083,037,142,103,037,243
7897 :142,104,037,142,097,037,008
7903 :189,070,030,157,089,037,027
7909 :232,224,007,208,245,169,034
7915 :255,141,098,037,160,000,158
7921 :177,155,016,003,076,071,227
7927 :032,201,031,240,034,153,170
7933 :249,036,200,238,096,037,085
7939 :173,096,037,205,090,037,129
7945 :144,230,136,140,130,036,057
7951 :177,155,201,032,240,009,061
7957 :206,096,037,136,208,244,180
7963 :172,130,036,140,130,036,159
7969 :152,056,101,155,133,155,017
7975 :165,156,105,000,133,156,242
7981 :160,000,173,098,037,201,202
7987 :255,208,003,032,234,031,046

7993 :032,036,032,173,130,036,240
7999 :141,129,036,169,249,133,152
8005 :169,169,036,133,170,032,010
8011 :105,034,032,051,032,173,246
8017 :098,037,205,093,037,144,183
8023 :003,032,146,031,056,165,008
8029 :155,237,131,036,133,167,184
8035 :165,156,237,132,036,005,062
8041 :167,240,035,144,033,169,125
8047 :000,141,083,037,141,092,093
8053 :037,032,146,031,032,225,108
8059 :255,240,251,169,013,032,059
8065 :210,255,032,204,255,169,230
8071 :001,032,195,255,032,187,069
8077 :019,096,076,239,030,056,145
8083 :173,091,037,237,098,037,052
8089 :168,136,136,240,010,048,123
8095 :008,169,013,032,210,255,078
8101 :136,208,248,173,084,037,027
8107 :240,019,141,129,036,173,141
8113 :087,037,133,169,173,088,096
8119 :037,133,170,032,036,032,111
8125 :032,105,034,169,013,032,062
8131 :210,255,032,210,255,032,165
8137 :210,255,173,095,037,208,155
8143 :026,032,204,255,032,203,191
8149 :019,169,084,160,036,032,201
8155 :097,019,032,228,255,240,066
8161 :251,032,086,030,162,001,019
8167 :032,201,255,238,097,037,067
8173 :173,083,037,240,019,141,162
8179 :129,036,173,085,037,133,068
8185 :169,173,086,037,133,170,249

8191 :032,036,032,032,105,034,014
8197 :169,013,032,210,255,172,088
8203 :092,037,140,098,037,136,039
8209 :136,240,015,048,013,169,126
8215 :013,032,210,255,032,225,022
8221 :255,240,018,136,208,243,105
8227 :096,169,032,172,089,037,118
8233 :140,096,037,032,210,255,043
8239 :136,208,250,096,172,094,235
8245 :037,024,152,109,098,037,254
8251 :141,098,037,169,013,032,037
8257 :210,255,136,208,250,096,196
8263 :141,100,037,041,127,032,037
8269 :051,030,201,049,144,007,047
8275 :201,058,176,003,076,073,158
8281 :033,174,160,032,221,160,101
8287 :032,240,012,202,208,248,013
8293 :206,096,037,173,100,037,238
8299 :076,248,030,202,138,010,043
8305 :170,140,099,037,169,032,248
8311 :072,169,132,072,189,172,157
8317 :032,072,189,171,032,072,181
8323 :096,056,173,099,037,101,181
8329 :155,133,155,165,156,105,238
8335 :000,133,156,076,239,030,009
8341 :177,155,201,031,240,001,186
8347 :136,140,099,037,096,010,161
8353 :087,065,076,082,084,066,109
8359 :083,078,072,070,190,032,180
8365 :199,032,208,032,218,032,126
8371 :228,032,238,032,248,032,221
8377 :002,033,005,033,043,033,078
8383 :169,000,141,095,037,200,065
8389 :076,149,032,200,169,001,056

8395 :141,103,037,076,149,032,229
8401 :200,032,024,029,141,089,212
8407 :037,076,149,032,200,032,229
8413 :024,029,141,090,037,076,106
8419 :149,032,200,032,024,029,181
8425 :141,092,037,076,149,032,248
8431 :200,032,024,029,141,093,246
8437 :037,076,149,032,200,032,003
8443 :024,029,141,094,037,076,140
8449 :149,032,076,146,031,056,235
8455 :152,101,155,141,085,037,166
8461 :165,156,105,000,141,086,154
8467 :037,032,035,033,056,152,108
8473 :237,099,037,141,083,037,147
8479 :200,076,149,032,200,177,097
8485 :155,201,031,208,249,136,249
8491 :096,056,152,101,155,141,232
8497 :087,037,165,156,105,000,087
8503 :141,088,037,032,035,033,165
8509 :056,152,237,099,037,141,015
8515 :084,037,200,076,149,032,133
8521 :200,177,155,201,061,240,083
8527 :004,136,076,104,032,200,119
8533 :032,024,029,072,173,100,003
8539 :037,041,015,170,202,104,148
8545 :157,077,030,032,149,032,062
8551 :076,132,032,032,231,255,093
8557 :169,000,032,189,255,169,155
8563 :015,162,008,160,015,032,251
8569 :186,255,032,192,255,144,161
8575 :001,096,032,203,019,169,135
8581 :062,032,210,255,032,134,090
8587 :026,240,025,162,015,032,127
```

```
8593 :201,255,176,012,169,179,113
8599 :160,036,032,097,019,169,152
8605 :013,032,210,255,032,231,162
8611 :255,076,187,019,032,231,195
8617 :255,169,000,032,189,255,045
8623 :169,015,162,008,160,015,192
8629 :032,186,255,032,192,255,109
8635 :176,228,032,203,019,162,239
8641 :015,032,198,255,032,134,091
8647 :026,032,231,255,169,001,145
8653 :141,127,036,096,173,141,151
8659 :002,201,005,240,005,173,069
8665 :101,037,208,040,032,203,070
8671 :019,169,105,160,036,032,232
8677 :097,019,032,134,026,208,233
8683 :004,032,187,019,096,169,230
8689 :001,141,101,037,141,127,021
8695 :036,173,019,018,133,155,013
8701 :173,020,018,133,156,076,061
8707 :018,034,165,038,133,155,034
8713 :165,039,133,156,160,001,151
8719 :076,020,034,160,000,162,211
8725 :000,189,179,036,032,235,180
8731 :019,209,155,240,002,162,046
8737 :255,200,208,011,230,156,069
8743 :165,156,205,132,036,240,205
8749 :002,176,035,232,236,134,092
8755 :036,208,224,024,152,101,028
8761 :155,133,038,165,156,105,041
8767 :000,133,039,056,165,038,238
8773 :237,134,036,133,038,165,044
8779 :039,233,000,133,039,076,083
8785 :067,021,032,203,019,169,080
8791 :115,160,036,032,097,019,034
8797 :169,001,141,127,036,169,224
8803 :000,141,101,037,096,096,058
8809 :160,000,204,129,036,240,106
8815 :248,177,169,048,038,032,055
8821 :051,030,032,015,035,032,056
8827 :210,255,173,104,037,240,118
8833 :010,169,008,032,210,255,045
8839 :169,095,032,210,255,032,160
8845 :225,255,208,005,104,104,018
8851 :076,121,031,200,076,107,246
8857 :034,140,099,037,041,127,119
8863 :032,051,030,201,049,144,154
8869 :017,201,058,176,013,041,159
8875 :015,170,202,189,077,030,086
8881 :032,210,255,076,150,034,166
8887 :201,067,208,026,056,169,142
8893 :080,237,129,036,074,056,033
8899 :237,089,037,168,169,032,159
8905 :032,210,255,136,208,250,012
8911 :172,099,037,076,150,034,007
8917 :201,069,208,017,056,173,169
8923 :090,037,237,129,036,056,036
8929 :237,089,037,168,169,032,189
8935 :076,201,034,201,085,208,012
8941 :008,173,104,037,073,001,121
8947 :141,104,037,201,035,240,233
8953 :003,076,150,034,174,097,015
8959 :037,169,000,032,205,221,151
8965 :160,054,132,001,172,099,111
8971 :037,076,150,034,174,103,073
8977 :037,240,026,133,167,041,149
8983 :127,201,065,144,018,201,011
8989 :091,176,014,170,165,167,044
8995 :041,128,073,128,074,074,041
9001 :133,167,138,005,167,096,235
9007 :032,203,019,056,173,021,039
9013 :018,237,131,036,170,173,050
9019 :022,018,237,132,036,032,024
9025 :205,221,169,001,141,127,161
9031 :036,096,014,008,144,211,068
9037 :080,069,069,068,211,067,129
9043 :082,073,080,084,000,194,084
9049 :085,070,070,069,082,032,241
9055 :195,076,069,065,082,069,139
9061 :068,000,194,085,070,070,076
9067 :069,082,032,198,085,076,137
9073 :076,000,196,069,076,069,087
9079 :084,069,032,040,211,044,087
9085 :215,044,208,041,000,058,179
9091 :211,085,082,069,063,032,161
9097 :217,047,206,000,197,210,246
9103 :193,211,197,032,212,197,161
9109 :216,212,000,197,082,065,153
9115 :083,069,032,040,211,044,122
9121 :215,044,208,041,058,060,019
9127 :210,197,212,213,210,206,135
9133 :062,000,203,069,089,058,142
9139 :000,211,065,086,069,058,156
9145 :000,212,065,080,069,032,131
9151 :197,210,210,207,210,000,201
9157 :211,084,079,080,080,069,032
9163 :068,000,214,069,082,073,197
9169 :070,089,032,197,082,082,249
9175 :079,082,000,206,079,032,181
9181 :069,082,082,079,082,083,186
9187 :000,147,032,018,212,146,014
9193 :065,080,069,032,079,082,128
9199 :032,018,196,146,073,083,019
9205 :075,063,000,204,079,065,219
9211 :068,058,000,214,069,082,230
9217 :073,070,089,058,000,208,243
9223 :082,069,083,083,032,018,118
9229 :210,197,212,213,210,206,237
9235 :146,000,036,048,206,079,022
9241 :032,210,079,079,077,000,246
9247 :206,079,032,084,069,088,077
9253 :084,032,073,078,032,066,146
9259 :085,070,070,069,082,046,209
9265 :000,196,069,086,073,067,028
9271 :069,032,035,000,211,069,215
9277 :067,079,078,068,046,032,175
9283 :193,068,068,082,046,032,044
9289 :035,000,208,082,073,078,037
9295 :084,073,078,071,000,206,079
9301 :069,088,084,032,083,072,001
9307 :069,069,084,044,032,146,023
9313 :210,197,212,213,210,206,065
9319 :018,000,200,085,078,084,056
9325 :032,070,079,082,058,000,174
9331 :206,079,084,032,198,079,025
9337 :085,078,068,000,000,000,096
```

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.

MLX For VIC And 64

(Article on page 171.)

Program 1: MLX—64 Version

```
100 PRINT"{CLR}{CYN}";CHR$(142);CHR$(8);:
    POKE53281,1:POKE53280,1          :rem 73
101 POKE 788,52:REM DISABLE RUN/STOP
                                     :rem 119
110 PRINT"{RVS}{40 SPACES}";         :rem 176
120 PRINT"{RVS}{15 SPACES}{RIGHT}{OFF}
    {*}£{RVS}{RIGHT} {RIGHT}{2 SPACES}
    {*}{OFF}{*}£{RVS}£{RVS}
    {13 SPACES}";                    :rem 250
130 PRINT"{RVS}{15 SPACES}{RIGHT} {G}
    {RIGHT} {2 RIGHT} {OFF}£{RVS}£{*}
    {OFF}{*}{RVS}{13 SPACES}";       :rem 35
140 PRINT"{RVS}{40 SPACES}"          :rem 120
200 PRINT"{2 DOWN}{PUR}{BLK}{3 SPACES}A F
    AILSAFE MACHINE LANGUAGE EDITOR
    {5 DOWN}"                        :rem 130
210 PRINT"{5}{2 UP}STARTING ADDRESS?
    {8 SPACES}{9 LEFT}";             :rem 143
215 INPUTS:F=1-F:C$=CHR$(31+119*F:rem 125
220 IFS<256OR(S>40960ANDS<49152)ORS>53247
    THENGOSUB3000:GOTO210            :rem 235
225 PRINT:PRINT:PRINT                :rem 180
230 PRINT"{5}{2 UP}ENDING ADDRESS?
    {8 SPACES}{9 LEFT}";:INPUTE:F=1-F:C$=
    CHR$(31+119*F)                   :rem 20
240 IFE<256OR(E>40960ANDE<49152)ORE>53247
    THENGOSUB3000:GOTO230            :rem 183
250 IFE<STHENPRINTC$;"{RVS}ENDING < START
    {2 SPACES}":GOSUB1000:GOTO 230
                                     :rem 176
260 PRINT:PRINT:PRINT                :rem 179
300 PRINT"{CLR}";CHR$(14):AD=S:POKEV+21,0
                                     :rem 225
310 PRINTRIGHT$("0000"+MID$(STR$(AD),2),5
    );":";:FORJ=1TO6                 :rem 234
320 GOSUB570:IFN=-1THENJ=J+N:GOTO320
                                     :rem 228
390 IFN=-211THEN 710                  :rem 62
400 IFN=-204THEN 790                  :rem 64
410 IFN=-206THENPRINT:INPUT"{DOWN}ENTER N
    EW ADDRESS";ZZ                   :rem 44
415 IFN=-206THENIFZZ<SORZZ>ETHENPRINT"
    {RVS}OUT OF RANGE":GOSUB1000:GOTO410
                                     :rem 225
417 IFN=-206THENAD=ZZ:PRINT:GOTO310
                                     :rem 238
420 IF N<>-196 THEN 480              :rem 133
430 PRINT:INPUT"DISPLAY:FROM";F:PRINT,"TO
    ";:INPUTT                        :rem 234
440 IFF<SORF>EORT<SORT>ETHENPRINT"AT LEAS
    T";S;"{LEFT}, NOT MORE THAN";E:GOTO43
    0                                :rem 159
450 FORI=FTOTSTEP6:PRINT:PRINTRIGHT$("000
    0"+MID$(STR$(I),2),5);":";       :rem 30
451 FORK=0TO5:N=PEEK(I+K):PRINTRIGHT$("00
    "+MID$(STR$(N),2),3);",";        :rem 66
460 GETA$:IFA$>""THENPRINT:PRINT:GOTO310
                                     :rem 25
470 NEXTK:PRINTCHR$(20);:NEXTI:PRINT:PRIN
    T:GOTO310                        :rem 50
480 IFN<0 THEN PRINT:GOTO310         :rem 168
490 A(J)=N:NEXTJ                     :rem 199
500 CKSUM=AD-INT(AD/256)*256:FORI=1TO6:CK
    SUM=(CKSUM+A(I))AND255:NEXT      :rem 200
510 PRINTCHR$(18);:GOSUB570:PRINTCHR$(20)
                                     :rem 234
515 IFN=CKSUMTHEN530                 :rem 255
520 PRINT:PRINT"LINE ENTERED WRONG : RE-E
    NTER":PRINT:GOSUB1000:GOTO310:rem 176
530 GOSUB2000                        :rem 218
540 FORI=1TO6:POKEAD+I-1,A(I):NEXT:POKE54
    272,0:POKE54273,0                :rem 227
550 AD=AD+6:IF AD<E THEN 310         :rem 212
560 GOTO 710                         :rem 108
570 N=0:Z=0                           :rem 88
580 PRINT"{+}";                      :rem 79
581 GETA$:IFA$=""THEN581              :rem 95
585 PRINTCHR$(20);:A=ASC(A$):IFA=13ORA=44
    ORA=32THEN670                    :rem 229
590 IFA>128THENN=-A:RETURN           :rem 137
600 IFA<>20 THEN 630                  :rem 10
610 GOSUB690:IFI=1ANDT=44THENN=-1:PRINT"
    {LEFT} {LEFT}";:GOTO690          :rem 172
620 GOTO570                          :rem 109
630 IFA<48ORA>57THEN580              :rem 105
640 PRINTA$;:N=N*10+A-48             :rem 106
650 IFN>255 THEN A=20:GOSUB1000:GOTO600
                                     :rem 229
660 Z=Z+1:IFZ<3THEN580                :rem 71
670 IFZ=0THENGOSUB1000:GOTO570       :rem 114
680 PRINT",";:RETURN                 :rem 240
690 S%=PEEK(209)+256*PEEK(210)+PEEK(211)
                                     :rem 149
691 FORI=1TO3:T=PEEK(S%-I)            :rem 67
695 IFT<>44ANDT<>58THENPOKES%-I,32:NEXT
                                     :rem 205
700 PRINTLEFT$("{3 LEFT}",I-1);:RETURN
                                       :rem 7
710 PRINT"{CLR}{RVS}*** SAVE ***{3 DOWN}"
                                     :rem 236
720 INPUT"{DOWN} FILENAME";F$        :rem 228
730 PRINT:PRINT"{2 DOWN}{RVS}T{OFF}APE OR
     {RVS}D{OFF}ISK: (T/D)"          :rem 228
740 GETA$:IFA$<>"T"ANDA$<>"D"THEN740
                                     :rem 36
750 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
                                     :rem 158
760 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$
    ):POKE782,ZK/256                  :rem 3
762 POKE781,ZK-PEEK(782)*256:POKE780,LEN(
    T$):SYS65469                     :rem 109
763 POKE780,1:POKE781,DV:POKE782,1:SYS654
    66                               :rem 69
765 POKE254,S/256:POKE253,S-PEEK(254)*256
    :POKE780,253                     :rem 12
766 POKE782,E/256:POKE781,E-PEEK(782)*256
    :SYS65496                        :rem 124
770 IF(PEEK(783)AND1)OR(ST AND191)THEN780
                                     :rem 111
775 PRINT"{DOWN}DONE.":END           :rem 106
780 PRINT"{DOWN}ERROR ON SAVE.{2 SPACES}T
    RY AGAIN.":IFDV=1THEN720         :rem 171
781 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$
    ;E2$:CLOSE15:GOTO720             :rem 103
790 PRINT"{CLR}{RVS}*** LOAD ***{2 DOWN}"
                                     :rem 212
800 INPUT"{2 DOWN} FILENAME";F$      :rem 244
810 PRINT:PRINT"{2 DOWN}{RVS}T{OFF}APE OR
     {RVS}D{OFF}ISK: (T/D)"          :rem 227
820 GETA$:IFA$<>"T"ANDA$<>"D"THEN820
                                     :rem 34
830 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
                                     :rem 157
840 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$
    ):POKE782,ZK/256                  :rem 2
```

```
841 POKE781,ZK-PEEK(782)*256:POKE780,LEN(
    T$):SYS65469                    :rem 107
845 POKE780,1:POKE781,DV:POKE782,1:SYS654
    66                              :rem 70
850 POKE780,0:SYS65493              :rem 11
860 IF(PEEK(783)AND1)OR(ST AND191)THEN870
                                    :rem 111
865 PRINT"{DOWN}DONE.":GOTO310      :rem 96
870 PRINT"{DOWN}ERROR ON LOAD.{2 SPACES}T
    RY AGAIN.{DOWN}":IFDV=1THEN800
                                    :rem 172
880 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$
    ;E2$:CLOSE15:GOTO800            :rem 102
1000 REM BUZZER                     :rem 135
1001 POKE54296,15:POKE54277,45:POKE54278,
    165                             :rem 207
1002 POKE54276,33:POKE 54273,6:POKE54272,
    5                               :rem 42
1003 FORT=1TO200:NEXT:POKE54276,32:POKE54
    273,0:POKE54272,0:RETURN        :rem 202
2000 REM BELL SOUND                 :rem 78
2001 POKE54296,15:POKE54277,0:POKE54278,2
    47                              :rem 152
2002 POKE 54276,17:POKE54273,40:POKE54272
    ,0                              :rem 86
2003 FORT=1TO100:NEXT:POKE54276,16:RETURN
                                    :rem 57
3000 PRINTC$;"{RVS}NOT ZERO PAGE OR ROM":
    GOTO1000                        :rem 89
```

Program 2: MLX—VIC Version

```
100 PRINT"{CLR}{PUR}";CHR$(142);CHR$(8);
                                    :rem 181
101 POKE 788,194:REM DISABLE RUN/STOP
                                    :rem 174
110 PRINT"{RVS}{14 SPACES}"         :rem 117
120 PRINT"{RVS} {RIGHT}?{OFF}{*}£{RVS}
    {RIGHT} {RIGHT}{2 SPACES}{*}{OFF}
    {*}£{RVS}£{RVS} "               :rem 191
130 PRINT"{RVS} {RIGHT} {G}{RIGHT}
    {2 RIGHT} {OFF}£{RVS}£{*}{OFF}
    {*}{RVS} "                      :rem 232
140 PRINT"{RVS}{14 SPACES}"         :rem 120
200 PRINT"{2 DOWN}{PUR}{BLK}A FAILSAFE MA
    CHINE":PRINT"LANGUAGE EDITOR{5 DOWN}"
                                    :rem 141
210 PRINT"{BLK}{3 UP}STARTING ADDRESS":IN
    PUTS:F=1-F:C$=CHR$(31+119*F)    :rem 97
220 IFS<256ORS>32767THENGOSUB3000:GOTO210
                                      :rem 2
225 PRINT:PRINT:PRINT:PRINT         :rem 123
230 PRINT"{BLK}{3 UP}ENDING ADDRESS":INPU
    TE:F=1-F:C$=CHR$(31+119*F)      :rem 158
240 IFE<256ORE>32767THENGOSUB3000:GOTO230
                                    :rem 234
250 IFE<STHENPRINTC$;"{RVS}ENDING < START
    {2 SPACES}":GOSUB1000:GOTO 230
                                    :rem 176
260 PRINT:PRINT:PRINT               :rem 179
300 PRINT"{CLR}";CHR$(14):AD=S      :rem 56
310 PRINTRIGHT$("0000"+MID$(STR$(AD),2),5
    );":";:FORJ=1TO6                :rem 234
320 GOSUB570:IFN=-1THENJ=J+N:GOTO320
                                    :rem 228
390 IFN=-211THEN 710                :rem 62
400 IFN=-204THEN 790                :rem 64
410 IFN=-206THENPRINT:INPUT"{DOWN}ENTER N
    EW ADDRESS";ZZ                  :rem 44
415 IFN=-206THENIFZZ<SORZZ>ETHENPRINT"
    {RVS}OUT OF RANGE":GOSUB1000:GOTO410
                                    :rem 225
417 IFN=-206THENAD=ZZ:PRINT:GOTO310
                                    :rem 238
420 IF N<>-196 THEN 480             :rem 133
430 PRINT:INPUT"DISPLAY:FROM";F:PRINT,"TO
    ";:INPUTT                       :rem 234
440 IFF<SORF>EORT<SORT>ETHENPRINT"AT LEAS
    T";S;"{LEFT}, NOT MORE THAN";E:GOTO43
    0                               :rem 159
450 FORI=FTOTSTEP6:PRINT:PRINTRIGHT$("000
    0"+MID$(STR$(I),2),5);":";      :rem 30
455 FORK=0TO5:N=PEEK(I+K):IFK=3THENPRINTS
    PC(10);                         :rem 34
457 PRINTRIGHT$("00"+MID$(STR$(N),2),3);"
    ,";                             :rem 157
460 GETA$:IFA$>""THENPRINT:PRINT:GOTO310
                                    :rem 25
470 NEXTK:PRINTCHR$(20);:NEXTI:PRINT:PRIN
    T:GOTO310                       :rem 50
480 IFN<0 THEN PRINT:GOTO310        :rem 168
490 A(J)=N:NEXTJ                    :rem 199
500 CKSUM=AD-INT(AD/256)*256:FORI=1TO6:CK
    SUM=(CKSUM+A(I))AND255:NEXT     :rem 200
510 PRINTCHR$(18);:GOSUB570:PRINTCHR$(20)
                                    :rem 234
515 IFN=CKSUMTHEN530                :rem 255
520 PRINT:PRINT"LINE ENTERED WRONG":PRINT
    "RE-ENTER":PRINT:GOSUB1000:GOTO310
                                    :rem 129
530 GOSUB2000                       :rem 218
540 FORI=1TO6:POKEAD+I-1,A(I):NEXT:rem 80
550 AD=AD+6:IF AD<E THEN 310        :rem 212
560 GOTO 710                        :rem 108
570 N=0:Z=0                          :rem 88
580 PRINT"{+}";                     :rem 79
581 GETA$:IFA$=""THEN581             :rem 95
585 PRINTCHR$(20);:A=ASC(A$):IFA=13ORA=44
    ORA=32THEN670                   :rem 229
590 IFA>128THENN=-A:RETURN          :rem 137
600 IFA<>20 THEN 630                 :rem 10
610 GOSUB690:IFI=1ANDT=44THENN=-1:PRINT"
    {LEFT} {LEFT}";:GOTO690         :rem 172
620 GOTO570                         :rem 109
630 IFA<48ORA>57THEN580             :rem 105
640 PRINTA$;:N=N*10+A-48            :rem 106
650 IFN>255 THEN A=20:GOSUB1000:GOTO600
                                    :rem 229
660 Z=Z+1:IFZ<3THEN580               :rem 71
670 IFZ=0THENGOSUB1000:GOTO570      :rem 114
680 PRINT",";:RETURN                :rem 240
690 S%=PEEK(209)+256*PEEK(210)+PEEK(211)
                                    :rem 149
692 FORI=1TO3:T=PEEK(S%-I)           :rem 68
695 IFT<>44ANDT<>58THENPOKES%-I,32:NEXT
                                    :rem 205
700 PRINTLEFT$("{3 LEFT}",I-1);:RETURN
                                      :rem 7
710 PRINT"{CLR}{RVS}*** SAVE ***{3 DOWN}"
                                    :rem 236
720 INPUT"{DOWN} FILENAME";F$       :rem 228
730 PRINT:PRINT"{2 DOWN}{RVS}T{OFF}APE OR
     {RVS}D{OFF}ISK: (T/D)"         :rem 228
740 GETA$:IFA$<>"T"ANDA$<>"D"THEN740
                                    :rem 36
750 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
                                    :rem 158
760 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$
    ):POKE782,ZK/256                 :rem 3
```

```
762 POKE781,ZK-PEEK(782)*256:POKE780,LEN(
    T$):SYS65469                    :rem 109
763 POKE780,1:POKE781,DV:POKE782,1:SYS654
    66                              :rem 69
765 POKE254,S/256:POKE253,S-PEEK(254)*256
    :POKE780,253                    :rem 12
766 POKE782,E/256:POKE781,E-PEEK(782)*256
    :SYS65496                       :rem 124
770 IF(PEEK(783)AND1)OR(ST AND191)THEN780
                                    :rem 111
775 PRINT"{DOWN}DONE.":END          :rem 106
780 PRINT"{DOWN}ERROR ON SAVE.{2 SPACES}T
    RY AGAIN.":IFDV=1THEN720        :rem 171
781 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$
    ;E2$:CLOSE15:GOTO720            :rem 103
782 GOTO720                         :rem 115
790 PRINT"{CLR}{RVS}*** LOAD ***{2 DOWN}"
                                    :rem 212
800 INPUT"{2 DOWN} FILENAME";F$     :rem 244
810 PRINT:PRINT"{2 DOWN}{RVS}T{OFF}APE OR
     {RVS}D{OFF}ISK: (T/D)"         :rem 227
820 GETA$:IFA$<>"T"ANDA$<>"D"THEN820
                                    :rem 34
830 DV=1-7*(A$="D"):IFDV=8THENF$="0:"+F$
                                    :rem 157
840 T$=F$:ZK=PEEK(53)+256*PEEK(54)-LEN(T$
    ):POKE782,ZK/256                :rem 2
841 POKE781,ZK-PEEK(782)*256:POKE780,LEN(
    T$):SYS65469                    :rem 107
845 POKE780,1:POKE781,DV:POKE782,1:SYS654
    66                              :rem 70
850 POKE780,0:SYS65493              :rem 11
860 IF(PEEK(783)AND1)OR(ST AND191)THEN870
                                    :rem 111
865 PRINT"{DOWN}DONE.":GOTO310      :rem 96
870 PRINT"{DOWN}ERROR ON LOAD.{2 SPACES}T
    RY AGAIN.{DOWN}":IFDV=1THEN800
                                    :rem 172
880 OPEN15,8,15:INPUT#15,E1$,E2$:PRINTE1$
    ;E2$:CLOSE15:GOTO800            :rem 102
1000 REM BUZZER                     :rem 135
1001 POKE36878,15:POKE36874,190     :rem 206
1002 FORW=1TO300:NEXTW              :rem 117
1003 POKE36878,0:POKE36874,0:RETURN
                                    :rem 74
2000 REM BELL SOUND                 :rem 78
2001 FORW=15TO0STEP-1:POKE36878,W:POKE368
     76,240:NEXTW                   :rem 22
2002 POKE36876,0:RETURN             :rem 119
3000 PRINTC$;"{RVS}NOT ZERO PAGE OR ROM":
     GOTO1000                       :rem 89
```

64 Electronic Notepad

(Article on page 112.)

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 BO=254:SC=246:CH=14               :rem 170
5 POKE53280,BO:POKE53281,SC:POKE646,CH
                                    :rem 17
7 PRINT"{CLR}";TAB(10);"{RVS} ELECTRONIC
  {SPACE}NOTEPAD ",TAB(6)"{DOWN} LOADING
  {SPACE}DATA...PLEASE WAIT"        :rem 83
10 B=885:C=998:FORA=BTOC:READD:POKEA,D:NE
   XT                               :rem 221
12 B=49152:C=49407:FORA=BTOC:READD:POKEA,
   D:NEXT                           :rem 157
14 B=679:C=753:FORA=BTOC:READD:POKEA,D:NE
   XT                               :rem 215
16 PRINT"{CLR}{DOWN} ENTER 2 SECRET CODES
    (0-255):":PRINT"{DOWN} (SEPERATE EACH
    BY A COMMA)"                    :rem 100
17 PRINT"{DOWN} (ENTER 0'S IF NO SECRET C
   ODE){2 DOWN}":POKE646,PEEK(53281):INPU
   TA,B                             :rem 25
19 POKE646,CH:IFA<0ORA>255ORB<0ORB>255THE
   N16                              :rem 196
20 POKE249,A:POKE250,B:A=0:B=0:SYS679
                                    :rem 187
22 PRINT"{CLR}{DOWN} {RVS}D{OFF}ISK OR
   {RVS}T{OFF}APE?"                 :rem 86
23 GETA$:IFA$=""THEN23              :rem 237
24 IFA$="D"THEN30                   :rem 177
25 IFA$="T"THEN29                   :rem 202
26 GOTO22                            :rem 5
29 POKE49303,1:POKE49305,1:POKE49307,255:
   POKE49177,1:POKE49179,1          :rem 163
30 PRINT"{CLR}";TAB(10);"{RVS} ELECTRONIC
    NOTEPAD "                       :rem 15
40 PRINT"{2 DOWN}{3 SPACES}FUNCTION";TAB(
   30);"PRESS"                      :rem 143
45 PRINT"{3 SPACES}{8 T}";TAB(30);"
   {5 T}"                          :rem 198
50 PRINT"{2 DOWN} VIEW NOTEPAD PAGE";TAB(
   31);"F1"                         :rem 120
55 PRINT"{DOWN} CREATE NOTEPAD PAGE";TAB(
   31);"F3"                         :rem 231
60 PRINT"{DOWN} VIEW DISK DIRECTORY";TAB(
   31);"F5"                         :rem 36
65 PRINT"{DOWN} CHANGE PROGRAM OPTIONS";T
   AB(31);"F6"                      :rem 249
70 PRINT"{DOWN} END PROGRAM";TAB(31);"F8"
                                    :rem 252
75 GETA$:IFA$="{F1}"THEN200         :rem 166
80 IFA$="{F3}"THEN600                :rem 40
85 IFA$="{F5}"THEN400                :rem 44
86 IFA$="{F8}"THENPRINT"{CLR}":CLR:POKE24
   9,0:POKE250,0:END                :rem 238
87 IFA$="{F6}"THENRESTORE:GOTO1     :rem 102
90 GOTO75                            :rem 14
200 PRINT"{CLR}"                    :rem 246
210 SYS49152                        :rem 151
215 GOSUB500                        :rem 173
220 GETA$:IFA$=""ORA$="←"THEN220    :rem 49
230 GOTO30                           :rem 49
400 PRINT"{CLR}"                    :rem 248
405 IFPEEK(49303)=1THEN450          :rem 215
410 SYS885                           :rem 57
420 PRINTTAB(7);"{RVS}(PRESS ANY KEY)"
                                    :rem 194
430 GETA$:IFA$=""THEN430             :rem 81
440 GOTO30                           :rem 52
450 PRINT"{DOWN} NO DIRECTORY AVAILABLE":
    PRINT"{DOWN} PROGRAM IN {RVS}TAPE
    {OFF} MODE"                     :rem 12
460 PRINT"{2 DOWN} {RVS} (PRESS ANY KEY)
    {SPACE}"                        :rem 78
470 GETA$:IFA$=""THEN470            :rem 89
480 GOTO30                          :rem 56
500 CLOSE15:OPEN15,8,15:INPUT#15,A,B$,C,D
    :IFA>21THEN510                  :rem 218
505 RETURN                          :rem 122
510 PRINT:PRINT"{DOWN}{3 SPACES}{RVS} *DI
    SK ERROR* ":PRINT"{DOWN}{3 SPACES}
    {RVS} ";B$                      :rem 41
```

```
515 RETURN                              :rem 123
600 PRINT"{CLR}";TAB(9);"{RVS} CREATE NOT
    EPAD PAGE "                         :rem 3
605 PRINT"{2 DOWN} PRESS {RVS}F1{OFF} TO
    {SPACE}SAVE PAGE."                  :rem 83
610 PRINT"{DOWN} PRESS {RVS}F8{OFF} TO AB
    ORT PAGE."                          :rem 142
615 PRINT"{2 DOWN} (PRESS ANY KEY)"
                                        :rem 62
620 GETA$:IFA$=""THEN620                :rem 83
622 PRINT"{CLR}";                       :rem 57
625 GETA$:IFA$=""THENPRINT"{P} {2 LEFT}
    ";:GOTO625                          :rem 197
630 IFA$="{LEFT}"THEN625                :rem 119
635 IFA$="{RIGHT}"THEN625               :rem 252
636 IFA$="{HOME}"THENGOTO625            :rem 44
637 IFA$=CHR$(34)THEN625                :rem 86
638 IFA$="←"THENPRINT" {LEFT}";:GOTO625
                                        :rem 93
640 IFA$="{UP}"THENPRINT" {LEFT}{UP}";:GO
    TO625                               :rem 25
645 IFA$="{DOWN}"THENPRINT" {LEFT}{DOWN}"
    ;:GOTO625                           :rem 30
646 IFA$=CHR$(13)THENPRINT" ";CHR$(13);:G
    OTO625                              :rem 195
650 IFA$="{F1}"THEN680                  :rem 98
655 IFA$="{F8}"THEN30                   :rem 51
675 PRINTA$;" {LEFT}";:GOTO625          :rem 251
680 PRINT" ";:POKE648,60:POKE53272,245:SY
    S49278                              :rem 193
685 GOSUB500:IFA<21THENPOKE648,4:POKE5327
    2,21:GOTO30                         :rem 146
690 PRINT"{DOWN} {RVS} (PRESS ANY KEY) "
                                        :rem 66
695 GETA$:IFA$=""THEN695                :rem 107
697 POKE648,4:POKE53272,21:SYS49374:GOTO6
    25                                  :rem 38
885 DATA 169,001,162,008,160,000        :rem 36
891 DATA 032,186,255,169,002,162        :rem 51
897 DATA 224,160,003,032,189,255        :rem 51
903 DATA 032,192,255,162,001,032        :rem 30
909 DATA 198,255,032,207,255,032        :rem 53
915 DATA 207,255,032,207,255,032        :rem 41
921 DATA 207,255,240,058,032,204        :rem 37
927 DATA 255,032,228,255,201,032        :rem 41
933 DATA 208,005,032,228,255,240        :rem 39
939 DATA 251,162,001,032,198,255        :rem 48
945 DATA 032,207,255,072,032,207        :rem 41
951 DATA 255,168,104,170,152,032        :rem 42
957 DATA 205,189,169,032,032,210        :rem 49
963 DATA 255,032,207,255,240,006        :rem 42
969 DATA 032,210,255,076,196,003        :rem 50
975 DATA 169,013,032,210,255,076        :rem 48
981 DATA 149,003,169,001,032,195        :rem 46
987 DATA 255,032,204,255,096,036        :rem 57
993 DATA 048,013,013,013,013,013        :rem 27
49152 DATA32,54,192, 160, 0, 162, 0, 32
                                        :rem 178
49160 DATA207,255,201,13,240,8,157, 240
                                        :rem 127
49168 DATA193,232,200,76,7,192, 152, 72
                                        :rem 97
49176 DATA169,8,162,8, 160, 1, 32, 186
                                        :rem 206
49184 DATA255,104,162,240,160,193,32,189
                                        :rem 243
49192 DATA255,169,0,162, 0, 160, 4, 32
                                        :rem 188
49200 DATA213,255,96,234,234,234,162, 0
                                        :rem 130
49208 DATA173,134,2,157,0,216,232, 208
                                        :rem 81
49216 DATA250,238,61,192,172,61,192,192
                                        :rem 196
49224 DATA220,208,235,169,216,141,61,192
                                        :rem 237
49232 DATA162,0,189,94,192,32,210, 255
                                        :rem 90
49240 DATA232,224,29,208,245,96,13, 17
                                        :rem 91
49248 DATA69,78,84,69, 82, 32, 78, 79
                                        :rem 192
49256 DATA84,69,80, 65, 68, 32, 80, 65
                                        :rem 172
49264 DATA71,69,32, 78, 65, 77, 69, 32
                                        :rem 175
49272 DATA32,32,13,234,234,234, 32, 187
                                        :rem 84
49280 DATA192,160,0,162,0, 32, 207, 255
                                        :rem 27
49288 DATA201,13,240,8,157,240,194, 232
                                        :rem 138
49296 DATA200,76,133,192,152,72,169, 8
                                        :rem 103
49304 DATA162,8,160,0,32,186,255,104
                                        :rem 31
49312 DATA162,240,160,194,32,189,255,169
                                        :rem 248
49320 DATA0,133,251,169,4,133,252, 169
                                        :rem 83
49328 DATA251,162,255,160,7,32,216, 255
                                        :rem 141
49336 DATA234,234,234,162,0,189,201,192
                                        :rem 188
49344 DATA32,210,255,232,224,18,208,245
                                        :rem 183
49352 DATA96,147,13, 69, 78, 84, 69, 82
                                        :rem 231
49360 DATA32,70,73, 76, 69, 32, 78, 65
                                        :rem 165
49368 DATA77,69,13,234,234,0, 173, 134
                                        :rem 50
49376 DATA2,162,0,157,0, 216, 232, 208
                                        :rem 237
49384 DATA250,172,229,192,192,219,240,7
                                        :rem 201
49392 DATA200,140,229,192,76,222,192,169
                                        :rem 248
49400 DATA216,141,229,192,96,234,234,0
                                        :rem 136
60679 DATA120,169,188,141,20, 3, 169, 2
                                        :rem 43
60687 DATA141,21,3,88,169, 0, 133, 253
                                        :rem 246
60695 DATA169,4,133,254,96,165,197, 201
                                        :rem 157
60703 DATA57,240,3,76,49,234, 160, 0
                                        :rem 189
60711 DATA177,253,69,249,145,253,200,177
                                        :rem 249
60719 DATA253,69,250,145,253,200,234,208
                                        :rem 238
60727 DATA239,230,254,165,254,201,8,208
                                        :rem 190
60735 DATA229,169,4,133,254,160, 0, 162
                                        :rem 86
60743 DATA0,232,208,253,200,208,250, 76
                                        :rem 125
60751 DATA 49, 234, 0, 0, 0, 0, 0, 0
                                        :rem 119
```

Canyon Cruiser

(Article on page 96.)

Program 1:

Canyon Cruiser—64 Version

```
100 RESTORE                          :rem 181
110 GOTO150                          :rem 97
120 WX=INT(255-W):IFWX<160THENWX=160
                                     :rem 216
130 POKE53250,WX:POKE53251,ABS(W-25)
                                     :rem 68
140 POKE53286,7:POKE53286,2:RETURN:rem 24
150 PRINT"{CLR}":POKE53280,6:POKE53281,6
                                     :rem 149
160 FORW=0TO10:READR$(W):NEXTW       :rem 138
170 DATA"STARTER"                    :rem 27
180 DATA"BEGINNER"                   :rem 65
190 DATA"LEARNER"                    :rem 1
200 DATA"FLYER"                      :rem 114
210 DATA"LEADER"                     :rem 158
220 DATA"ACE"                        :rem 187
230 DATA"LIEUTENANT"                 :rem 236
240 DATA"GENERAL"                    :rem 242
250 DATA"COMMODORE"                  :rem 154
260 DATA"FLEET COMMANDER"            :rem 252
270 DATA"HAN SOLO ?????????"         :rem 66
280 PRINT"{HOME}{5 DOWN}";SPC(15);"{WHT}6
    4-CANYON"                        :rem 142
290 PRINT                            :rem 40
300 PRINT"{HOME}{9 DOWN}";SPC(11);"{RVS}I
    NSTRUCTIONS (Y/N)"               :rem 81
310 FORW=1TO200:NEXTW                :rem 69
320 PRINT"{HOME}{9 DOWN}";SPC(11);"INSTRU
    CTIONS (Y/N)"                    :rem 65
330 FORW=1TO200:NEXTW                :rem 71
340 GETZ$:IFZ$=""THEN300             :rem 127
350 POKE53280,0:POKE53281,0          :rem 237
360 IFZ$="Y"THEN1540                 :rem 121
370 PRINT"{HOME}{5 DOWN}";SPC(15);"{4}6
    4-CANYON"                        :rem 32
380 PRINT                            :rem 40
390 PRINT"{HOME}{9 DOWN}";SPC(11);"
    {18 SPACES}"                     :rem 108
400 FORW=1TO50:P=INT(RND(TI)*1000)+1024:I
    FPEEK(P)=32THENPOKEP,46:POKEP+54272,1
                                     :rem 180
410 NEXTW                            :rem 43
420 POKE53271,0:POKE53277,0          :rem 240
430 FORW=12288TO12350:READN:POKEW,N:NEXTW
                                     :rem 99
440 FORW=12352TO12414:READN:POKEW,N:NEXTW
                                     :rem 93
450 POKE53276,PEEK(53276)OR2↑0+2↑1:REM SE
    T MULTI COLOR MODE               :rem 254
460 POKE53285,8:POKE53286,2:POKE53287,6:R
    EM SET COLORS                    :rem 179
470 POKE53248,0:POKE53249,0          :rem 248
480 POKE53264,PEEK(53264)AND(255-(2↑0+2↑1
    ))                               :rem 95
490 POKE2040,192:POKE2041,193        :rem 83
500 POKE53269,PEEK(53269)OR(2↑0+2↑1)
                                     :rem 22
510 POKE53275,0                      :rem 39
520 FORW=20TO160STEP2:POKE53248,W:POKE532
    49,W:GOSUB120:NEXTW              :rem 30
530 FORW=160TO180STEP2:POKE53249,PEEK(532
    49)+3:GOSUB120:NEXTW             :rem 232
540 PRINT"{HOME}{19 DOWN}{YEL}{RVS}";SPC(
    13);"GET READY..."               :rem 164
550 FORW=181TO255STEP2:L=PEEK(53249)+2:PO
    KE53249,(L)*((L>255)+1):GOSUB120:NEXT
                                     :rem 70
560 FORW=235TO255:POKE53251,W:GOSUB140:NE
    XTW                              :rem 250
570 GOTO740                          :rem 112
580 REM ** DATA FOR THE SHIP **      :rem 63
590 DATA 3, 0, 192, 15, 195, 240, 67, 0,
    {SPACE}193                       :rem 236
600 DATA 170, 170, 170, 171, 85, 234, 42,
     150, 168                        :rem 18
610 DATA 42, 150, 168, 26, 170, 164, 34,
    {SPACE}170, 136                  :rem 224
620 DATA 32, 170, 8, 32, 40, 8, 48, 40, 1
    2                                :rem 123
630 DATA 0, 40, 0, 0, 40, 0, 0, 60, 0
                                     :rem 97
640 DATA 0, 20, 0, 0, 40, 0, 0, 0, 0
                                     :rem 42
650 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0
                                     :rem 197
660 REM ** DATA FOR THE ASTEROID **
                                     :rem 101
670 DATA 0, 0, 0, 0, 0, 0, 1, 0, 0
                                     :rem 200
680 DATA 15, 92, 208, 0, 253, 252, 63, 20
    7, 253                           :rem 130
690 DATA 63, 223, 255, 12, 252, 252, 63,
    {SPACE}245, 60                   :rem 180
700 DATA 7, 127, 112, 15, 208, 245, 12, 1
    24, 127                          :rem 165
710 DATA 19, 255, 252, 55, 255, 220, 63,
    {SPACE}197, 244                  :rem 239
720 DATA 13, 63, 204, 15, 31, 76, 3, 61,
    {SPACE}240                       :rem 221
730 DATA 0, 3, 192, 0, 0, 0, 0, 0, 0
                                     :rem 51
740 GOTO910:REM ** RUN THE GAME **:rem 36
750 DATA"{6}{RVS} {OFF}{K}{3 SPACES}
    {RVS}{K} {OFF}"                  :rem 29
760 DATA"{5}{RVS} {OFF}{K}{3 SPACES}
    {RVS}{K} {OFF}"                  :rem 29
770 DATA"{4}{RVS} {OFF}{K}{4 SPACES}
    {RVS}{K} {OFF}"                  :rem 29
780 DATA"{3}{RVS} {OFF}{K}{5 SPACES}
    {RVS}{K} {OFF}"                  :rem 29
790 DATA"{2}{RVS} {OFF}{K}{6 SPACES}
    {RVS}{K} {OFF}"                  :rem 29
800 DATA"{1}{RVS} {OFF}{K}{7 SPACES}
    {RVS}{K} {OFF}"                  :rem 1
810 DATA"{YEL}{RVS} {OFF}{K}{8 SPACES}
    {RVS}{K} {OFF}"                  :rem 31
820 DATA"{BLU}{RVS} {OFF}{K}{9 SPACES}
    {RVS}{K} {OFF}"                  :rem 161
830 DATA"{7}{RVS} {L}{OFF}{3 SPACES}
    {RVS}{K} {OFF}"                  :rem 50
840 DATA"{5}{RVS} {L}{OFF}{3 SPACES}
    {RVS}{K} {OFF}"                  :rem 49
850 DATA"{4}{RVS} {L}{OFF}{4 SPACES}
    {RVS}{K} {OFF}"                  :rem 49
860 DATA"{3}{RVS} {L}{OFF}{5 SPACES}
    {RVS}{K} {OFF}"                  :rem 49
870 DATA"{2}{RVS} {L}{OFF}{6 SPACES}
    {RVS}{K} {OFF}"                  :rem 49
880 DATA"{1}{RVS} {OFF}{K}{7 SPACES}
    {RVS}{H} {OFF}"                  :rem 28
890 DATA"{YEL}{RVS} {OFF}{K}{8 SPACES}
    {RVS}{H} {OFF}"                  :rem 58
900 DATA"{BLU}{RVS} {OFF}{K}{9 SPACES}
    {RVS}{H} {OFF}"                  :rem 179
```

```
910 FORW=53248TO53264:POKEW,0:NEXTW
                                    :rem 174
920 PRINT"{CLR}{2 DOWN}";SPC(15);"{RED}
    {RVS}{A}+++++{S}"               :rem 75
930 POKE53271,2:POKE53277,2         :rem 250
940 POKE53275,253                   :rem 152
950 FORW=1TO10                      :rem 78
960 PRINTSPC(15);"{RED}{RVS}-{Q}{OFF}
    {3 SPACES}{RVS}{W}{RED}{RVS}-{OFF}"
                                    :rem 242
970 NEXTW                           :rem 54
980 PRINT"{RED}{RVS}{4 SPACES}{PUR}{OFF}T
    IME: 5{RED}{RVS}{2 SPACES}{A}{R}+
    {X}{OFF}{3 SPACES}{RVS}{Z}+{R}
    {S}{16 SPACES}"                 :rem 32
990 PRINT"{UP}";:FORW=1TO10:PRINTSPC(12);
    "{RED}{RVS} {Q}{OFF}{9 SPACES}{RVS}
    {W} {OFF}":NEXTW                :rem 155
1000 POKE53248,160:POKE53249,74     :rem 192
1010 PRINT"{HOME}"                  :rem 166
1020 TI$="000005"                   :rem 42
1030 PRINT"{HOME}{13 DOWN}{9 RIGHT}{PUR}"
     ;                              :rem 97
1040 A1=INT(10-VAL(TI$)):PRINTA1    :rem 108
1050 IFA1<>0THEN1030                :rem 100
1060 LDP=53248                      :rem 234
1070 FORSYP=74TO140:POKE53249,SYP:GOSUB14
     0:NEXTSYP                      :rem 231
1080 PRINT"{HOME}{23 DOWN}";        :rem 111
1090 FORW=0TO7:READC$(W):NEXTW      :rem 132
1100 FORW=0TO7:READBC$(W):NEXTW     :rem 190
1110 TL=12:C$=C$(7):BC$=BC$(7)      :rem 33
1120 GOSUB1440:TI$="000000"         :rem 169
1130 C$=C$(ABS(LEN(C$)-11))         :rem 110
1140 E=PEEK(53279)AND2↑0:IFE=1THEN1290
                                    :rem 210
1150 IFTI>800THENC$=C$((LEN(C$)-12)):BC$=
     BC$((LEN(C$)-11)):GOTO1460     :rem 155
1160 IFINT(RND(TI)*100)<20THENC$=BC$
                                    :rem 58
1170 IFINT(RND(TI)*100)=4THENGOSUB1440
                                    :rem 10
1180 TL=ABS(TL+((INT(RND(TI)*3)-1)))
                                    :rem 4
1190 TL=TL-(1*((TL<(40-LEN(C$)))+1))
                                    :rem 116
1200 TL=TL*((TL<0)+1)               :rem 116
1210 PRINT:PRINTSPC(TL);C$;         :rem 204
1220 IFAS=1THEN1470                 :rem 81
1230 P=PEEK(203):IFP=45THEN1280     :rem 23
1240 IFP=50THEN1270                 :rem 65
1250 P=PEEK(56320)AND12:IFP=8THEN1280
                                    :rem 137
1260 IFP<>4THEN1290                 :rem 81
1270 GOSUB140:POKELD,PEEK(LD)+2:POKELD,PE
     EK(LD)+3:GOTO1130              :rem 40
1280 GOSUB140:POKELD,PEEK(LD)-2:POKELD,PE
     EK(LD)-3:GOTO1130              :rem 45
1290 E=PEEK(53279)AND2↑0:IFE=0THEN1150
                                    :rem 210
1300 PRINT"{UP}{YEL}":PRINTSPC(TL);"{RVS}
     YOU CRASHED!"                  :rem 11
1310 R=R+INT(TI/400):IFR>10THENR=10
                                    :rem 70
1320 FORW=0TO150STEP2:GOSUB140:POKE2040,W
     :POKE2040,192:GOSUB140         :rem 95
1325 POKE2041,W:NEXT                :rem 189
1330 POKE53280,0:POKE53281,0        :rem 28
1340 PRINTSPC(6)"{RVS}YOUR RATING: ";R$(R
     )                              :rem 186
1350 FORW=1TO10:GETZ$:NEXTW         :rem 225
1360 GETZ$:IFZ$=":"ORZ$=";"ORZ$=""THEN136
     0                              :rem 158
1370 POKE53251,0                    :rem 86
1380 POKE2040,192:FORW=PEEK(53249)TO120ST
     EP-1:POKE53249,W:GOSUB140:NEXTW
                                    :rem 157
1390 POKE53248,ABS(PEEK(53248)-10):POKE53
     275,0                          :rem 42
1400 POKE53277,1:POKE53271,1:FORW=PEEK(53
     249)TO0STEP-1:POKE53249,W:GOSUB140
                                    :rem 2
1405 NEXT                           :rem 9
1410 RUN                            :rem 187
1420 POKE53287,PEEK(53287)+1        :rem 6
1430 TI$="000000":R=R+2:IFR>10THENR=10
                                    :rem 123
1440 IFAS=1THEN1210                 :rem 77
1450 AS=1:POKE53250,ABS(PEEK(53248)-20):P
     OKE53251,255:RETURN            :rem 221
1460 GOSUB1420:GOTO1210             :rem 73
1470 POKE53251,PEEK(53251)-5        :rem 255
1480 POKE53251,PEEK(53251)-5        :rem 0
1490 POKE53251,PEEK(53251)-5        :rem 1
1500 POKE53251,PEEK(53251)-5        :rem 249
1510 POKE53250,PEEK(53250)-(5-INT(RND(TI)
     *10))                          :rem 15
1520 IFPEEK(53251)<20THENPOKE53251,0:AS=0
                                    :rem 96
1530 GOTO1230                       :rem 200
1540 PRINT"{CLR}{DOWN} YOU ARE THE PILOT
     {SPACE}OF A NEW SPACESHIP."    :rem 11
1550 PRINT"{DOWN}YOU MUST TEST THE CRAFT
     {SPACE}TO ITS LIMITS."         :rem 23
1560 PRINT"{DOWN}YOU CAN FLY USING THE :
     {SPACE}AND ; KEYS,"            :rem 127
1570 PRINT"{DOWN}OR USE A JOYSTICK (PORT
     {SPACE}2)."                    :rem 228
1580 PRINT"{DOWN} THE SHIP WILL CHANGE CO
     LOR TO WARN OF"                :rem 146
1590 PRINT"{DOWN}CHANGES IN THE CANYON SI
     ZE."                           :rem 83
1610 PRINT"{DOWN} AT THE END OF YOUR FLIG
     HT YOU WILL BE"                :rem 85
1620 PRINT"{DOWN}RANKED. YOU MUST NOT HIT
      THE SIDES OF"                 :rem 152
1630 PRINT"{DOWN}THE CANYON, BUT ASTEROID
     S WILL FLY BY."                :rem 6
1640 PRINT"{DOWN} {WHT}GOOD LUCK!{5} (H
     IT A KEY TO RESTART)"          :rem 235
1650 WAIT198,1:GETZ$:PRINT"{CLR}";:RUN
                                    :rem 160
```

Program 2:

Canyon Cruiser—VIC Initialization

If using tape rather than disk, line 30 should read:

```
30 PRINT"LO";CHR$(34);"VIC-CANYON.PRG"
```

```
10 POKE52,28:POKE56,28:CLR          :rem 18
15 FORI=7168TO7679:POKEI,PEEK(25600+I):NE
   XTI                              :rem 176
20 FORX=0TO19                       :rem 27
21 READC                            :rem 194
22 FORD=0TO7:READE:POKE7168+C*8+D,E:NEXTD
                                    :rem 29
23 NEXTX                            :rem 252
30 PRINT"LO";CHR$(34);"VIC-CANYON.PRG";CH
   R$(34);",8"                      :rem 135
40 PRINT:PRINT:PRINT:PRINT:PRINT:PRINT"RU
   N"                               :rem 13
```

```
50 PRINT"{9 UP}";                          :rem 138
60 END                                     :rem 61
63000 DATA033,231,231,231,000,000,231,231
      ,231                                 :rem 173
63001 DATA034,231,231,231,231,231,231,231
      ,231                                 :rem 187
63002 DATA035,000,003,007,055,127,127,127
      ,063                                 :rem 200
63003 DATA036,000,192,248,248,252,124,238
      ,254                                 :rem 223
63004 DATA037,123,127,063,061,063,015,003
      ,000                                 :rem 195
63005 DATA038,254,238,124,252,248,248,192
      ,000                                 :rem 227
63006 DATA042,214,124,254,186,186,146,016
      ,016                                 :rem 223
63007 DATA043,000,000,000,000,000,000,000
      ,000                                 :rem 145
63008 DATA044,255,255,255,255,255,255,255
      ,255                                 :rem 243
63009 DATA047,224,224,224,224,224,224,224
      ,224                                 :rem 215
63010 DATA048,240,240,240,240,240,240,240
      ,240                                 :rem 192
63011 DATA049,231,231,231,007,007,231,231
      ,231                                 :rem 196
63012 DATA050,231,231,231,224,224,231,231
      ,231                                 :rem 191
63013 DATA051,231,231,231,224,224,255,255
      ,255                                 :rem 211
63014 DATA052,231,231,231,007,007,255,255
      ,255                                 :rem 211
63015 DATA053,255,255,255,000,000,231,231
      ,231                                 :rem 199
63016 DATA054,255,255,255,255,255,000,000
      ,000                                 :rem 207
63017 DATA055,255,255,255,224,224,231,231
      ,231                                 :rem 219
63018 DATA056,255,255,255,007,007,231,231
      ,231                                 :rem 219
63019 DATA057,007,007,007,007,007,007,007
      ,007                                 :rem 209
```

Program 3:

Canyon Cruiser—VIC Main Program

This program should be SAVEd as VIC-CANYON.PRG (see line 30 of Program 2).

```
90 POKE36869,240                           :rem 106
100 PRINT"{WHT}"                           :rem 103
110 GOTO150                                 :rem 97
120 WX=INT(255-W):IFWX<160THENWX=160
                                           :rem 216
150 PRINT"{CLR}"                           :rem 250
160 FORW=0TO10:READR$(W):NEXTW             :rem 138
170 DATA"STARTER"                           :rem 27
180 DATA"BEGINNER"                          :rem 65
190 DATA"LEARNER"                            :rem 1
200 DATA"FLYER"                            :rem 114
210 DATA"LEADER"                           :rem 158
220 DATA"ACE"                              :rem 187
230 DATA"LIEUTENANT"                       :rem 236
240 DATA"GENERAL"                          :rem 242
250 DATA"COMMODORE"                        :rem 154
260 DATA"FLEET COMMANDER"                  :rem 252
270 DATA"HAN SOLO ?????????"                :rem 66
350 POKE36879,8                             :rem 60
370 PRINT"{HOME}{4 DOWN}{6 SPACES}VIC-CAN
    YON"                                   :rem 153
380 PRINT                                   :rem 40
400 FORW=1TO50:P=INT(RND(TI)*506)+7680:IF
    PEEK(P)=32THENPOKEP,46:POKEP+30720,1
                                           :rem 148
410 NEXTW                                   :rem 43
540 PRINT"{BLU}{HOME}{18 DOWN}{RVS}";SPC(
    5);"GET READY..."                      :rem 229
580 TI$="000005"                             :rem 4
590 PRINT"{HOME}{11 DOWN}{RED}{8 SPACES}
    {RVS}TIME 5{2 LEFT}";                   :rem 68
600 A1=INT(10-VAL(TI$)):PRINTA1;"{3 LEFT}
    ";                                     :rem 206
610 IFA1<>0THEN600                           :rem 7
740 GOTO920:REM ** RUN THE GAME **:rem 37
750 DATA"{WHT},/{4 SPACES}9,{3 SPACES}"
                                           :rem 191
760 DATA"{CYN},/{4 SPACES}9,{3 SPACES}"
                                            :rem 90
770 DATA"{PUR},/{5 SPACES}9,{3 SPACES}"
                                            :rem 88
780 DATA"{GRN},/{6 SPACES}9,{3 SPACES}"
                                           :rem 219
790 DATA"{BLU},/{7 SPACES}9,{3 SPACES}"
                                           :rem 221
800 DATA"{YEL},/{8 SPACES}9,{3 SPACES}"
                                            :rem 84
810 DATA"{RED},/{9 SPACES}9,{3 SPACES}"
                                           :rem 211
820 DATA"{WHT},/{10 SPACES}9,{3 SPACES}"
                                           :rem 189
830 DATA"{WHT},/{4 SPACES}9,{3 SPACES}"
                                           :rem 190
840 DATA"{CYN},/{4 SPACES}9,{3 SPACES}"
                                            :rem 89
850 DATA"{PUR},/{5 SPACES}9,{3 SPACES}"
                                            :rem 87
860 DATA"{GRN},/{6 SPACES}9,{3 SPACES}"
                                           :rem 218
870 DATA"{BLU},/{7 SPACES}9,{3 SPACES}"
                                           :rem 220
880 DATA"{YEL},/{8 SPACES}9,{3 SPACES}"
                                            :rem 92
890 DATA"{RED},/{9 SPACES}9,{3 SPACES}"
                                           :rem 219
900 DATA"{WHT},/{10 SPACES}9,{3 SPACES}"
                                           :rem 188
920 PRINT"{CLR}":PRINT"{UP}";:POKE36869,2
    55                                      :rem 23
925 PRINTSPC(9);"{RED}7!!!8"                :rem 10
950 FORW=1TO10                              :rem 78
960 PRINTSPC(9);"{RED}";CHR$(34);"2 1";CH
    R$(34)                                   :rem 1
970 NEXTW                                   :rem 54
980 PRINT"{RED},,,,,,,75!4 3!58,,,,,,";
                                           :rem 135
990 FORW=1TO10:PRINTSPC(6);"{RED},2
    {7 SPACES}1,":NEXTW                    :rem 172
1000 POKE7713,42:POKE38433,4                :rem 34
1020 TI$="000005"                           :rem 42
1070 FOR SYP=55TO297STEP22:POKE38400+SYP-
     22,0:POKE7680+SYP-22,32                :rem 58
1075 POKE38400+SYP,4:POKE7680+SYP,42:NEXT
      SYP:SP=297:AB=32:CB=0                 :rem 53
1080 PRINT"{HOME}{20 DOWN}";                :rem 60
1090 FORW=0TO7:READC$(W):NEXTW              :rem 132
1100 FORW=0TO7:READBC$(W):NEXTW             :rem 190
1110 TL=6:C$=C$(7):BC$=BC$(7)               :rem 244
1120 GOSUB1440:TI$="000000"                 :rem 169
1130 C$=C$(ABS(LEN(C$)-11))                 :rem 110
1150 IFTI>800THENC$=C$((LEN(C$)-12)):BC$=
     BC$((LEN(C$)-11)):GOTO1460             :rem 155
1160 IFINT(RND(TI)*100)<20THENC$=BC$
                                            :rem 58
```

```
1170 IFINT(RND(TI)*100)=4THENGOSUB1440
                                    :rem 10
1180 TL=ABS(TL+((INT(RND(TI)*3)-1)))
                                    :rem 4
1190 TL=TL-(1*((TL<(22-LEN(C$)))+1))
                                   :rem 116
1200 TL=TL*((TL<0)+1)              :rem 116
1205 POKESP+7680,AB:POKESP+38400,CB
                                    :rem 48
1210 PRINT:PRINTSPC(TL);C$;        :rem 204
1220 IFAS=1THEN1470                 :rem 81
1230 P=PEEK(203):IF P=64 THEN 1285:rem 29
1235 IF P=45 THEN 1280              :rem 74
1240 IF P=22 THEN 1270              :rem 64
1260 GOTO 1285                     :rem 210
1270 SP=SP+1:IF PEEK(7680+SP)<>32 THEN 13
     00                             :rem 93
1271 AB=PEEK(7680+SP):CB=PEEK(38400+SP)
                                   :rem 227
1272 POKESP+38400,4:POKESP+7680,42:GOTO 1
     130                           :rem 254
1280 SP=SP-1:IF PEEK(7680+SP)<>32 THEN 13
     00                             :rem 96
1281 AB=PEEK(7680+SP):CB=PEEK(38400+SP)
                                   :rem 228
1282 POKESP+38400,4:POKESP+7680,42:GOTO 1
     130                           :rem 255
1285 IF PEEK(7680+SP)<>32AND PEEK(7680+SP
     )<>42THEN 1300                 :rem 22
1286 AB=PEEK(7680+SP):CB=PEEK(38400+SP)
                                   :rem 233
1287 POKESP+38400,4:POKESP+7680,42:GOTO 1
     130                            :rem 4
1300 POKE36869,240:PRINT"{CLR}":PRINTSPC(
     5);"{RVS}YOU CRASHED!"         :rem 63
1330 POKE36879,8                   :rem 107
1340 PRINTSPC(5);"{RVS}YOUR RATING:"
                                   :rem 160
1345 PRINTSPC(11-LEN(R$(R))/2);"{RVS}";R$
     (R)                           :rem 175
1350 FORW=1TO10:GETZ$:NEXTW        :rem 225
1360 GETZ$:IFZ$="":"ORZ$=";"ORZ$=""THEN136
     0                             :rem 158
1410 RUN                           :rem 187
1430 TI$="000000":R=R+1:IFR>10THENR=10
                                   :rem 122
1440 IFAS=1THEN1205                 :rem 81
1450 AS=1:RETURN                   :rem 230
1460 GOSUB1430:GOTO1205             :rem 78
1470 AS=0                          :rem 205
1475 FORQ=1TOLEN(C$)/2:PRINT"{LEFT}";:NEX
     TQ                            :rem 191
1480 PRINT"{2 UP}#${DOWN}{2 LEFT}%&"
                                   :rem 157
1530 GOTO1230                      :rem 200
```

Tetracrystals Of Veluria

(Article on page 86.)

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: Tetracrystals—VIC Version

```
10 POKE52,25:POKE56,25:CLR:GOSUB5000:CLR
                                   :rem 166
13 S7$="{RVS}{3 SPACES}{OFF}{2 SPACES}":C
   7$="BWRCPGBY"                   :rem 240
20 G1$="  -- Y VALUE --{2 SPACES}":G2$="AB
   CDEFGHIJKLMNOP":G3$="+{+3+{+3+
   {+3+{+3+{+3+{+3+{+3+{+3+":
   CJ$="XYPC"                      :rem 101
60 TG=1:GOSUB1100                  :rem 235
100 GOSUB8200:GOSUB6000:GOSUB8200:rem 223
160 PRINT"{CLR}SPEED":INPUTHF      :rem 196
180 PRINT"{DOWN}CHR?{DOWN}"        :rem 168
190 GOSUB8000                      :rem 226
195 IF(G$<"1")OR(G$>"4")THEN190    :rem 119
197 G=VAL(G$)*4-1:POKE7167,G        :rem 49
255 PRINT"{CLR}{3 SPACES}** X VALUE
    **":PRINT"{2 SPACES}";G2$:SYS6401
                                   :rem 188
260 H=0:FORG=1TO16:PRINTMID$(G1$,G,1);MID
    $(G2$,G,1);MID$(G3$,H+1,16):H=ABS(H-1
    ):NEXTG                        :rem 125
280 FORG=0TO3                       :rem 11
282 PRINT"{HOME}"MID$(CJ$,G+1,1)   :rem 172
285 GOSUB8000                      :rem 231
290 IFG$="{F1}"THEN280             :rem 100
295 IFG$="{F7}"THEN510             :rem 104
300 H=ASC(G$)-49:ONG+1GOTO325,345,365,385
                                   :rem 240
301 GOTO285                        :rem 108
305 NEXTG                           :rem 30
325 H=H-16:IF((H<0)OR(H>15))THEN285
                                   :rem 198
330 PRINT"{HOME}{DOWN}{2 RIGHT}";G2$;:POK
    E7704+H,81:XV=7726+H:GOTO305   :rem 148
345 H=H-16:IF((H<0)OR(H>15))THEN285
                                   :rem 200
350 FORJ=1TO16:POKE7703+J*22,J:NEXTJ:POKE
    7725+H*22,81:YV=H:GOTO305       :rem 4
365 IF((H<0)OR(H>1))THEN285        :rem 250
370 PV=H+49:SB=2↑H:GOTO305          :rem 76
385 IF((H<0)OR(H>7))THEN285         :rem 2
390 POKEXV+YV*22+30720,H:POKEXV+YV*16+302
    08,H:POKEXV+YV*22,PV:POKEXV+YV*16-558
    ,PV                            :rem 14
395 POKEXV+YV*16-302,((H*16)+SB):GOTO280
                                   :rem 31
510 POKE36879,SS:TG=1:CP=0:GOTO750:rem 52
600 SYS6406:SYS6431                :rem 108
650 SYS6577                        :rem 115
655 FORG=1TOHF:NEXTG                :rem 45
656 GETG$:IFG$<>""THENGOSUB7000     :rem 84
657 IFCPTHENGOSUB7310               :rem 62
660 SYS6475:SYS6542                :rem 123
690 GOSUB1000:TG=1                  :rem 35
695 GOSUB1100                      :rem 230
700 SYS6401:SYS6424                :rem 106
750 SYS6573                        :rem 112
755 FORG=1TOHF:NEXTG                :rem 46
756 GETG$:IFG$<>""THENGOSUB7000     :rem 85
757 IFCPTHENGOSUB7310               :rem 63
760 SYS6468:SYS6535                :rem 128
790 GOSUB1000:TG=0                  :rem 35
795 GOSUB1100:GOTO600              :rem 240
1000 J=INT(RND(1)*16):H=INT(RND(1)*3):POK
     E36874+H,PEEK(6993+J):POKE36878,(JOR
     8):RETURN                     :rem 70
1100 POKE36866,128*TG+16:POKE36867,32:POK
     E36864,10:POKE648,2*TG+28:Z=TG*2+156
                                   :rem 119
```

```
1120 FORH=217TO228:POKEH,Z:POKEH+12,Z+1:N
     EXTH:PRINT"{HOME}";:RETURN  :rem 149
5000 OPEN1,1,0,"CRYSTALS T3/JAN"  :rem 85
5020 FORG=0TO39:FORH=1TO16        :rem 82
5030 INPUT#1,Y:X=G*16+H+6400:POKEX,Y:PRIN
     TY;                          :rem 225
5060 NEXTH,G                      :rem 197
5075 INPUT#1,L:IFL<>99999THENSTOP :rem 78
5080 CLOSE1:RETURN                :rem 142
6000 SS=PEEK(36879):PRINT"{CLR}"SPC(66)S7
     $"F1 BDR":PRINTS7$"F3 SCN    :rem 166
6010 PRINTS7$"F5 REV":PRINTS7$"F7 FIN
     {DOWN}"                      :rem 210
6012 FORG=1TO8:PRINTS7$;G;MID$(C7$,G,1);S
     7$:NEXTG                     :rem 216
6020 FORG=0TO7:FORH=0TO7:POKE38583+G*22+H
     ,G:NEXTH:NEXTG               :rem 200
6030 POKE36879,SS                 :rem 219
6050 GOSUB8000:IFG$="{F1}"THENG=(SS+1)AND
     7:SS=(SSAND248)ORG           :rem 202
6052 IFG$="{F3}"THENSS=(SS+16)AND255
                                  :rem 216
6054 IFG$="{F5}"THENG=(SS+8)AND15:SS=(SSA
     ND240)ORG                    :rem 124
6056 IFG$="{F7}"THENRETURN        :rem 227
6060 GOTO6030                     :rem 206
7000 IF(G$<"{F1}")OR(G$>"{F6}")THENRETURN
                                  :rem 144
7010 G=ASC(G$)-132:ONGGOTO7100,7200,7210,
     7300,7400,7450,7600          :rem 142
7100 CP=0:RETURN                  :rem 226
7200 H=25:GOTO7220                :rem 243
7210 H=-25                        :rem 227
7220 GOSUB8000:G=ASC(G$)-48:IF(G<0)OR(G>9
     )THEN7220                    :rem 116
7240 HF=HF+G*H:RETURN              :rem 36
7300 CP=1                         :rem 203
7310 GOSUB8000:IFG$="{F7}"THENRETURN
                                   :rem 95
7320 IFG$="{F1}"THENCP=0:RETURN    :rem 21
7330 GOTO7000                     :rem 205
7400 PV=INT(RND(1)*8)*16+1:GOTO7500
                                  :rem 222
7450 GOSUB8000:G=ASC(G$)-49:IF(G<0)OR(G>7
     )THEN7450                    :rem 125
7460 PV=G*16+1                    :rem 232
7500 XV=INT(RND(1)*16):YV=INT(RND(1)*16):
     QV=XV+YV*16+7424             :rem 118
7510 POKEQV,PV:POKEQV+512,PV:RETURN
                                   :rem 52
7600 RUN13                         :rem 38
8000 GETG$:IFG$=""THEN8000        :rem 191
8010 RETURN                       :rem 169
8200 POKE36866,150:POKE36867,46:POKE36864
     ,5:POKE36879,27:RETURN       :rem 211
```

Program 2:

Tetracrystals—VIC DATA Maker

```
2 REM PREPARES{14 SPACES}DATAFILE FOR
  {10 SPACES}TETRACRYSTALS        :rem 142
5 DIMCS(40)                        :rem 90
8 FORG=1TO40:READH:CS(G)=H:NEXTG   :rem 38
10 PRINT"{CLR} CHECKSUM";          :rem 83
13 FORJ=1TO40                      :rem 10
15 TL=0                           :rem 115
18 FORK=1TO16                      :rem 19
20 READL:TL=TL+L                  :rem 248
23 NEXTK                          :rem 239
25 IFTL<>CS(J)THENPRINT:PRINT"TYPO IN LIN
   E";100+J*10:STOP                :rem 95
28 NEXTJ                          :rem 243
30 READL:IFL<>99999THENPRINT"INSUFFICIENT
   DATA":STOP                     :rem 152
32 PRINT" OK{2 DOWN}"             :rem 242
35 PRINT"GET DATA TAPE READY{3 SPACES}HIT
   A KEY TO CONTINUE              :rem 199
36 GETG$:IFG$=""THEN36              :rem 1
37 PRINT:PRINT                    :rem 190
38 CLR                             :rem 76
40 OPEN1,1,1,"CRYSTALS T3/JAN"    :rem 245
43 FORJ=1TO40:READL:NEXTJ         :rem 114
45 FORJ=1TO40                      :rem 15
50 FORK=1TO16                      :rem 15
53 READL:PRINT#1,L;CHR$(13);      :rem 143
55 NEXTK                          :rem 244
58 NEXTJ                          :rem 246
60 READL:IFL<>99999THENPRINT"CHECK DATA":
   GOTO65                         :rem 208
63 PRINT#1,L                      :rem 194
65 CLOSE1                          :rem 18
99 END                             :rem 73
101 DATA2175,2423,2474,1942,2180  :rem 98
102 DATA2431,2113,1935,2660,2567  :rem 98
103 DATA2317,1710,2352,2061,1928  :rem 93
104 DATA2461,2246,1533,2072,1693 :rem 100
105 DATA1626,1883,1556,1844,2604 :rem 113
106 DATA1715,2310,2659,1771,1908 :rem 109
107 DATA2411,1971,2168,2142,1770 :rem 101
108 DATA2510,1251,3302,0,0        :rem 28
110 DATA169,29,208,3,234,169,31,133,252,1
    69,0,133,251,234,160,0        :rem 68
120 DATA145,251,200,208,251,96,234,169,29
    ,162,31,208,5,234,169,31      :rem 179
130 DATA162,29,133,252,134,254,169,0,133,
    251,133,253,234,160,0,177     :rem 216
140 DATA253,10,9,240,133,1,177,253,9,15,3
    7,1,145,251,200,208           :rem 166
150 DATA238,96,234,169,28,162,29,208,5,23
    4,169,30,162,31,133,252       :rem 143
160 DATA134,254,169,0,133,251,133,253,234
    ,172,255,27,162,4,185,65      :rem 182
170 DATA27,149,0,136,202,208,247,169,32,1
    33,5,234,234,160,0,177        :rem 75
180 DATA253,162,0,9,16,24,232,106,176,3,1
    44,250,234,181,0,145          :rem 220
190 DATA251,200,208,235,96,234,169,148,16
    2,29,208,5,234,169,150,162    :rem 38
200 DATA31,133,252,134,254,169,0,133,251,
    133,253,234,160,0,177,253     :rem 208
210 DATA74,74,74,74,145,251,200,208,245,9
    6,234,234,169,29,208,2        :rem 94
220 DATA169,31,133,2,133,252,160,0,132,1,
    234,177,1,41,4,240            :rem 101
230 DATA5,32,204,25,164,254,200,208,242,9
    6,234,132,254,169,0,133       :rem 116
240 DATA253,152,41,240,208,4,169,17,133,2
    53,152,41,15,208,6,169        :rem 75
250 DATA34,5,253,133,253,200,152,41,15,20
    8,6,169,68,5,253,133          :rem 229
260 DATA253,165,254,9,15,168,200,208,6,16
    9,136,5,253,133,253,234       :rem 138
270 DATA76,11,26,5,253,133,253,96,234,234
    ,164,254,169,48,37,253        :rem 101
280 DATA208,19,152,56,233,17,168,177,1,41
    ,12,240,5,169,3,32            :rem 137
290 DATA4,26,164,254,234,169,80,37,253,20
    8,19,152,56,233,15,168        :rem 100
300 DATA177,1,41,12,240,5,169,5,32,4,26,1
    64,254,234,169,160            :rem 128
310 DATA37,253,208,19,152,24,105,15,168,1
    77,1,41,12,240,5,169          :rem 230
```

```
320 DATA10,32,4,26,164,254,234,169,192,37
    ,253,208,19,152,24,105          :rem 76
330 DATA17,168,177,1,41,12,240,5,169,12,3
    2,4,26,164,254,234              :rem 129
340 DATA169,1,37,253,208,50,165,254,56,23
    3,16,168,177,1,41,15            :rem 244
350 DATA234,234,234,234,208,34,165,254,56
    ,233,16,170,41,240,240,11       :rem 215
360 DATA138,56,233,16,168,177,1,41,12,208
    ,13,164,254,177,1,56            :rem 243
370 DATA233,3,134,251,164,251,145,1,169,2
    ,37,253,208,41,164,254          :rem 76
380 DATA136,177,1,41,15,234,234,234,234,2
    34,234,234,208,25,164,254       :rem 228
390 DATA136,152,41,15,240,7,136,177,1,41,
    12,208,10,164,254,177           :rem 22
400 DATA1,56,233,3,136,145,1,169,4,37,253
    ,208,44,164,254,200             :rem 176
410 DATA177,1,41,15,234,234,234,234,234,2
    34,234,208,28,169,1,133         :rem 121
420 DATA251,164,254,200,200,152,41,15,240
    ,6,177,1,41,12,208,9            :rem 208
430 DATA164,254,177,1,56,233,3,145,251,16
    9,8,37,253,208,44,165           :rem 45
440 DATA254,24,105,16,168,177,1,41,15,234
    ,234,234,234,208,28,169         :rem 134
450 DATA16,133,251,165,254,24,105,32,176,
    7,168,177,1,41,12,208           :rem 27
460 DATA10,164,254,177,1,56,233,3,145,251
    ,234,164,254,96,234,234         :rem 135
470 DATA87,81,90,0,219,102,160,42,65,90,8
    3,88,43,37,36,28                :rem 57
480 DATA163,175,175,187,195,201,207,209,2
    15,215,219,221,225,228,232,235
                                    :rem 235
490 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
                                    :rem 75
500 DATA0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0
                                    :rem 67
50000 DATA99999                     :rem 44
```

Program 3: Tetracrystals—64 version

```
10 GOTO9000:REM MACHINE LANGUAGE LOADER
                                    :rem 113
13 S7$="{RVS}{7 SPACES}{OFF}{4 SPACES}":C
   7$="BLKWHTREDCYNPURGBLUYEL":PS=53280
   :RV=49760                        :rem 17
20 G1$=" -- Y VALUE --{2 SPACES}":G2$="AB
   CDEFGHIJKLMNOP":G3$="+E+3+E+3+
   E+3+E+3+E+3+E+3+E+3+E+3+"
                                    :rem 181
30 CJ$="XVALUEYVALUEPLANT#COLOR"    :rem 97
60 GOSUB1100:REM NORMAL COLORS      :rem 97
100 GOSUB6000:REM SET UP SCREEN/BORDER
                                    :rem 51
110 GOSUB1100                       :rem 212
160 PRINT"{CLR}SPEED? (THE LOWER THE NUMB
    ER THE FASTER)":INPUTHF         :rem 14
180 PRINT"{DOWN}WHICH CHARACTER SET?
    {SHIFT-SPACE}(1-4){DOWN}"       :rem 58
190 GETG$:IFG$=""THEN190            :rem 99
195 IF(G$<"1")OR(G$>"4")THEN190     :rem 119
197 G=VAL(G$)*4-1:POKE49664,G       :rem 105
255 PRINT"{CLR}"SPC(130)"{3 SPACES}** X
     VALUE **":PRINTSPC(52);G2$:SYS49670
                                    :rem 100
260 H=0:FORG=1TO16:PRINTSPC(9)MID$(G1$,G,
    1)" "MID$(G2$,G,1);             :rem 90
270 PRINTMID$(G3$,H+1,16):H=ABS(H-1):NEXT
    G                               :rem 194
280 FORG=0TO3                       :rem 11
282 PRINT"{HOME}"MID$(CJ$,G*6+1,6):PRINT"
    *-F1 TO CANCEL-*-F7 TO BEGIN GAME-*"
                                    :rem 173
285 GETG$:IFG$=""THEN285            :rem 109
290 IFG$="{F1}"THEN280              :rem 100
295 IFG$="{F7}"THEN510              :rem 104
300 H=ASC(G$)-49:ONG+1GOTO325,345,365,385
                                    :rem 240
301 GOTO285                         :rem 108
305 NEXTG                           :rem 30
325 H=H-16:IF((H<0)OR(H>15))THEN285
                                    :rem 198
330 PRINT"{HOME}{5 DOWN}"SPC(12);G2$;:POK
    E1236+H,81:XV=1276+H:GOTO305    :rem 44
345 H=H-16:IF((H<0)OR(H>15))THEN285
                                    :rem 200
350 FORJ=1TO16:POKE1235+J*40,J:NEXTJ:POKE
    1275+H*40,81:YV=H:GOTO305       :rem 248
365 IF((H<0)OR(H>1))THEN285         :rem 250
370 PV=H+49:SB=2↑H:GOTO305          :rem 76
385 IF((H<0)OR(H>7))THEN285         :rem 2
390 POKEXV+YV*40+54272,H:POKEXV+YV*40,PV
                                    :rem 213
395 POKEXV+YV*16+48132,((H*16)+SB):GOTO28
    0                               :rem 138
510 POKEPS,RB:POKEPS+1,SC:CP=0      :rem 140
520 PRINT"{CLR}{DOWN}"SPC(7);"T E T R A C
     R Y S T A L S":GOTO700         :rem 248
600 SYS49670:SYS49695               :rem 233
650 SYS49841                        :rem 164
655 FORG=1TOHF:NEXTG                :rem 45
656 GETG$:IFG$<>""THENGOSUB7000     :rem 84
657 IFCPTHENGOSUB7310               :rem 62
660 SYS49806:SYS49739               :rem 239
690 GOSUB1000:REM MUSIC             :rem 127
700 SYS49665:SYS49688               :rem 240
750 SYS49837                        :rem 170
755 FORG=1TOHF:NEXTG                :rem 46
756 GETG$:IFG$<>""THENGOSUB7000     :rem 85
757 IFCPTHENGOSUB7310               :rem 63
760 SYS49799:SYS49732               :rem 244
790 GOSUB1000:REM MUSIC             :rem 128
795 GOTO600                         :rem 116
1000 REM MUSIC                      :rem 38
1050 VM=(INT(RND(1)*3))*16:VM=VM+(INT(RND
     (1)*16))                       :rem 243
1060 POKE50688,VM                   :rem 208
1070 SYS50689                       :rem 211
1080 RETURN                         :rem 169
1099 RETURN                         :rem 179
1100 POKE53280,254:POKE53281,246:RETURN
                                    :rem 8
6000 RB=PEEK(PS):SC=PEEK(PS+1):PRINT"
     {CLR}"SPC(131)S7$"F1 BORDER":rem 195
6005 PRINTSPC(11)S7$"F3 SCREEN      :rem 250
6008 PRINTSPC(11)S7$"F5 REVERSE     :rem 91
6010 PRINTSPC(11)S7$"F7 CONTINUE{DOWN}
                                    :rem 176
6012 FORG=1TO8:PRINTSPC(11)S7$;G;MID$(C7$
     ,G*3-2,3):NEXTG                :rem 70
6020 FORG=0TO7:FORH=0TO7:POKE55627+G*40+H
     ,G:NEXTH:NEXTG                 :rem 198
6030 POKEPS,RB:POKEPS+1,SC          :rem 133
6050 GETG$:IFG$=""THEN6050          :rem 197
6051 IFG$="{F1}"THENRB=(RB+1)AND15:rem 70
6052 IFG$="{F3}"THENSC=(SC+1)AND15:rem 76
6054 IFG$="{F5}"THENPOKERV,((PEEK(RV)+128
     )AND255)                       :rem 248
6055 IFG$="{F5}"THENFORX=1344TO1663:POKEX
     ,((PEEK(X)+128)AND255):NEXTX   :rem 31
```

```
6056 IFG$="{F7}"THENRETURN          :rem 227
6060 GOTO6030                       :rem 206
7000 IF(G$<"{F1}")OR(G$>"{F6}")THENRETURN
                                    :rem 144
7010 G=ASC(G$)-132:ONGGOTO7100,7200,7210,
     7300,7400,7450,7600            :rem 142
7100 CP=0:RETURN                    :rem 226
7200 H=25:GOTO7220                  :rem 243
7210 H=-25                          :rem 227
7220 GOSUB8000:G=ASC(G$)-48:IF(G<0)OR(G>9
     )THEN7220                      :rem 116
7240 HF=HF+G*H:RETURN                :rem 36
7300 CP=1:RETURN                    :rem 229
7310 GOSUB8000:IFG$="{F7}"THENRETURN
                                     :rem 95
7320 IFG$="{F1}"THENCP=0:RETURN      :rem 21
7330 GOTO7000                       :rem 205
7400 PV=INT(RND(1)*8)*16+1:GOTO7500
                                    :rem 222
7450 GOSUB8000:G=ASC(G$)-49:IF(G<0)OR(G>7
     )THEN7450                      :rem 125
7460 PV=G*16+1                      :rem 232
7500 XV=INT(RND(1)*16):YV=INT(RND(1)*16):
     QV=XV+YV*16+49152              :rem 170
7510 POKEQV,PV:POKEQV+256,PV:RETURN
                                     :rem 57
7600 RUN13                           :rem 38
8000 GETG$:IFG$=""THEN8000          :rem 191
8010 RETURN                         :rem 169
9000 DIMCS(54)                      :rem 243
9010 FORG=1TO54:READCS(G):NEXTG     :rem 182
9020 PRINT"{CLR}"SPC(6);"T E T R A C R Y
     {SPACE}S T A L S{2 DOWN}":PRINT"-- C
     HECKSUM:";                     :rem 95
9030 FORJ=1TO54                     :rem 119
9040 TL=0                           :rem 218
9050 FORK=1TO16                     :rem 120
9060 READL:TL=TL+L                  :rem 101
9070 NEXTK                          :rem 90
9080 IFTL<>CS(J)THENPRINT:PRINT"TYPO IN L
     INE";9340+J*10:STOP             :rem 8
9090 NEXTJ                          :rem 91
9100 READL:IFL<>99999THENSTOP       :rem 83
9110 PRINT" OK{DOWN}":PRINT"-- LOADING MA
     CHINE LANGUAGE."               :rem 17
9120 CLR:M=49664                    :rem 126
9130 POKEM,3                        :rem 168
9150 FORJ=1TO54:READK:NEXTJ         :rem 222
9155 FORJ=1TO40                     :rem 122
9160 FORK=1TO16:M=M+1               :rem 231
9165 READL:POKEM,L                  :rem 107
9170 NEXTK                           :rem 91
9175 NEXTJ                           :rem 95
9179 M=50688                        :rem 111
9180 FORJ=41TO46                    :rem 178
9185 FORK=1TO16:M=M+1               :rem 238
9190 READL:POKEM,L                  :rem 105
9195 NEXTK                           :rem 98
9200 NEXTJ                           :rem 84
9204 M=50943                         :rem 94
9205 FORJ=47TO54                    :rem 181
9210 FORK=1TO16:M=M+1               :rem 227
9215 READL:POKEM,L                  :rem 103
9220 NEXTK                           :rem 87
9225 NEXTJ                           :rem 91
9230 READL:IFL<>99999THENSTOP       :rem 87
9240 RUN13                           :rem 40
9270 DATA2500,2910,2637,1944,2015:rem 155
9272 DATA2122,2553,2150,2153,2209:rem 148
9274 DATA1612,1875,2521,2061,1928:rem 166
9276 DATA2461,2415,1534,2241,1863:rem 163
9278 DATA1627,2052,1726,1845,2604:rem 171
9280 DATA1717,2311,2660,1772,1910:rem 158
9282 DATA2412,1972,2169,2143,1771:rem 167
9284 DATA2511,1251,2546,2115,1718:rem 158
9286 DATA1762,2481,2247,1939,2497:rem 189
9288 DATA1436,546,282,778,929      :rem 248
9290 DATA1246,1495,1077,1002       :rem 164
9350 DATA169,192,208,3,234,169,193,133,25
     2,169,0,133,251,234,160,0     :rem 237
9360 DATA145,251,200,208,251,96,234,169,1
     92,162,193,208,5,234,169,193:rem 149
9370 DATA162,192,133,252,134,254,169,0,13
     3,251,133,253,234,160,0,177   :rem 72
9380 DATA253,10,9,240,133,2,177,253,9,15,
     37,2,145,251,200,208          :rem 231
9390 DATA238,96,234,169,4,32,81,196,240,5
     ,169,4,32,85,196,234           :rem 20
9400 DATA172,0,194,162,4,185,65,196,149,1
     ,136,202,208,247,169,32       :rem 146
9410 DATA133,6,234,234,234,234,234,234,23
     4,160,15,177,253,162,0,9      :rem 176
9420 DATA16,24,232,106,176,3,144,250,234,
     181,1,145,251,136,16,235      :rem 170
9430 DATA32,101,196,144,228,96,169,216,32
     ,81,196,240,5,169,216,32      :rem 201
9440 DATA85,196,234,160,15,177,253,74,74,
     74,74,145,251,136,16,245      :rem 213
9450 DATA32,101,196,144,238,96,234,0,0,0,
     0,0,169,192,208,2              :rem 77
9460 DATA169,193,133,3,133,252,160,0,132,
     2,234,177,2,41,4,240          :rem 224
9470 DATA5,32,204,194,164,254,200,208,242
     ,96,234,132,254,169,0,133     :rem 234
9480 DATA253,152,41,240,208,4,169,17,133,
     253,152,41,15,208,6,169       :rem 138
9490 DATA34,5,253,133,253,200,152,41,15,2
     08,6,169,68,5,253,133          :rem 36
9500 DATA253,165,254,9,15,168,200,208,6,1
     69,136,5,253,133,253,234      :rem 192
9510 DATA76,11,195,5,253,133,253,96,234,2
     34,164,254,169,48,37,253      :rem 210
9520 DATA208,19,152,56,233,17,168,177,2,4
     1,12,240,5,169,3,32           :rem 192
9530 DATA4,195,164,254,234,169,80,37,253,
     208,19,152,56,233,15,168      :rem 209
9540 DATA177,2,41,12,240,5,169,5,32,4,195
     ,164,254,234,169,160          :rem 247
9550 DATA37,253,208,19,152,24,105,15,168,
     177,2,41,12,240,5,169          :rem 38
9560 DATA10,32,4,195,164,254,234,169,192,
     37,253,208,19,152,24,105      :rem 194
9570 DATA17,168,177,2,41,12,240,5,169,12,
     32,4,195,164,254,234          :rem 248
9580 DATA169,1,37,253,208,50,165,254,56,2
     33,16,168,177,2,41,15          :rem 52
9590 DATA234,234,234,234,208,34,165,254,5
     6,233,16,170,41,240,240,11     :rem 22
9600 DATA138,56,233,16,168,177,2,41,12,20
     8,13,164,254,177,2,56          :rem 43
9610 DATA233,3,134,251,164,251,145,2,169,
     2,37,253,208,41,164,254       :rem 131
9620 DATA136,177,2,41,15,234,234,234,234,
     234,234,234,208,25,164,254     :rem 27
9630 DATA136,152,41,15,240,7,136,177,2,41
     ,12,208,10,164,254,177         :rem 77
9640 DATA2,56,233,3,136,145,2,169,4,37,25
     3,208,44,164,254,200           :rem 241
9650 DATA177,2,41,15,234,234,234,234,234,
     234,234,234,208,28,169,1,133   :rem 185
```

```
9660 DATA251,164,254,200,200,152,41,15,24
     0,6,177,2,41,12,208,9          :rem 16
9670 DATA164,254,177,2,56,233,3,145,251,1
     69,8,37,253,208,44,165         :rem 109
9680 DATA254,24,105,16,168,177,2,41,15,23
     4,234,234,234,208,28,169       :rem 198
9690 DATA16,133,251,165,254,24,105,32,176
     ,7,168,177,2,41,12,208         :rem 91
9700 DATA10,164,254,177,2,56,233,3,145,25
     1,234,164,254,96,234,234       :rem 190
9710 DATA87,81,90,0,219,102,160,42,65,90,
     83,88,43,37,36,28              :rem 111
9720 DATA162,192,208,2,162,193,133,252,13
     4,254,169,132,133,251,169,0    :rem 79
9730 DATA133,253,96,234,24,169,16,101,253
     ,144,1,96,133,253,169,40       :rem 193
9740 DATA101,251,144,2,230,252,133,251,24
     ,96,234,0,0,0,0,0              :rem 50
9749 REM -- MACHINE LANGUAGE MUSIC --
     {18 SPACES}COULD BE OPTIONAL:rem 147
9750 DATA173,0,198,10,170,74,41,240,74,72
     ,74,74,74,133,251,104          :rem 40
9760 DATA56,229,251,168,189,32,199,153,0,
     199,232,200,189,32,199,153     :rem 69
9770 DATA 0,199,136,132,252,162,7,185,0,1
     99,153,0,212,200,202,208       :rem 128
9780 DATA246,169,1,141,3,212,169,10,141,2
     4,212,164,252,169,1,25         :rem 79
9790 DATA4,199,153,4,212,162,5,160,0,234,
     234,234,200,208,250,238        :rem 128
9800 DATA3,212,238,24,212,202,208,241,96,
     0,0,0,0,0,0,0                  :rem 99
9809 REM -- MUSIC DATA STATEMENTS --
                                    :rem 21
9810 DATA0,0,0,5,64,140,48,0,0,0,0,32,152
     ,105,0,0                       :rem 99
9820 DATA0,0,16,57,194,0,0,0,15,0,0,0,0,0
     ,0,0                           :rem 152
9830 DATA134,3,180,4,71,5,71,6,12,7,12,7,
     104,9,143,10                   :rem 80
9840 DATA143,10,143,12,24,14,24,14,195,16
     ,209,18,31,21,30,25            :rem 168
9850 DATA49,28,165,31,135,33,162,37,62,42
     ,193,44,60,50,99,56            :rem 216
9860 DATA75,63,15,67,15,67,69,75,125,84,2
     14,94,121,100,199,112          :rem 63
9870 DATA12,7,97,8,143,10,143,10,218,11,1
     43,12,24,14,210,15             :rem 118
9880 DATA195,16,209,18,31,21,30,25,49,28,
     49,28,162,37,62,42             :rem 161
10000 DATA99999                     :rem 40
```

Hardhat Climber

(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.

Program 1:

Hardhat Climber—VIC Version

```
0 POKE51,192:POKE52,29:POKE55,192:POKE56,
  29:POKE36869,255:POKE36878,15:POKE36879
  ,25                               :rem 83
1 D=37154:P1=37151:P2=37152:DO(0)=-1:DO(1
  )=1:DI=DO(INT(RND(1)*2))          :rem 54
2 A$=">>>>>>>>>>>>>>>>>>>>{LEFT}{INST}>"
  :DIMB(11):G=30720:E2=0            :rem 126
3 SC=1:CH=2:E1=0:D(0)=4:D(1)=2:D(4)=7:Z=5
  7:E3=1:Q=10000:J=56               :rem 32
10 GOSUB100:POKE36869,255:E4=0:E5=2
                                    :rem 246
15 GOSUB70:H=0:Y=0                  :rem 60
16 S=8143+INT(RND(1)*20):IFPEEK(S+22)=620
   RPEEK(S)=59THEN16                :rem 145
17 T=PEEK(S):POKES,58:POKES+G,0     :rem 161
19 V=7712+B(Y):W=62:DO=DO(INT(RND(1)*2))
                                    :rem 7
20 SYS828:ONPEEK(1)GOTO35,26,28,31,33
                                    :rem 110
21 FORN=1TO23:NEXT:GOTO41           :rem 95
26 IFPEEK(S+22)=ZTHENPOKES,T:POKES+G,D(T-
   J):S=S+22:GOTO40                 :rem 84
27 GOTO41                           :rem 7
28 DI=-1:IFPEEK(S+21)<62THENPOKES,T:POKES
   +G,D(T-J):S=S-1:GOTO40           :rem 147
29 IFT<>ZTHENPOKES,T:POKES+G,D(T-J):S=S+D
   I:T=PEEK(S):GOTO55               :rem 83
30 GOTO41                           :rem 1
31 IFT=ZTHENPOKES,T:POKES+G,D(T-J):S=S-22
   :GOTO40                          :rem 78
32 GOTO41                           :rem 3
33 DI=1:IFPEEK(S+23)<62THENPOKES,T:POKES+
   G,D(T-J):S=S+1:GOTO40            :rem 98
34 GOTO29                           :rem 11
35 POKE36876,240:POKES,T:POKES+G,D(T-J):S
   =S-22+DI:T=PEEK(S):POKES,58:IFT=60THEN
   55                               :rem 5
36 IFPEEK(S+22)=60THENSS=SS+1000:PRINT"
   {HOME}{RVS}"TAB(8-LEN(STR$(SS)))SS
                                    :rem 62
37 FORN=1TO5:NEXT:POKES,T:POKES+G,D(T-J):
   S=S+22+DI:T=PEEK(S):POKES,58:IFPEEK(S+
   22)>61THEN55
38 POKES+G,0:POKE36876,0:GOTO41     :rem 102
40 POKE36876,200:POKE36876,0:T=PEEK(S):PO
   KES,58:POKES+G,0                 :rem 165
41 IFT=61THENSS=SS+150:PRINT"{HOME}{RVS}"
   TAB(8-LEN(STR$(SS)))SS:H=H+1:T=62:IFH=
   16THEN64                         :rem 240
42 IFT=60THEN55                     :rem 133
43 GOSUB98                          :rem 88
45 POKEV,W:POKEV+G,D(W-J):V=V+DO:W=PEEK(V
   ):POKEV,60:POKEV+G,7             :rem 145
46 IFDO=22ANDPEEK(V+22)=56THENDO=DO(INT(R
   ND(1)*2)):GOTO48                 :rem 195
47 IFW=63THENDO=22                  :rem 90
48 IFW=58THEN55                     :rem 149
49 IFV<8164THEN20                   :rem 242
50 Y=Y+1:IFY=12THEN55               :rem 11
51 POKEV,62:GOTO19                  :rem 92
55 POKE36876,0:SO=250:IFT=60THENT=W
                                    :rem 138
56 POKE36874,SO:IFPEEK(S+22)<>56ANDS<8164
   THENPOKES,T:POKES+G,D(T-J):S=S+22:T=PE
   EK(S):POKES,58:POKES+G,0         :rem 185
57 FORN=1TO17:NEXT:SO=SO-5:IFSO>150THEN56
                                    :rem 137
58 POKE36874,0:CH=CH-1:IFCH=-1THENPOKED,2
   55:POKE36869,240:PRINT"{CLR}{BLK}"SS:E
   ND                               :rem 104
59 PRINT"{HOME}{RVS}"TAB(14)CH::POKEV,W:P
   OKEV+G,D(W-J):Y=Y+1:IFW=58THENPOKEV,T:
   POKEV+G,D(T-J)                   :rem 248
60 IFY>10THEN15                     :rem 130
61 IFS>8163THENPOKES,T:GOTO16       :rem 100
```

```
62 POKES,58:POKES+G,0:GOTO19          :rem 234
64 IFY=11THEN67                       :rem 141
65 FORN=Y+1TO11:POKE7712+B(N),62:SS=SS+10
   0:PRINT"{HOME}{RVS}"TAB(8-LEN(STR$(SS)
   ))SS:GOSUB98                       :rem 194
66 POKE36877,250:FORM=240TO250:POKE36876,
   M:NEXT:POKE36876,0:POKE36877,0:NEXT
                                      :rem 113
67 E2=E2+.05:SC=SC+1:E1=E1+1:IFE1>8THENE1
   =8                                 :rem 227
68 GOTO15                             :rem 13
69 GOTO69                             :rem 23
70 PRINT"{CLR}{PUR}";:FORN=1TO21:PRINTA$:
   NEXT:PRINTA$"{HOME}":B$=">888888888888
   88888888"                          :rem 108
71 PRINT"{2 DOWN}"TAB(6)"?>>>>>>>>>?{RED}"
   :PRINTTAB(6)"9{PUR}88888888{RED}9":PRI
   NTTAB(6)"9>>>>>>>>>9":PRINTTAB(6)"9>>>>
   >>>>9{PUR}"                        :rem 12
72 FORN=1TO3:PRINTB$"{4 DOWN}":NEXT:PRINT
   B$"{HOME}";:POKE8185,62            :rem 162
73 PRINT"{RVS}"TAB(8-LEN(STR$(SS)))SS;TAB
   (14)CH;TAB(17)SC:POKE7697,163:FORN=0TO
   11:POKE7712+B(N),60                :rem 61
74 POKE7712+B(N)+G,7:NEXT:FORN=7834TO8164
   STEP110:IFN=8164THEN80             :rem 73
75 FORO=1TO3                          :rem 230
76 R=N+1+INT(RND(1)*20):IFPEEK(R)<>56THEN
   76                                 :rem 17
77 FORM=RTOR+88STEP22:POKEM,57:POKEM+G,2:
   NEXT:IFO>1ANDRND(1)<E2THENPOKER+(INT(R
   ND(1)*2)+2)*22,63                  :rem 203
78 IFRND(1)<.5ANDPEEK(R-22)=62THENPOKER-2
   2,63                               :rem 170
79 NEXT                               :rem 175
80 FORO=1TOE1                         :rem 37
81 R=N+3+INT(RND(1)*16):IFPEEK(R)<>56ORPE
   EK(R-22)<>62ORPEEK(R+1)=62ORPEEK(R-1)=
   62THEN85                           :rem 198
84 POKER,62:POKER-22,63               :rem 98
85 NEXT                               :rem 172
86 FORO=1TO4                          :rem 233
87 R=N-21+INT(RND(1)*20):IFPEEK(R)<>62ORP
   EEK(R+22)=62THEN87                 :rem 227
88 POKER,61:POKER+G,0:NEXT:NEXT       :rem 255
89 POKE7710,63:POKE7715,63:POKE7731,63:PO
   KE7738,63                          :rem 125
90 FORN=7812TO8142STEP110:POKEN,63:NEXT:F
   ORN=7833TO8163STEP110:POKEN,63:NEXT:RE
   TURN                               :rem 85
98 IFSS>=Q*E3THENCH=CH+1:E3=E3+1:PRINT"
   {HOME}{RVS}"TAB(14)CH              :rem 253
99 RETURN                             :rem 82
100 DATA,1,21,22,23,24,42,43,44,45,46,47
                                      :rem 197
101 DATA255,255,153,102,102,153,255,255,1
    95,255,255,195,195,255,255,195,60,60,
    25,255,188                        :rem 64
102 DATA60,36,231,3,4,24,24,60,126,126,60
    ,60,66,165,153,153,165,66,60,,24,36,1
    26,126,126                        :rem 11
103 DATA126,,,,,,,,,,,,,,,,           :rem 51
106 DATA169,,133,1,169,255,141,34,145,169
    ,32,44,31,145,208,5,169,1,133,1,96,16
    9,8,44                            :rem 98
107 DATA31,145,208,5,169,2,133,1,96,169,1
    6,44,31,145,208,5,169,3,133,1,96,169,
    4,44,31                           :rem 142
108 DATA145,208,3,133,1,96,169,127,141,34
    ,145,169,128,44,32,145,208,4,169,5,13
    3,1,96                            :rem 102
109 FORN=0TO11:READB(N):NEXT:FORN=7616TO7
    679:READM:POKEN,M:NEXT            :rem 61
110 FORN=828TO899:READM:POKEN,M:NEXT:RETU
    RN                                :rem 89
```

Program 2:

Hardhat Climber—64 Version

```
1 PRINT"{CLR}{7 DOWN}{12 RIGHT}HARDHAT CL
  IMBER":POKE53280,1:POKE53281,1:GOSUB780
                                      :rem 76
2 POKE52,48:POKE56,48:CLR             :rem 231
5 PRINT"{CLR}"                        :rem 153
6 PRINT"{7 DOWN}{12 RIGHT}HARDHAT CLIMBER
  "                                   :rem 212
10 POKE53280,1:POKE53281,1            :rem 184
20 DO(0)=-1:DO(1)=1:DI=DO(INT(RND(1)*2))
                                      :rem 180
30 A$=">>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
   >>>>>{LEFT}{INST}>":DIMB(11):G=54272:E
   2=0                                :rem 19
40 SC=1:CH=2:E1=0:D(0)=4:D(1)=2:D(4)=7:Z=
   57:E3=1:Q=10000:J=56               :rem 81
44 FORI=GTOG+24:POKEI,0:NEXT          :rem 30
45 POKEG+24,15:POKEG+5,17:POKEG+6,240:POK
   EG,100                             :rem 230
50 GOSUB710:E4=0:E5=2                 :rem 192
60 GOSUB500:H=0:Y=0                   :rem 106
70 S=1905+INT(RND(1)*38):IFPEEK(S+40)=620
   RPEEK(S)=59THEN70                  :rem 153
80 T=PEEK(S):POKES,58:POKES+G,0       :rem 161
90 V=1123+B(Y):W=62:DO=DO(INT(RND(1)*2))
                                      :rem 252
100 JV=255-PEEK(56321):GETQ$          :rem 61
101 IFJV=1THENJV=4:GOTO109            :rem 124
102 IFJV=4THENJV=3:GOTO109            :rem 127
103 IFJV=8THENJV=5:GOTO109            :rem 134
104 IFJV=2THEN109                     :rem 252
105 IFJVAND16=16THENJV=1:GOTO109      :rem 237
108 JV=0                              :rem 166
109 ONJVGOTO210,120,140,170,190       :rem 173
110 FORN=1TO23:NEXT:GOTO260           :rem 193
120 IFPEEK(S+40)=ZTHENPOKES,T:POKES+G,D(T
    -J):S=S+40:GOTO250                :rem 178
130 GOTO260                           :rem 101
140 DI=-1:IFPEEK(S+39)<62THENPOKES,T:POKE
    S+G,D(T-J):S=S-1:GOTO250          :rem 250
150 IFT<>ZTHENPOKES,T:POKES+G,D(T-J):S=S+
    DI:T=PEEK(S):GOTO360              :rem 173
160 GOTO260                           :rem 104
170 IFT=ZTHENPOKES,T:POKES+G,D(T-J):S=S-4
    0:GOTO250                         :rem 181
180 GOTO260                           :rem 106
190 DI=1:IFPEEK(S+41)<62THENPOKES,T:POKES
    +G,D(T-J):S=S+1:GOTO250           :rem 201
200 GOTO150                           :rem 97
210 POKEG+1,17:POKEG+4,17             :rem 156
215 POKES,T:POKES+G,D(T-J):S=S-40+DI:T=PE
    EK(S):POKES,58:IFT=60THEN360      :rem 43
220 IFPEEK(S+40)=60THENSS=SS+1000:PRINT"
    {HOME}{RVS}"TAB(8-LEN(STR$(SS)))SS
                                      :rem 105
230 FORN=1TO5:NEXT:POKES,T:POKES+G,D(T-J)
    :S=S+40+DI:T=PEEK(S):POKES,58:rem 204
235 IFS>1943ORPEEK(S+40)>61THEN360
                                      :rem 241
240 POKEG+4,16:POKES+G,0:GOTO260      :rem 147
250 POKEG+1,9:POKEG+4,17:POKEG+4,16:T=PEE
    K(S):POKES,58:POKES+G,0           :rem 134
```

```
260 IFT<>61THEN270                          :rem 36
265 SS=SS+150:PRINT"{HOME}{RVS}"TAB(8-LEN
    (STR$(SS)))SS:H=H+1:T=62:IFH=16THEN44
    0                                       :rem 160
270 IFT=60THEN360                           :rem 231
280 GOSUB690                                :rem 185
290 POKEV,W:POKEV+G,D(W-J):V=V+DO:W=PEEK(
    V):POKEV,60:POKEV+G,7                   :rem 195
300 IFDO=40ANDPEEK(V+40)=56THENDO=DO(INT(
    RND(1)*2)):GOTO320                      :rem 21
310 IFW=63THENDO=40                         :rem 131
320 IFW=58THEN360                           :rem 237
330 IFV<1944THEN100                         :rem 73
340 Y=Y+1:IFY=12THEN60                      :rem 57
350 POKEV,62:GOTO90                         :rem 141
360 POKEG+4,16:SO=250:IFT=60THENT=W
                                            :rem 136
361 FORQQ=1TO30:NEXT                        :rem 16
365 T1=T                                    :rem 180
370 POKEG+1,SO:POKEG+4,33:IFPEEK(S+40)<>5
    6ANDS<1944THEN375                       :rem 227
372 GOTO 380                                :rem 112
375 POKES,T:POKES+G,D(T-J):S=S+40:T=PEEK(
    S):POKES,58:POKES+G,0                   :rem 122
380 FORN=1TO17:NEXT:SO=SO-5:IFSO>150THEN3
    70                                      :rem 231
390 POKEG+4,16:CH=CH-1:IFCH=-1THENPOKE532
    72,21:GOTO1000                          :rem 2
400 PRINT"{HOME}{RVS}"TAB(25)CH:POKEV,W:P
    OKEV+G,D(W-J):Y=Y+1                     :rem 43
405 IFW=58THENPOKEV,T1:POKEV+G,D(T1-J)
                                            :rem 124
410 IFY>10THEN60                            :rem 177
420 IFS>1943THENPOKES,T:GOTO70              :rem 146
430 POKES,58:POKES+G,0:GOTO90               :rem 24
440 IFY=11THEN470                           :rem 233
450 FORN=Y+1TO11:POKE1123+B(N),62:SS=SS+1
    00:PRINT"{HOME}{RVS}"TAB(8-LEN(STR$(S
    S)))SS                                  :rem 187
455 GOSUB690:NEXT                           :rem 54
460 POKEG+4,33:FORM=9TO176STEP2:POKEG+1,M
    :NEXT:POKEG+4,32                        :rem 50
470 E2=E2+.05:SC=SC+1:E1=E1+1:IFE1>8THENE
      1=8                                   :rem 17
480 GOTO60                                  :rem 59
490 GOTO490                                 :rem 115
500 PRINT"{CLR}{PUR}";:FORN=1TO22:PRINTA$
    :NEXT:PRINTA$"{HOME}"                   :rem 220
505 B$=">888888888888888888888888888888888
    888888"                                 :rem 15
510 PRINT"{3 DOWN}"TAB(15)"?>>>>>>>>?
    {RED}":PRINTTAB(15)"9{PUR}88888888
    {RED}9"                                 :rem 121
515 PRINTTAB(15)"9>>>>>>>>>9":PRINTTAB(15)
    "9>>>>>>>>>9{PUR}"                      :rem 243
520 FORN=1TO3:PRINTB$"{4 DOWN}":NEXT:PRIN
    TB$"{HOME}";:POKE1983,62                :rem 207
530 PRINT"{RVS}"TAB(8-LEN(STR$(SS)))SS;TA
    B(25)CH;TAB(31)SC:POKE1054,163
                                            :rem 157
535 FORN=0TO11:POKE1123+B(N),60             :rem 18
540 POKE1123+B(N)+G,7:NEXT:FORN=1344TO194
    4STEP200:IFN=1944THEN600                :rem 143
550 FORO=1TO3                               :rem 20
560 R=N+1+INT(RND(1)*38):IFPEEK(R)<>56THE
    N560                                    :rem 118
570 FORM=RTOR+160STEP40:POKEM,57:POKEM+G,
    2:NEXT                                  :rem 3
575 IFO>1ANDRND(1)<E2THENPOKER+(INT(RND(1
    )*2)+2)*40,63                           :rem 132
580 IFRND(1)<.5ANDPEEK(R-40)=62THENPOKER-
    40,63                                   :rem 216
590 NEXT                                    :rem 221
600 FORO=1TOE1                              :rem 83
610 R=N+3+INT(RND(1)*36)                    :rem 50
615 IFPEEK(R)<>56ORPEEK(R-40)<>62ORPEEK(R
    +1)=62ORPEEK(R-1)=62THEN630             :rem 82
620 POKER,62:POKER-40,63                    :rem 142
630 NEXT                                    :rem 216
640 FORO=1TO4                               :rem 21
650 R=N-39+INT(RND(1)*38):IFPEEK(R)<>62OR
    PEEK(R+40)=62THEN650                    :rem 77
660 POKER,61:POKER+G,0:NEXT:NEXT            :rem 43
670 POKE1122,63:POKE1125,63:POKE1161,63:P
    OKE1166,63                              :rem 129
680 FORN=1304TO1904STEP200:POKEN,63:NEXT:
    FORN=1343TO1943STEP200:POKEN,63:NEXT
                                            :rem 90
685 RETURN                                  :rem 131
690 IFSS>=Q*E3THENCH=CH+1:E3=E3+1:PRINT"
    {HOME}{RVS}"TAB(25)CH                   :rem 45
700 RETURN                                  :rem 119
710 DATA,1,39,40,41,42,78,79,80,81,82,83
                                            :rem 231
720 DATA255,255,153,102,102,153,255,255,1
    95,255,255,195,195,255,255,195,60,60
                                            :rem 31
725 DATA25,255,188                          :rem 180
730 DATA60,36,231,3,4,24,24,60,126,126,60
    ,60,66,165,153,153,165,66,60,,24,36
                                            :rem 195
735 DATA126,126,126                         :rem 220
740 DATA126,,,,,,,,,,,,,,,,,                :rem 58
750 DATA169,,133,1,169,255,141,34,145,169
    ,32,44,31,145,208,5,169,1,133,1,96
                                            :rem 163
755 DATA169,8,44                            :rem 83
760 DATA31,145,208,5,169,2,133,1,96,169,1
    6,44,31,145,208,5,169,3,133,1,96,169
                                            :rem 15
765 DATA4,44,31                             :rem 20
770 DATA145,208,3,133,1,96,169,127,141,34
    ,145,169,128,44,32,145,208,4,169,5
                                            :rem 176
775 DATA133,1,96                            :rem 76
776 FORN=0TO11:READB(N):NEXT                :rem 251
777 RETURN                                  :rem 133
780 FORI=0TO11:READX:NEXT:PRINT"{9 DOWN}
    {9 RIGHT}REDEFINING CHARACTERS"
                                            :rem 204
782 POKE56334,PEEK(56334)AND254:POKE1,PEE
    K(1)AND251                              :rem 194
783 FORI=12288TO12288+256*8:POKEI,PEEK(I+
    40960):NEXTI                            :rem 115
784 POKE1,PEEK(1)OR4:POKE56334,PEEK(56334
    )OR1                                    :rem 146
785 FORI=12288+56*8TO12288+63*8+7:READM:P
    OKEI,M:NEXTI                            :rem 158
786 POKE53272,(PEEK(53272)AND240)+12
                                            :rem 196
800 RETURN                                  :rem 120
1000 PRINT"{CLR}YOUR SCORE IS ";SS
                                            :rem 109
1010 PRINT"{6 DOWN}DO YOU WISH TO PLAY AG
     AIN (Y/N) ?";                          :rem 158
1020 GETA$:IFA$=""THEN1020                  :rem 169
1030 IFA$="Y"THENPOKE53272,(PEEK(53272)AN
     D240)+12:GOTO2                         :rem 133
1040 IFA$<>"N"THEN1020                      :rem 183
1050 SYS2048                                :rem 147
```

Cave-In For 64

(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.

```
1 POKE56,28:CLR:DEFFNW(X)=PEEK(37151)AND3
  2:B=36865:GOSUB6                    :rem 186
2 PRINT"{CLR}{WHT}"SPC(73)"CAVE-IN"SPC(10
  1)"{BLK}F1 FOR HELP":GOTO56         :rem 101
3 A$="+01-22-01+22+01":F=A+VAL(MID$(A$,D*
  3+1,3))*V                           :rem 248
4 X=VAL(MID$(A$,D*3+4,3)):L=F+X:R=F-X:RET
  URN                                 :rem 118
5 PRINTSPC(230)"V":RETURN             :rem 194
6 Y=30:POKEB+14,42:POKEB+1,150:GOTO8
                                      :rem 112
7 Y=28:POKEB+14,107:POKEB+1,22:POKEB,25:P
  OKEA,VAL(MID$("235241243242",D*3+1,3))
                                      :rem 146
8 POKE648,Y:IFFNW(W)THENRETURN        :rem 249
9 GOTO8                               :rem 170
10 PRINTSPC(207)"M{DOWN}{LEFT}{M}{DOWN}
   {LEFT}N":RETURN                    :rem 186
11 PRINTSPC(161)"M{DOWN}M{DOWN}{LEFT}
   {M}{DOWN}{LEFT}{M}{DOWN}{LEFT}
   {M}{2 DOWN}{2 LEFT}N{UP}N":RETURN
                                      :rem 79
12 PRINTSPC(92)W$MID$(X$,37)"{2 DOWN}
   {3 LEFT}N{UP}N{UP}N":RETURN        :rem 81
13 PRINT"{DOWN}{RIGHT}"W$MID$(X$,19)"
   {2 DOWN}{3 LEFT}N{UP}N{UP}N":RETURN
                                      :rem 34
14 PRINT"M"X$"{LEFT}N":RETURN         :rem 72
15 PRINTSPC(209)"N{DOWN}{LEFT}{G}{DOWN}
   {LEFT}M":RETURN                    :rem 191
16 PRINTSPC(188)"N{UP}N{2 DOWN}{2 LEFT}
   {G}{DOWN}{LEFT}{G}{DOWN}{LEFT}
   {G}{DOWN}{LEFT}M{DOWN}M":RETURN
                                      :rem 87
17 PRINTSPC(146)"N{UP}N{UP}N{3 DOWN}
   {3 LEFT}{G}"MID$(Y$,40)W$:RETURN
                                      :rem 55
18 PRINTSPC(83)"N{UP}N{UP}N{3 DOWN}
   {3 LEFT}{G}"MID$(Y$,22)W$:RETURN
                                      :rem 8
19 PRINTSPC(20)"N"Y$"M{HOME}":RETURN
                                      :rem 93
20 PRINTSPC(229)"P{DOWN}{LEFT}{T}":RETU
   RN                                 :rem 66
21 PRINTSPC(205)"{T}P{DOWN}{LEFT}{M}
   {DOWN}{2 LEFT}{@}@":RETURN         :rem 141
22 PRINTSPC(158)"{2 T}P"MID$(X$,40)"
   {3 LEFT}{3 T}":RETURN              :rem 153
23 PRINTSPC(89)"{2 T}P"MID$(X$,22)"
   {3 LEFT}{3 T}":RETURN              :rem 109
24 PRINT"{DOWN}P"MID$(X$,4)"{LEFT}{T}":
   RETURN                             :rem 225
25 PRINTSPC(231)"O{DOWN}{LEFT}{T}":RETU
   RN                                 :rem 63
26 PRINTSPC(210)"O{T}{DOWN}{2 LEFT}
   {G}{DOWN}{LEFT}L{@}":RETURN
                                      :rem 157
27 PRINTSPC(168)"O{2 T}{DOWN}{3 LEFT}
   {G}"MID$(Y$,43)"{3 T}":RETURN
                                      :rem 88
28 PRINTSPC(105)"O{2 T}{DOWN}{3 LEFT}
   {G}"MID$(Y$,25)"{3 T}":RETURN
                                      :rem 80
29 PRINTSPC(42)"O"MID$(Y$,4)"{T}":RETUR
   N                                  :rem 213
30 PRINTSPC(230)"{T}{DOWN}{LEFT}{T}":
   RETURN                             :rem 14
31 PRINTSPC(207)"{3 T}{2 DOWN}{3 LEFT}
   {3 @}":RETURN                      :rem 237
32 PRINTSPC(161)MID$(Z$,13)SPC(147)MID$(Z
   $,13):RETURN                       :rem 104
33 PRINTSPC(92)MID$(Z$,7)"{DOWN}"SPC(251)
   MID$(Z$,7):RETURN                  :rem 51
34 PRINTSPC(23)Z$SPC(154)SPC(245)Z$"
   {HOME}":RETURN                     :rem 160
35 PRINT"{CLR}{WHT}":FORV=0TO5:GOSUB3:IFP
   EEK(F)=32THENPRINT"{HOME}":ONVGOSUB34,
   33,32,31,30:GOTO41                 :rem 226
36 PRINT"{HOME}":IFPEEK(L)=32THENONV+1GOS
   UB14,13,12,11,10,5:GOTO38          :rem 86
37 ONV+1GOSUB24,23,22,21,20,5         :rem 62
38 PRINT"{HOME}":IFPEEK(R)=32THENONV+1GOS
   UB19,18,17,16,15,5:GOTO40          :rem 112
39 ONV+1GOSUB29,28,27,26,25,5         :rem 89
40 NEXT                               :rem 163
41 GOSUB77:W=PEEK(37151):IFY=30GOTO44
                                      :rem 251
42 IF(WAND32)=0THENPOKEB,Z:GOSUB6:POKEA,X
                                      :rem 134
43 GOTO41                              :rem 5
44 IFK=0ANDP<TITHENX=PEEK(B)+1:POKEB,X:P=
   TI+40:IFX=122THENRETURN            :rem 167
45 IF(WAND4)GOTO51                    :rem 70
46 V=1:GOSUB3:IFPEEK(F)<>32THENA=F:POKE30
   720+A,1:GOTO35                     :rem 123
47 IFA=7397ANDK=0THENRETURN           :rem 236
48 IFPEEK(A)<>13GOTO51                :rem 47
49 K=K-1:PRINTSPC(116)"MAN FOUND"SPC(34)"
   MEN LEFT="K:POKEA,160              :rem 178
50 IFK=0THENPRINTSPC(72)"CAVE-IN":IFGTHEN
   O=1                                :rem 43
51 IF(WAND16)=0THEND=VAL(MID$(D$,D+3,1)):
   GOTO35                             :rem 229
52 POKE37154,127:X=PEEK(37152)AND128:POKE
   37154,255:IFX=0THEND=VAL(MID$(D$,D+1,1
   )):GOTO35                          :rem 119
53 IF(WAND8)=0THEND=VAL(MID$(D$,D+4,1)):G
   OTO35                              :rem 185
54 IF(WAND32)=0ANDO=0THENZ=PEEK(B):X=PEEK
   (A):GOSUB7                         :rem 201
55 GOTO41                              :rem 8
56 D=3:D$="+02-44-02+44":PRINTSPC(91)"
   {WHT}MINE BEING DUG                :rem 131
57 POKE648,28:A=7397:X$="{21 SPACES}":PRI
   NT"{CLR}{RVS}{CYN} "X$;:FORW=1TO21
                                      :rem 142
58 PRINT"{OFF}{BLU}"X$"{CYN}{RVS} ";:NEXT
   :PRINTX$"{HOME}":POKE7673,160:POKE3839
   3,3:PRINTSPC(141)"{WHT}E           :rem 131
59 GOSUB77:X=INT(RND(1)*4):Y=X        :rem 83
60 W=A+VAL(MID$(D$,X*3+1,3))          :rem 237
61 IFPEEK(W)=32THENZ=0:POKEW,X:POKEA+VAL(
   MID$(D$,X*3+1,3))/2,160:A=W:GOTO59
                                      :rem 15
62 X=(X+1)*-(X<3):IFX<>YGOTO60        :rem 4
63 X=PEEK(A):POKEA,160:IFZ=0THENPOKEA,13:
   Z=1:K=K+1                          :rem 76
```

```
64 IFX<>5THENA=A-VAL(MID$(D$,X*3+1,3)):GO
   TO59                                :rem 131
65 W$="M{DOWN}M{DOWN}M":X$="{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}{LEFT}[M]{DOWN}{LEFT}
   [M]{DOWN}                       :rem 101
66 Y$="{DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}[G]
   {DOWN}{LEFT}[G]{DOWN}{LEFT}  :rem 31
67 D$="3012301":GOSUB6:Z$="[19 T]":POKE
   A,209:GOSUB35                   :rem 133
68 GOSUB7:POKE217,156:POKE218,156:rem 188
69 PRINT"{HOME}{RVS}{CYN}PRESS THE FIRE B
   UTTON{OFF}{RIGHT}{WHT}TWICE=PLAY - ONC
   E=END                           :rem 91
70 FORW=37933TO38329STEP22:FORX=0TO18:POK
   EW+X,1:NEXT:NEXT                :rem 187
71 GOSUB77:IFFNW(X)GOTO71          :rem 84
72 GOSUB6:PRINT"{CLR}              :rem 155
73 IFFNW(X)=0GOTO73                :rem 157
74 FORW=0TO30:IFFNW(X)=0GOTO88     :rem 143
75 NEXT                            :rem 171
76 POKE56,30:CLR:END               :rem 194
77 GETA$:IFA$<>CHR$(133)THENRETURN:rem 83
78 POKEB,25:GOSUB6:PRINT"{CLR}PICK ONE
   {BLK}":PRINT"{DOWN} F1=NOVICE":PRINT"
   {DOWN} F3=ADVANCED              :rem 20
79 PRINT"{DOWN} F5=OLD MAP":PRINT"{DOWN}
   {SPACE}F7=END":PRINT"{DOWN}{WHT}*CURRE
   NT LEVEL                        :rem 56
80 PRINT"{3 DOWN}GOAL-{BLK}FIND THE MINER
   S"SPC(7)"AND GET BACK           :rem 112
81 PRINT"{WHT}{2 DOWN}JOYSTICK-{BLK}MOVE"
   :PRINT"SEE MAP Q↑               :rem 152
82 PRINT"SEE LEFT<W>SEE RIGHT"SPC(11)"V"S
   PC(18)"SEE BACK[HOME]{WHT}":IFGTHENPRI
   NT"{3 DOWN}*":GOTO84            :rem 88
83 PRINT"{DOWN}*                   :rem 85
84 GETA$:IFA$<>""THENW=ASC(A$)-132:ONABS(
   W)GOTO86,87,68,72               :rem 118
85 GOTO84                          :rem 18
86 G=0:GOTO88                      :rem 5
87 G=1                             :rem 36
88 O=0:K=0:PRINT"{CLR}":GOTO56     :rem 154
```

Alpha-Shoot

(Article on page 118.)

Program 1: Alpha-Shoot — VIC Version

```
5 PRINT"{CLR}":Y=7900:SV=36878:SS=36876:C
  L=36879:POKECL,78:POKESV,14      :rem 225
7 H=8108:CL=30720:J=37137:POKE650,128:POK
  E651,1                           :rem 130
8 DIM AB(26):FORX=1TO26:AB(X)=32:NEXTX
                                   :rem 25
10 GOSUB200                        :rem 115
12 GOSUB40                         :rem 71
14 GOSUB215:FORX=8120TO8141:POKEX,67:POKE
   X+30720,3:NEXTX                 :rem 225
16 GOSUB250:GOSUB265               :rem 213
30 GOSUB435:GOTO16                 :rem 89
40 PRINT"{CLR}":RESTORE:Y=7900     :rem 198
42 READL:IFL=0THEN90               :rem 232
43 POKE Y,L                        :rem 103
50 POKEY,L:READ P:POKESS,P:READ D  :rem 48
60 FORX=1TOD:NEXTX:POKESS,0        :rem 54
70 Y=Y+1:FORX=1TO10:NEXTX:IFL=32THENY=Y-1
                                   :rem 161
80 IFY=7922THENY=7952              :rem 253
85 GOTO 42                         :rem 12
90 READA$:IFA$="0"THEN100          :rem 137
92 READP:READD                     :rem 113
95 PRINT"{4 RIGHT}"A$:POKESS,P     :rem 163
97 FORX=1TOD:NEXTX:POKESS,0:FORX=1TO10:NE
   XTX:PRINT"{HOME}":GOTO90        :rem 246
100 FORX=1TO1500:NEXTX:RETURN      :rem 146
170 DATA 1,135,310,2,135,310,3,175,310,4,
    175,310,5,183,310,6,183,310    :rem 239
173 DATA 7,175,615,8,163,310,9,163,310,10
    ,159,310,11,159,310            :rem 131
175 DATA 12,147,120,13,147,120,14,147,120
    ,15,147,120,16,135,602         :rem 5
177 DATA17,175,310,18,175,310,19,163,601,
    20,159,310
178 DATA21,159,310,22,147,601,23,175,121
                                   :rem 221
179 DATA 32,175,121,32,175,231,24,163,601
    ,25,159,310,32,159,310         :rem 26
181 DATA 26,147,605,0              :rem 7
183 DATA"NOW",135,310,"{4 RIGHT}I",135,31
    0,"{6 RIGHT}KNOW",175,310      :rem 11
184 DATA"{11 RIGHT}MY",175,310     :rem 105
185 DATA"{DOWN}{4 RIGHT}A",183,310,"
    {DOWN}{5 RIGHT}B",183,310,"{DOWN}
    {6 RIGHT}C'S",175,610          :rem 158
187 DATA"{2 DOWN}NEXT",163,310,"{2 DOWN}
    {5 RIGHT}TIME",163,310,"{2 DOWN}
    {10 RIGHT}WON'T",159,310       :rem 107
189 DATA"{3 DOWN}YOU",159,310,"{3 DOWN}
    {4 RIGHT}SING",147,310,"{3 DOWN}
    {9 RIGHT}WITH",147,310         :rem 247
190 DATA"{5 DOWN}{5 RIGHT}ME",135,630,"0"
                                   :rem 154
200 PRINTTAB(5):PRINT"ALPHA-SHOOT"
                                   :rem 167
202 LE=-1:KR=0                     :rem 4
204 PRINT"{2 DOWN}WHICH GAME- 1,2,3,OR 4"
                                   :rem 50
205 POKE198,0:WAIT198,1:GETA$:     :rem 235
206 IFA$="1"THEN211                :rem 1
207 IFA$="2"THENLE=0:GOTO 211      :rem 116
208 IFA$="3"THENLE=1:GOTO 211      :rem 119
209 IFA$="4"THENLE=2:GOTO 211      :rem 122
210 GOTO205                        :rem 99
211 RETURN                         :rem 116
215 R$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
                                   :rem 110
225 PRINT"{CLR}":POKEH,83:POKE36879,10:GO
    SUB228:POKE7703+V,R:RETURN     :rem 245
228 R=INT(LEN(R$)*RND(1)+1):P=ASC(MID$(R$
    ,R,1))-64                      :rem 31
229 IFLE=1THEN232                  :rem 241
```

```
230 R$=LEFT$(R$,R-1)+RIGHT$(R$,LEN(R$)-R)
                                    :rem 31
232 R=P:V=INT(RND(1)*350+1)         :rem 3
236 IFLE=0THENKR=KR+1:R=KR          :rem 144
237 IFLE=1THENWAIT198,1:GETB$:R=ASC(B$)-6
    4                               :rem 251
238 IFR>26ORR<1THENR=1              :rem 115
239 RETURN                          :rem 126
250 POKE37139,0:X=(PEEK(37137)AND60)/4
                                    :rem 96
252 POKE37154,127:J=PEEK(37152)AND128:POK
    E37154,255                      :rem 110
255 IFX=11THEND=-1:GOSUB275         :rem 136
257 IFJ=0 THEND= 1:GOSUB275         :rem 29
259 IFX=7THENGOSUB300               :rem 61
260 RETURN                          :rem 120
265 GETA$:IFA$=""THENGOTO270        :rem 146
266 IFA$="C"THEND=-1:GOSUB275       :rem 188
267 IFA$="B"THEND=+1:GOSUB275       :rem 186
268 IFA$=" "THENGOSUB300            :rem 87
270 RETURN                          :rem 121
275 X=H+D:IFX<8098ORX>8119THENRETURN
                                    :rem 57
276 POKESS,130:POKEH,32:POKEX,83:H=X
                                    :rem 229
278 POKESS,0:RETURN                 :rem 236
300 G=H:FORU=1TO19:G=G-22:IFPEEK(G)<>32TH
    ENPOKEG,32:POKEG+22,32:GOTO350:rem 92
305 POKESS,U+220:POKEG,30:IFU>1THENPOKEG+
    22,32                           :rem 62
306 GOSUB435:NEXTU:POKESS,0:POKEG,32:RETU
    RN                              :rem 73
350 POKESS,0:POKE36877,220:FORL=13TO0STEP
    -1:POKE36878,L:POKE36879,40:GOSUB375
                                    :rem 208
355 NEXTL:POKE36877,0:POKE36878,14:GOSUB3
    90                              :rem 92
357 POKE36879,10:GOSUB228:POKE7703+V,R:RE
    TURN                            :rem 21
375 POKEG,90:POKEG+22,42:POKEG-22,42:POKE
    G+1,42:POKEG-1,42               :rem 140
377 POKEG+23,77:POKEG-23,77:POKEG-21,78:P
    OKEG+21,78                      :rem 209
379 POKEG,32:POKEG+22,32:POKEG-22,32:POKE
    G-1,32:POKEG+1,32               :rem 136
381 POKEG-23,32:POKEG+23,32:POKEG-21,32:P
    OKEG+21,32:RETURN               :rem 192
390 AB(R)=R:FORX=1TO22:POKE8141+X,AB(X):P
    OKE8141+X+30720,7               :rem 149
392 NEXTX:FORX=23TO26:POKE8150+X,AB(X):PO
    KE8150+X+30720,7:NEXT           :rem 42
394 FORX=1TO26:IFAB(X)=32THENRETURN
                                    :rem 254
395 NEXTX:FORX=1TO26:AB(X)=32:NEXTX:POKE3
    6879,78:FORW=1TO1000:NEXTW:GOSUB40:RU
    N                               :rem 127
435 IFLE<2THENRETURN                :rem 57
436 Q=V+7703:IFPEEK(162)<41 THEN RETURN
                                    :rem 5
440 IFQ>8074THENPOKEQ,32:V=2:RETURN
                                    :rem 226
442 POKEQ,32:POKEQ+1,R:V=V+1:POKE162,0
                                    :rem 28
445 RETURN                          :rem 125
```

Program 2: Alpha-Shoot — 64 Version

```
4 POKE53280,6:POKE53281,1:FORT=54272TO542
  96:POKET,0:NEXTT                  :rem 248
5 PRINT"{CLR}":Y=1424:HF=54273:LF=54272:C
  L=54272:POKE54296,15:POKE54277,66
                                    :rem 100
6 POKE54278,68:POKE54284,17:POKE54285,250
  :POKE54279,100:POKE54280,100      :rem 155
7 H=1798:POKE650,128:POKE651,1      :rem 194
8 DIM AB(26):FORX=1TO26:AB(X)=32:NEXTX
                                    :rem 25
10 GOSUB200                         :rem 115
12 GOSUB40                          :rem 71
14 GOSUB215:FORX=1824TO1863:POKEX,67:POKE
   X+54272,3:NEXTX                  :rem 241
16 GOSUB250:GOSUB265                :rem 213
30 GOSUB435:GOTO16                  :rem 89
40 PRINT"{CLR}":RESTORE:POKE53281,1:Y=155
   1                                :rem 139
42 READL:IFL=0THEN90                :rem 232
43 POKE Y,L:POKEY+CL,2              :rem 65
50 READHP:POKEHF,HP:READ LP:POKELF,LP:REA
   D:POKE54276,17                   :rem 40
60 FORX=1TOD:NEXTX:POKE54276,16     :rem 207
70 Y=Y+1:FORX=1TO10:NEXTX:IFL=32THENY=Y-1
                                    :rem 161
80 IFY=1464THENY=1517               :rem 239
85 GOTO 42                          :rem 12
90 READA$:IFA$="0"THEN100           :rem 137
92 READHP:READLP:READD              :rem 171
95 PRINT"{13 RIGHT}"A$:POKE54276,17:POKEH
   F,HP:POKELF,LP                   :rem 160
97 FORI=1TOD+7:NEXT:POKE54276,16:FORX=1TO
   10:NEXT:PRINT"{HOME}":GOTO90     :rem 50
100 FORX=1TO1500:NEXTX:RETURN       :rem 146
170 DATA 1,34,75,310,2,34,75,310,3,51,97,
    310,4,51,97,310,5,57,172,310    :rem 49
171 DATA6,57,172,310                :rem 7
173 DATA 7,51,97,615,8,45,198,310,9,45,19
    8,310,10,43,52,310,11,43,52,310
                                    :rem 216
175 DATA 12,38,126,120,13,38,126,120,14,3
    8,126,120,15,38,126,120,16,34,75,602
                                    :rem 187
177 DATA17,51,97,310,18,51,97,310,19,45,1
    98,601,20,43,52,310,21,43,52,310
                                    :rem 55
178 DATA22,38,126,601,23,51,97,121
                                    :rem 188
179 DATA 32,51,97,121,32,51,97,231,24,45,
    198,601,25,43,52,310,32,43,52,310
                                    :rem 55
181 DATA 26,38,126,605,0            :rem 155
182 DATA"NOW"                       :rem 237
183 DATA34,75,310,"{4 RIGHT}I",34,75,310,
    "{6 RIGHT}KNOW",51,97,310,"{11 RIGHT}
    MY"                             :rem 45
184 DATA51,97,310                   :rem 121
185 DATA"{DOWN}{4 RIGHT}A",57,172,310,"
    {DOWN}{5 RIGHT}B",57,172,310,"{DOWN}
    {6 RIGHT}C'S",51,97,610         :rem 47
187 DATA"{2 DOWN}NEXT",45,198,310,"
    {2 DOWN}{5 RIGHT}TIME",45,198,310,"
    {2 DOWN}{10 RIGHT}WON'T"        :rem 26
188 DATA43,52,310                   :rem 117
189 DATA"{3 DOWN}YOU",43,52,310,"{3 DOWN}
    {4 RIGHT}SING",38,126,310,"{3 DOWN}
    {9 RIGHT}WITH"                  :rem 94
190 DATA38,126,310,"{5 DOWN}{5 RIGHT}ME",
    34,75,630,"0"                   :rem 28
200 PRINT"{HOME}{10 DOWN}{14 RIGHT}{RVS}
    {BLU}ALPHA SHOOT"               :rem 218
202 LE=-1:KR=0                      :rem 4
```

```
204 PRINT"{HOME}{12 DOWN}{9 RIGHT}WHICH G
    AME-1,2,3,OR 4"                :rem 244
205 POKE198,0:WAIT198,1:GETA$:     :rem 235
206 IFA$="1"THEN211                  :rem 1
207 IFA$="2"THENLE=0:GOTO 211      :rem 116
208 IFA$="3"THENLE=1:GOTO 211      :rem 119
209 IFA$="4"THENLE=2:GOTO 211      :rem 122
210 GOTO205                         :rem 99
211 RETURN                         :rem 116
215 R$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
                                   :rem 110
225 PRINT"{CLR}":POKEH,83:POKE53281,0:POK
    E53280,2:GOSUB228:POKE1065+V,R:RETURN
                                   :rem 122
228 R=INT(LEN(R$)*RND(1)+1):P=ASC(MID$(R$
    ,R,1))-64                      :rem 31
229 IFLE=1THEN232                  :rem 241
230 R$=LEFT$(R$,R-1)+RIGHT$(R$,LEN(R$)-R)
    :R=P                           :rem 56
232 V=INT(RND(1)*350+1)            :rem 234
236 IFLE=0THENKR=KR+1:R=KR         :rem 144
237 IFLE=1THENPOKE198,0:WAIT198,1:GETB$:R
    =ASC(B$)-64                    :rem 98
239 IFR>26ORR<0THENPOKE198,0:GOTO237
                                   :rem 239
240 RETURN                         :rem 118
250 XV=(PEEK(56320)AND15)          :rem 130
252 JV=15-XV:KV=(PEEK(56320)AND16)
                                   :rem 208
255 IFJV=4THEND=-1:GOSUB274        :rem 161
257 IFJV=8THEND=1:GOSUB274         :rem 122
259 IFKV=0THENGOSUB300             :rem 127
260 RETURN                         :rem 120
265 GETA$:IFA$=""THEN270            :rem 89
266 IFA$="C"THEND=-1:GOSUB274      :rem 187
267 IFA$="B"THEND=1:GOSUB274       :rem 142
268 IFPEEK(197)=60THENPOKE198,0:POKE197,0
    :GOSUB300                      :rem 248
270 RETURN                         :rem 121
274 H=H+D:IFH<1784THENH=1784       :rem 130
275 IFH>1823THENH=1823               :rem 3
276 POKEHF,10:POKELF,70:POKEH-D,32:POKEH,
    83:POKEH+CL,1                   :rem 58
277 POKE54276,33:POKE54276,32:RETURN
                                   :rem 133
300 G=H:FORU=1TO19:G=G-40:IFPEEK(G)<>32TH
    ENPOKEG,32:POKEG+40,32:GOTO350:rem 92
305 POKE54276,17:POKEHF,U+60:POKELF,U+125
    :POKEG,30:IFU>1THENPOKEG+40,32:rem 60
307 GOSUB435:NEXTU:POKEG,32:POKE54276,16:
    RETURN                         :rem 227
350 POKE54283,129:GOSUB374:POKE53280,2:PO
    KE53281,0                      :rem 126
355 POKE54283,128:GOSUB390         :rem 239
357 GOSUB228:POKE1065+V,R:RETURN     :rem 9
374 POKE53280,0:POKE53281,2        :rem 245
375 POKE54276,16:POKEG,90:POKEG+40,42:POK
    EG-40,42:POKEG+1,42:POKEG-1,42
                                   :rem 144
377 POKEG+41,77:POKEG-41,77:POKEG-39,78:P
    OKEG+39,78:FORL=0TO300:NEXT    :rem 108
379 POKEG,32:POKEG+40,32:POKEG-40,32:POKE
    G-1,32:POKEG+1,32              :rem 136
381 POKEG-41,32:POKEG+41,32:POKEG-39,32:P
    OKEG+39,32:RETURN              :rem 210
390 AB(R)=R:FORX=1TO22:POKE1873+X,AB(X):P
    OKE1873+X+54272,7              :rem 167
392 NEXTX:FORX=23TO26:POKE1938+X,AB(X):PO
    KE1938+X+54272,7:NEXT          :rem 64
394 FORX=1TO26:IFAB(X)=32THENRETURN
                                   :rem 254
395 NEXTX:FORX=1TO26:AB(X)=32:NEXTX:FORW=
    1TO1000:NEXTW:GOSUB40:RUN      :rem 106
435 IFLE<2THENRETURN               :rem 57
436 Q=V+1065:IFPEEK(162)<41 THEN RETURN
440 IFQ>1742THENPOKEQ,32:V=2:RETURN
                                   :rem 221
442 POKEQ,32:POKEQ+1,R:V=V+1:POKE162,0
                                   :rem 28
445 RETURN                         :rem 125
```

Machine Language For Beginners

(Article on page 150.)

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

```
12288   LDY    # 0
12290   LDA    # 8
12292   STA    37888 ,Y
12295   STA    38144 ,Y
12298   STA    38656 ,Y
12301   INY
12302   BNE          12292
12304   LDY    # 0
12306   LDA    # 224
12308   STA    4096 ,Y
12311   STA    4580 ,Y
12314   INY
12315   CPY    # 22
12317   BNE          12308
12319   LDA    # 21
12321   STA    71
12323   LDA    # 16
12325   STA    72
12327   LDX    # 24
12329   LDY    # 0
12331   LDA    # 224
12333   STA  ( 71 ),Y
12335   INY
12336   STA  ( 71 ),Y
12338   DEX
12339   BEQ          12357
12341   CLC
12342   LDA    71
12344   ADC    # 22
12346   STA    71
12348   LDA    72
12350   ADC    # 0
12352   STA    72
12354   JMP     --->    12329
12357   RTS
```

Program 3: VIC BASIC Loader

```
1000 FORI=12288TO12359:READ DA:POKEI,DA:N
     EXT                           :rem 145
12288 DATA 160,000,169,008,153,000
                                   :rem 131
```

```
12294 DATA 148,153,000,149,153,000
                                    :rem 133
12300 DATA 151,200,208,244,160,000
                                    :rem 112
12306 DATA 169,224,153,000,016,153
                                    :rem 131
12312 DATA 228,017,200,192,022,208
                                    :rem 127
12318 DATA 245,169,021,133,071,169
                                    :rem 146
12324 DATA 016,133,072,162,024,160
                                    :rem 127
12330 DATA 000,169,224,145,071,200
                                    :rem 123
12336 DATA 145,071,202,240,016,024
                                    :rem 126
12342 DATA 165,071,105,022,133,071
                                    :rem 127
12348 DATA 165,072,105,000,133,072
                                    :rem 131
12354 DATA 076,041,048,096,013,013
                                    :rem 138
```

Program 4: 64 BASIC Loader

```
1000 FORI=49152TO49229:READ DA: POKEI,DA:
     NEXT                           :rem 151
49152 DATA 160,000,169,008,153,000
                                    :rem 131
49158 DATA 216,153,000,217,153,000
                                    :rem 134
49164 DATA 218,153,000,219,200,208
                                    :rem 138
49170 DATA 241,160,000,169,224,153
                                    :rem 138
49176 DATA 000,004,153,192,007,200
                                    :rem 131
49182 DATA 192,040,208,245,169,039
                                    :rem 159
49188 DATA 133,071,169,004,133,072
                                    :rem 151
49194 DATA 162,024,160,000,169,224
                                    :rem 143
49200 DATA 145,071,200,145,071,202
                                    :rem 127
49206 DATA 240,016,024,165,071,105
                                    :rem 136
49212 DATA 040,133,071,165,072,105
                                    :rem 134
49218 DATA 000,133,072,076,044,192
                                    :rem 143
49224 DATA 096,013,013,013,013,013
                                    :rem 126
```

Graph Plotter

(Article on page 145.)

Program 1: Graph Plotter—64 Version

```
100 PRINT"{CLR}":POKE53281,0:POKE53280,6
                                    :rem 138
110 FORI=1230TO1261:POKEI,114:POKEI+54272
    ,5:NEXT:REM TOP                 :rem 114
120 FORI=1270TO1790STEP40:POKEI,107:POKEI
    +54272,5:NEXT:REM LEFT          :rem 88
130 FORI=1301TO1821STEP40:POKEI,115:POKEI
    +54272,5:NEXT:REM RIGHT         :rem 161
140 FORI=1830TO1861:POKEI,113:POKEI+54272
    ,5:NEXT:REM BOTTOM              :rem 98
150 FORI=1351TO1380:POKEI,67:POKEI+54272,
    5:NEXT                          :rem 66
160 FORI=1471TO1500:POKEI,67:POKEI+54272,
    5:NEXT                          :rem 64
170 FORI=1591TO1620:POKEI,67:POKEI+54272,
    5:NEXT                          :rem 71
180 FORI=1711TO1740:POKEI,67:POKEI+54272,
    5:NEXT:PRINT:PRINT:PRINT        :rem 154
190 PRINT: PRINTSPC(4)"15":PRINTSPC(4)"14
    ":PRINTSPC(4)"13":PRINTSPC(4)"12"
                                    :rem 145
200 PRINTSPC(4)"11":PRINTSPC(4)"10":PRINT
    SPC(5)"9":PRINTSPC(5)"8"        :rem 102
210 PRINTSPC(5)"7":PRINTSPC(5)"6":PRINTSP
    C(5)"5":PRINTSPC(5)"4":PRINTSPC(5)"3"
                                    :rem 181
220 PRINTSPC(5)"2":PRINTSPC(5)"1":PRINTSP
    C(5)"0"                         :rem 82
230 PRINTTAB(9)"A"SPC(4)"B"SPC(4)"C"SPC(4
    )"D"SPC(4)"E"SPC(4)"F"          :rem 199
240 IFZ$="Y"THEN310                 :rem 64
250 DATAA,B,C,D,E,F                 :rem 34
260 FORI=1TO6:READA$(I):NEXTI       :rem 38
270 DATA7,6,4,13,8,14               :rem 49
280 FORI=1TO6:READD(I):NEXTI         :rem 7
290 DATA1792,1797,1802,1807,1812,1817
                                    :rem 116
300 FORI=1TO6:READA(I):NEXTI        :rem 253
310 C=54272:FORJ=1TO6               :rem 202
320 FORK=1971TO1975:POKEK,32:NEXTK:PRINT"
    {HOME}":FORL=1TO22:PRINT:NEXTL
                                    :rem 228
330 PRINT"VALUE FOR COLUMN ";A$(J);" (0-1
    5) ";                           :rem 162
340 INPUTA$:IFVAL(A$)=0THEN320      :rem 31
350 AA=VAL(A$):AA=INT(AA+.5):D=D(J):X=A(J
    )                               :rem 122
360 IFAA<0ORAA>15THEN320            :rem 161
370 A=A(J)-(AA*40)+80:GOSUB430:NEXTJ
                                    :rem 41
380 PRINT"{HOME}":FORI=1TO22:PRINT:NEXT:F
    ORI=1971TO1983:POKEI,32:NEXT    :rem 75
390 PRINT"WISH TO DO IT AGAIN? (Y/N)"
                                    :rem 65
400 GETZ$:IFZ$=""THEN400            :rem 125
410 IFZ$="Y"THEN100                 :rem 60
420 END                             :rem 109
430 IFAA=1THEN530                   :rem 221
440 IFAA=0THEN RETURN               :rem 37
450 POKEX,160:POKEX+1,231:POKEX+2,105
                                    :rem 162
460 POKEX+C,D:POKE(X+1)+C,D:POKE(X+2)+C,D
    :IFAA=2THEN490                  :rem 30
470 FORI=X-40TOASTEP-40:POKEI,160:POKEI+1
    ,231:POKEI+2,160                :rem 185
480 POKEI+C,D:POKE(I+1)+C,D:POKE(I+2)+C,D
    :NEXTI:GOTO510                  :rem 56
490 POKEA,247:POKEA+1,208:POKEA+2,105
                                    :rem 107
500 POKEA+C,D:POKE(A+1)+C,D:POKE(A+2)+C,D
    :GOTO530                        :rem 89
510 POKEA,247:POKEA+1,208:POKEA+2,224
                                    :rem 102
520 POKEA+C,D:POKE(A+1)+C,D:POKE(A+2)+C,D
                                    :rem 80
530 POKEA-40,233:POKEA-39,160:POKEA-38,20
    6                               :rem 105
540 POKE(A-40)+C,D:POKE(A-39)+C,D:POKE(A-
    38)+C,D                         :rem 172
550 RETURN                          :rem 122
```

Program 2:
Graph Plotter—VIC Version

```
100 PRINT"{CLR}":POKE36879,11     :rem 253
110 FORI=7726TO7745:POKEI,114:POKEI+30720
    ,5:NEXTI                      :rem 191
120 FORI=7748TO8034STEP22:POKEI,107:POKEI
    +30720,5:NEXTI                :rem 94
130 FORI=7767TO8053STEP22:POKEI,115:POKEI
    +30720,5:NEXTI                :rem 96
140 FORI=8056TO8075:POKEI,113:POKEI+30720
    ,5:NEXTI                      :rem 187
150 FORI=7793TO7810:POKEI,64:POKEI+30720,
    5:NEXTI                       :rem 148
160 FORI=7859TO7876:POKEI,64:POKEI+30720,
    5:NEXTI                       :rem 164
170 FORI=7925TO7942:POKEI,64:POKEI+30720,
    5:NEXTI                       :rem 153
180 FORI=7991TO8008:POKEI,64:POKEI+30720,
    5:NEXTI:PRINT                 :rem 94
190 PRINTSPC(0)"15":PRINTSPC(0)"14":PRINT
    SPC(0)"13":PRINTSPC(0)"12":PRINTSPC(0
    )"11"                         :rem 142
200 PRINTSPC(0)"10":PRINTSPC(1)"9":PRINTS
    PC(1)"8":PRINTSPC(1)"7"       :rem 44
210 PRINTSPC(1)"6":PRINTSPC(1)"5":rem 170
220 PRINTSPC(1)"4":PRINTSPC(1)"3":PRINTSP
    C(1)"2":PRINTSPC(1)"1":PRINTSPC(1)"0"
                                  :rem 147
230 PRINTTAB(4)"A"SPC(2)"B"SPC(2)"C"SPC(2
    )"D"SPC(2)"E"SPC(2)"F"        :rem 184
240 IFZ$="Y"THEN310               :rem 64
250 DATAA,B,C,D,E,F               :rem 34
260 FORI=1TO6:READA$(I):NEXTI     :rem 38
270 DATA7,6,4,3,5,1               :rem 201
280 FORI=1TO6:READD(I):NEXTI       :rem 7
290 DATA8035,8038,8041,8044,8047,8050
                                  :rem 113
300 FORI=1TO6:READA(I):NEXTI      :rem 253
310 C=30720:FORJ=1TO6             :rem 194
320 FORK=8138TO8141:POKEK,32:NEXTK:PRINT"
    {HOME}{19 DOWN}"              :rem 179
330 PRINT"COLUMN ";A$(J);" (0-15) ";
                                  :rem 62
340 INPUT Y$:IFVAL(Y$)=0THEN320   :rem 79
350 Y=VAL(Y$):Y=INT(Y+.5):D=D(J):X=A(J)
                                  :rem 23
360 IFY<0ORY>15THEN320            :rem 79
370 A=A(J)-(Y*22)+44:GOSUB430:NEXTJ
380 PRINT"{HOME}{20 DOWN}"        :rem 211
390 PRINT"DO IT AGAIN? (Y/N)"     :rem 99
400 GETZ$:IFZ$=""THEN400          :rem 125
410 IFZ$="Y"THEN100               :rem 60
420 END                           :rem 109
430 IFY=1THEN530                  :rem 180
440 IFY=0THENRETURN               :rem 252
450 POKEX,160:POKEX+1,231:POKEX+2,105
                                  :rem 162
460 POKEX+C,D:POKE(X+1)+C,D:POKE(X+2)+C,D
    :IFY=2THEN490                 :rem 245
470 FORI=X-22TOASTEP-22:POKEI,160:POKEI+1
    ,231:POKEI+2,160              :rem 185
480 POKEI+C,D:POKE(I+1)+C,D:POKE(I+2)+C,D
    :NEXTI:GOTO510                :rem 56
490 POKEA,227:POKEA+1,208:POKEA+2,105
                                  :rem 105
500 POKEA+C,D:POKE(A+1)+C,D:POKE(A+2)+C,D
    :GOTO530                      :rem 89
510 POKEA,227:POKEA+1,208:POKEA+2,224
                                  :rem 100
520 POKEA+C,D:POKE(A+1)+C,D:POKE(A+2)+C,D
                                  :rem 80
530 POKEA-22,233:POKEA-21,160:POKEA-20,20
    6                             :rem 87
540 POKE(A-22)+C,D:POKE(A-21)+C,D:POKE(A-
    20)+C,D                       :rem 154
550 RETURN                        :rem 122
```

64 BASIC Aid

(Article on page 156.)

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.

```
39852 :173,254,159,133,055,173,095
39858 :255,159,133,056,169,076,002
39864 :133,124,173,217,155,133,095
39870 :125,173,218,155,133,126,096
39876 :076,143,156,240,003,076,122
39882 :008,175,169,201,133,124,244
39888 :169,058,133,125,169,176,014
39894 :133,126,096,219,155,133,052
39900 :139,134,151,186,189,001,252
39906 :001,201,140,240,016,208,008
39912 :002,164,140,166,151,165,252
39918 :139,201,058,176,003,076,123
39924 :128,000,096,189,002,001,148
39930 :201,164,208,237,165,139,084
39936 :016,002,230,122,132,140,130
39942 :162,000,134,165,202,232,133
39948 :164,122,185,000,002,056,029
39954 :253,217,159,240,019,201,083
39960 :128,240,019,230,165,232,014
39966 :189,216,159,016,250,189,025
39972 :217,159,208,228,240,191,255
39978 :232,200,208,224,132,122,136
39984 :165,165,010,170,189,245,224
39990 :159,072,189,244,159,072,181
39996 :032,233,155,076,115,000,159
40002 :032,178,157,165,095,166,091
40008 :096,133,036,134,037,032,028
40014 :019,166,165,095,166,096,017
40020 :144,010,160,001,177,095,159
40026 :240,004,170,136,177,095,144
40032 :133,122,134,123,165,036,041
40038 :056,229,122,170,165,037,113
40044 :229,123,168,176,030,138,204
40050 :024,101,045,133,045,152,102
40056 :101,046,133,046,160,000,094
40062 :177,122,145,036,200,208,246
40068 :249,230,123,230,037,165,142
40074 :046,197,037,176,239,032,097
40080 :051,165,165,034,166,035,248
40086 :024,105,002,133,045,144,091
40092 :001,232,134,046,032,089,178
40098 :166,076,131,164,032,124,087
40104 :165,032,115,000,133,139,240
40110 :162,000,134,073,032,140,203
40116 :157,165,165,201,000,208,052
40122 :007,162,002,134,073,032,084
40128 :140,157,032,115,000,240,108
```

```
40134 :003,032,253,174,032,178,102
40140 :157,165,095,166,096,133,248
40146 :122,134,123,032,215,170,238
40152 :208,011,200,152,024,101,144
40158 :122,133,122,144,002,230,207
40164 :123,032,202,159,240,005,221
40170 :032,220,157,176,003,076,130
40176 :143,156,132,085,230,085,047
40182 :164,085,166,049,165,050,157
40188 :133,139,177,122,240,216,255
40194 :221,000,002,208,237,232,134
40200 :200,198,139,208,241,136,106
40206 :132,011,132,151,165,073,166
40212 :240,091,032,240,157,165,177
40218 :052,056,229,050,133,167,201
40224 :240,040,200,240,202,177,107
40230 :122,208,249,024,152,101,126
40236 :167,201,002,144,064,201,055
40242 :075,176,060,165,167,016,197
40248 :002,198,139,024,101,011,019
40254 :133,151,176,005,032,036,083
40260 :158,240,003,032,012,158,159
40266 :165,151,056,229,052,168,127
40272 :200,165,052,240,015,133,117
40278 :140,166,051,189,000,002,122
40284 :145,122,232,200,198,140,105
40290 :208,245,024,165,045,101,118
40296 :167,133,045,165,046,101,249
40302 :139,133,046,165,122,166,113
40308 :123,133,095,134,096,166,095
40314 :067,165,068,032,061,158,161
40320 :032,225,255,169,000,133,174
40326 :198,164,151,076,242,156,097
40332 :164,122,200,148,049,169,224
40338 :000,149,050,185,000,002,020
40344 :240,021,197,139,240,005,226
40350 :246,050,200,208,242,132,212
40356 :122,096,201,171,240,004,230
40362 :201,045,208,001,096,076,029
40368 :008,175,144,005,240,003,239
40374 :032,166,157,032,107,169,077
40380 :032,019,166,032,121,000,046
40386 :240,011,032,166,157,032,064
40392 :115,000,032,107,169,208,063
40398 :224,165,020,005,021,208,081
40404 :006,169,255,133,020,133,160
40410 :021,096,032,202,159,133,093
40416 :067,032,202,159,133,068,117
40422 :056,165,020,229,067,165,164
40428 :021,229,068,096,165,122,169
40434 :133,034,165,123,133,035,097
40440 :165,045,133,036,165,046,070
40446 :133,037,096,165,034,197,148
40452 :036,208,004,165,035,197,137
40458 :037,096,164,011,200,177,183
40464 :034,164,151,200,145,034,232
40470 :032,001,158,208,001,096,006
40476 :230,034,208,236,230,035,233
40482 :208,232,164,011,177,036,094
40488 :164,151,145,036,032,001,057
40494 :158,208,001,096,165,036,198
40500 :208,002,198,037,198,036,219
40506 :076,036,158,160,000,132,108
40512 :165,132,015,032,205,189,034
40518 :169,032,164,165,041,127,000
40524 :032,210,255,201,034,208,248
40530 :006,165,015,073,255,133,217
40536 :015,200,177,095,240,025,072
40542 :016,236,201,255,240,232,250
40548 :036,015,048,228,132,165,212
40554 :032,124,158,200,177,174,203
40560 :048,214,032,210,255,208,055
40566 :246,032,215,170,056,096,165
40572 :160,157,132,174,160,160,043
40578 :132,175,056,233,127,170,255
40584 :160,000,202,240,238,230,182
40590 :174,208,002,230,175,177,084
40596 :174,016,246,048,241,032,137
40602 :107,169,165,020,133,053,033
40608 :165,021,133,054,032,253,050
40614 :174,032,107,169,165,020,065
40620 :133,051,165,021,133,052,215
40626 :032,142,166,032,202,159,143
40632 :032,202,159,208,033,032,082
40638 :172,159,032,202,159,032,178
40644 :202,159,208,003,076,143,219
40650 :156,032,202,159,165,099,247
40656 :145,122,032,202,159,165,009
40662 :098,145,122,032,183,159,185
40668 :240,226,032,202,159,032,087
40674 :202,159,032,202,159,201,157
40680 :034,208,011,032,202,159,110
40686 :240,197,201,034,208,247,085
40692 :240,238,170,240,188,016,056
40698 :233,162,004,221,212,159,217
40704 :240,005,202,208,248,240,119
40710 :221,165,122,133,059,165,103
40716 :123,133,060,032,115,000,219
40722 :176,211,032,107,169,032,233
40728 :081,159,165,060,133,123,233
40734 :165,059,133,122,160,000,157
40740 :162,000,189,000,001,201,077
40746 :048,144,017,072,032,115,214
40752 :000,144,003,032,130,159,004
40758 :104,160,000,145,122,232,049
40764 :208,232,032,115,000,176,055
40770 :008,032,145,159,032,121,051
40776 :000,144,248,201,044,240,181
40782 :184,208,150,032,172,159,215
40788 :032,202,159,032,202,159,102
40794 :208,008,169,255,133,099,194
40800 :133,098,048,014,032,202,111
40806 :159,197,020,208,015,032,221
40812 :202,159,197,021,208,011,138
40818 :032,209,189,169,032,076,053
40824 :210,255,032,202,159,032,242
40830 :183,159,240,210,032,162,088
40836 :159,230,151,032,036,158,130
40842 :230,045,208,002,230,046,131
40848 :096,032,162,159,198,151,174
40854 :032,012,158,165,045,208,002
40860 :002,198,046,198,045,096,229
40866 :032,240,157,160,000,132,115
40872 :011,132,151,096,165,053,008
40878 :133,099,165,054,133,098,088
40884 :076,142,166,165,099,024,084
40890 :101,051,133,099,165,098,065
40896 :101,052,133,098,032,202,042
40902 :159,208,251,096,160,000,048
40908 :230,122,208,002,230,123,095
40914 :177,122,096,137,138,141,253
40920 :167,067,072,065,078,071,224
40926 :197,068,069,076,069,084,017
40932 :197,070,073,078,196,075,149
40938 :073,076,204,078,085,077,059
40944 :066,069,210,000,165,156,138
40950 :065,156,165,156,198,155,117
40956 :152,158,172,013,013,013,005
```

PRODUCT MART

VIC & 64

AT LAST! The information you need, without always going 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[1]	☐	☐	Graphic printer (1515 & 1525)[1]
☐		Vicmon[1]	☐		UMI Wordcraft 20[2]
☐		Super Expander[1]	☐		HES Vic Forth[1]
☐		Vic Typewriter[1]	☐	☐	HES Writer[3]
☐		Victerm 1[1]		☐	Wordpro 3 plus
	☐	Term 64[1]		☐	Easy Script[1]
☐	☐	Quick Brown Fox	☐	☐	Basic

CG184

Send check or money order plus $ 1.00 (postage and handling)
PA residents add 6% sales tax.

Name_____

Address_____

City_____ State_____ Zip_____

[1] Products of Commodore Business Machines, Inc. [2] Product of United Microware Industries, Inc.
[3] Products of Human Engineered Software. VICMON is a Trademark of Commodore Business Machines, Inc.

CHEATSHEET PRODUCTS™
P.O. Box 8299 Pittsburgh PA. 15218 (412) 456-7420

FREE CATALOG !
HOME, EDUCATIONAL, AND BUSINESS SOFTWARE FOR THE VIC AND 64

New Items...
Checkbook/64 (Disk) Handles all checking account data. $16.00
Capitals/64 Teaches U.S. and Foreign capitals. Disk $10.00
Tape $8.00

Over 50 other titles!
Low-Priced! Practical!
Put your VIC or 64 to work with quality software from

Farthest Fringe S.A.
101 Highway Blvd.
N. Pekin, IL 61554

VIC 20™/COMMODORE 64™

Ridge Runner — for the unexpanded VIC 20 on tape or disk format. 100% machine language. Maneuver your multicolor U.F.O. through a field of blinking mines while blasting spinning asteroids and various enemy ships with laser fire. Features include a horizontally scrolling playfield, hi-resolution/multicolor graphics, excellent sound, high score, pause button, bonus ships, and ever increasing levels of difficulty. Joystick required. Price: $14.95

Dungeons — for the VIC 20 with 16K expansion and tape or disk. Create characters to explore a twelve level dungeon which contains 1200 individual rooms. After you purchase your weapon and armor, you will find vast treasures and do battle with over fifty types of monsters which you must slay for experience points. Your character also has the ability to cast numerous spells and you are given the option of saving the game to tape or disk as your character gains strength and experience. Excellent sound and three dimensional graphics add to the excitement. Price: $14.95

Pak Alien — for the unexpanded VIC 20 with tape or disk. 100% machine language arcade-style game. Custom graphic characters and 100 levels of increasing difficulty. Guide your alien through a maze of interplanetary space particles dodging the seven evil aliens and clear the board before the bonus timer runs out. Joystick or keyboard. Includes pause feature. Price: $14.95

BYTES and BITS
524 E. Canterbury Ln.
Phoenix, AZ 85022

(602) 942-1475
Please specify tape or disk
Add $2.00 for postage and handling

VIC 20 and Commodore 64 are trademarks of Commodore Electronics Ltd.

for Commodore 64™ and VIC 20™ Creative and hilarious FUN HOUSE

- Great as a gift and for having fun at parties
- Has color graphics fun sounds & accepts words
- Find special keys for extra fun

On cassette. Just $22. Get 2 of with the word "GAZETTE" in your order. Specify Computer. Send check or money order to:

CONTINENTAL CONCEPTS
P.O.Box 7631 NEWPORT BEACH, CA 92660

CA add sales Tax. Postage extra if fast delivery desired. VIC 20 version uses 16K expansion. Order Today!

"VIC 20" and "Commodore 64" are trademarks of Commodore Business Machines

'PUBLIC DOMAIN'™ — SOFTWARE —

Supporting all COMMODORE computers
Written by users, for users
★ GAMES ★ UTILITIES ★ EDUCATIONAL ★

VIC 20™
collection #1 – collection #2 – collection #3
collection #4 – collection #5 – collection #6
70+ programs per collection – Tape/Disk – $10.00

COMMODORE 64™
64 collection #1 – 64 collection #2 – 64 collection #3
64 collection #4 – 64 collection #5
25+ programs per collection – Tape/Disk – $10.00

PET® / CBM®
5 Utility – Tapes/Disks – $10.00 each
11 Game – Tapes/Disks – $10.00 each
6 Educational – Tapes/Disks – $10.00 each

DINSET™: Reset Switch
Works on Vic 20 or Commodore 64 – $5.00

All prices include shipping and handling.
CHECK, MONEY ORDERS,
VISA and MASTERCARD accepted.

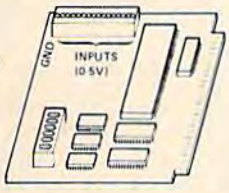
For A Free Catalog Write:

Public Domain, Inc.
5025 S. Rangeline Rd., W. Milton, OH 45383
10:00 a.m. - 5:00 p.m. EST – Mon. thru Fri.
(513) 698-5638 or (513) 339-1725

VIC 20™, CBM™ and Commodore 64™ are Trademarks of Commodore Electronics Ltd.
PET® is a Registered Trademark of Commodore Business Machines, Inc.

MULTIPLEXED DIGITAL VOLTMETER FOR THE VIC-20 AND C-64

INTRODUCTORY OFFER
$79
Expires April 84



FEATURES
- 16 channel Analog to Digital converter (0-5V range)
- High accuracy (1/2% of full scale)
- Simply uses BASIC statements (one POKE and PEEK)
- Compatible with other I/O devices (switchable address)
- Can be used in both VIC-20 and C-64
- Can be daisy chained together for up to 256 channels

Great for engineering experiments, Science fair projects, Household monitoring of temperatures, light switches, smoke detectors, alarm systems, or invent your own uses.

Send check or money order
DRIVER ELECTRONICS
194-B Gladys Ave.
Mountain View, CA 94043

California residents add 6.5% sales tax.
Send self-addressed envelope for free brochure.

VIC 20 C64

1983 TAX RETURN HELPER

Fast and easy
income tax preparation.

- Form 1040 and Schedules A,B,C,D,E
- Enter and modify data on a screen copy of the form.
- Works like a spreadsheet - all the lines affected by a change are instantly updated.
- Form 1040 and Schedule A are automatically corelated.
- Price is **tax deductible.** Cassette $23 (+ $1.50 shipping & handling). Specify VIC 20 (16K RAM) or C64.
- Check, MO or credit card.

KSOFT CO.
845 WELLNER RD.
NAPERVILLE, IL 60540
(312) 961-1250

Dealer inquiries welcome

VIC-20 USERS

Cartridge Back-Up

- SYSTEM IS AN EASY TO USE PROGRAM AND A HIGH QUALITY CARTRIDGE INTERFACE BOARD
- BACK-UP YOUR CARTRIDGES ONTO CASSETTE OR DISK
- 8K RAM REQUIRED

$49.95 POST PAID

Cassette Back-Up

- EASY TO USE PROGRAMS
- BACK-UP ANY CASSETTE TAPE ONTO TAPE OR DISK
- REQUIRES **NO USER MEMORY**

$19.95 POST PAID

VISA/MASTERCARD ORDERS:
PHONE (215) 269-4803

MAIL CHECK OR MONEY ORDER TO:
E-M TECHNOLOGIES
P.O. BOX 185
DOWNINGTOWN, PA. 19335

PA. RESIDENTS ADD 6%
6 MONTH REPLACEMENT GUARANTEE

LOW COST SOFTWARE

Are you tired of paying high prices for your software? Let John Henry Software save you money! We distribute public domain software for your **Commodore 64™** or **VIC 20™**. We've tested and documented each program to guarantee you hours of fun and useful learning experiences. We specialize in prompt delivery of your software, even if you order tapes, and we guarantee our product.

You'll also receive our free program reference book when you place your order.

COMMODORE 64

Group CG	26 Games for Everyone	$7.95
Group CP	30 Programming, Demo, Business and Home	$7.95
Group CE	16 Educational Programs	$7.95
Group CA	5 Adventure Games	$7.95

VIC 20

Group VG	62 Games for Everyone	$7.95
Group VP	54 Programming, Demo, Business and Home	$7.95
Group VE	35 Educational Programs	$7.95

When ordering, specify group & tape or disk. Group CA is on disk only. Send check or money order to:

John Henry Software
P.O. Box 39021
Cincinnati, Ohio 45239

Don't wait! Order your software today! Or write for your free program reference book. You'll be glad you did!

VIC 20 and Commodore 64 are trademarks of Commodore Electronics Limited.

MAKE YOUR OWN CARTRIDGES!

- Complete System • Programs 8K Cartridge to "AUTO-RUN" (opt) at Power-up — YOUR BASIC or Machine-Code Prgm.
- Prototype GAMES
- EDUCATIONAL use — NO Load Delay
- OS-3721 plugs into VIC-20
- Expansion socket on board
- Carts. Erasable & BLK Locatable

OS-3723 ZIF socket module Allows programming 2764, 2732A, 2732, 2716, 2564, 2532, 2516 EPROMS	$24.50
OS-37A System (Pgmr., 8K Cart)	$88.50
OS-37B System (Pgmr., ZIF mod) (Software Tape incl. Disk add $2)	$88.50
OS-3722 Blank 8K Carts.	$24.50
Blank 2764 EPROMS	$11.50

OTTO SYSTEMS
8135 ENGINEER ROAD
SAN DIEGO, CA 92111
(619) 569-5665

Add $4 Shipg. & Hndlg. — CA res. 6% tax

FREE OFFER!

COMPUTER CASSETTES 58¢

FREE
"World Capitals Game" with each order of 20 or more C-10's

Specify VIC-20 or Commodore 64

- C-10 Length
- 5 Screw Shell/Free Labels
- Lifetime money back guarantee
- Storage Box add 12¢ each
- $2.00 shipping charge — any quantity (Canadian orders $4.00 shipping)
- NJ Residents add 6% sales tax
- Send check or money order to

PARALLEL SYSTEMS
Box 772
Blackwood, NJ 08012
609-227-9634

OTOG USERS GROUP

Join One Of The Largest Active Users Groups. You Will Get...

__ Access To Hundreds of Public Domain Programs For Your Commodore 64, vic 20, and PET/CBM
__ A Monthly News Letter
__ And A Lot More...

For more information send $1.00 to:

OTOG USERS GROUP
321 Alma Street
Kissimmee, Fl. 32741

Membership Fees for 12 months

USA—$20.00
Canada—$20.00
Over Seas—$20.00

TRADE SOFTWARE DIRECTLY WITH OTHER PEOPLE

SofTraders' Catalog
SAVES MONEY

★ *Select Traders*
★ *List Your Software*

Send **$1.00** for **Catalog**
• Deduct cost from next order

SofTraders™
INTERNATIONAL
1610 Shomaker Drive
Murphysboro, IL 62966

VIC 20 COMMODORE 64

UMI / VIC 20

Spiders of Mars [C]	$29.95
Meteor Run [C]	$29.95
Amok [T]	$16.95
Sat & Met [C]	$29.95

UMI / New for C-64

Pennant Drive 2 player baseball strategy	$29.95
Motor Mania hi-performance racing game	$29.95
Fuego fight fire-throwing drones to save the space crew	$29.95

TOTL SOFTWARE

TOTL Text 2.6 [D]	$34.00
TOTL Label 2.6 [D]	$20.00

Send cash, check or money order to:
ARIES MARKETING CO.
P.O. Box 4196
4200 Shannon Drive
Baltimore, Maryland 21205
Md. residents add 5% state sales tax

TREASURE

RAIDERS

ARK Innovations, Inc. for VIC 20™

$2.97
ARK INNOVATIONS, INC.
18133 SCHOOL ST.
P. O. BOX 155
AMADOR CITY, CA. 95601

VIC 20 is a trademark of Commodore Electronics Ltd.

cassette uses joystick, 3.5 K

NEW! TAXI DRIVER
reading development: inquire

ACTION MATH

FOR COMMODORE 64

Develop your children's math skills the fun and easy way.
Perfect for kids preschool through 6th grade.

MENU SELECTIONS INCLUDE:
- Addition
- Multiplication
- Fractions
- Subtraction
- Number Spelling
- Division
- A skill level setting for all the above

OTHER FEATURES INCLUDE:
- Music
- Color Graphics
- Large Character Set
- User Defined Pacing Values—lets you determine when to auto-advance to the next level. Keeps it challenging!
- For added fun, play against the clock with a time value that you can select!

SCHOOL DISCOUNTS AVAILABLE Prices good thru 12/31/83

SEND CHECK OR MONEY ORDER FOR
$19.95 (CASSETTE) **$21.95** (DISKETTE) TO
Educational Software Products
P.O. Box 620211 • Littleton, Colorado 80123

KRAFT Commodore® COMPATIBLE
JOYSTICK

SPECIAL NEW PRICE
REDUCED TO $ 9.95

Kraft Commodore® Compatible Joysticks for Atari VCS, Atari 400/800/1200 XL, Atari 2600, Commodore VIC-20, Commodore 64, Sears Video Arcade, NEC 6001 and Panasonic JR200. ® All of the above are tradenames or registered trademarks.

The June/July 83 issue of Video Magazine stated that this outstanding joystick was rated number 1 in actual game performance over 2nd place Spectravideo, 3rd place Wico, and 4th place Discwasher.

Now, you can purchase this game winning KRAFT COMMODORE COMPATIBLE JOYSTICK for only $9.95. Originally introduced and sold nationwide for $16.95. If you are a "left", the "SWITCHHITTER" has also been reduced to $12.95. Order several now and give your family and friends this perfect computer gift. Price includes Kraft's full 1 year limited warranty.

KRAFT COMMODORE STICK	$ 9.95
KRAFT SWITCHHITTER	$12.95

Include with your check or money order, $2.00 per stick for handling and shipping charges. California residents add 6% sales tax. Please allow 3 weeks for delivery. Sorry, no COD's.

C R CONCEPTS INC
104 CALLE CUERVO
SAN CLEMENTE, CA 92672

EXPAND YOUR VIC 20™ MEMORY!

Affordable Memory Expander lets you add 2K RAM Circuits as your needs increase. (Up to 35K.)

Mother Board with Instructions	$39.95
Mother Board & Sockets with Instructions	$54.95
Complete Kit with Cabinet (35K)	$159.95
Assembled & Tested Expander (35K)	$189.95
Each Add'l 2K Chip (Up to 280K paged RAM) [$1 Shipping]	$8.95

Send Check or Money Order to

PERIPHERAL DEVELOPMENT
P. O. Box 28247
St. Louis, MO 63132

(Add $5 S. H. Missouri residence add 5½% tax)
Prices subject to change without notice

Vic 20 is a trademark of Commodore Electronics, Ltd.

Soft Cellars

PRESENTS

GAMES AND UTILITIES FOR THE VIC & 64
on cassette or disk

Digital Derby—Pari-mutuel betting with galloping graphics and sound.
Unex. Vic C64 ..... $14.95

High Risk—A possible mission fraught with audio visual & mental gymnastics. Joystick required.
Unex. Vic only $19.95

Super Cipher—Decipher color or symbol codes. Select length and time. Infinite levels of difficulty. 1 or 2 players.
Unex. Vic C64 ..... $12.95

Program Cellar—Pixel-by-pixel movement techniques in BASIC. Auto renumber delete. Easy entry BASIC program lines. Sub-routine library.
Vic C64 ..... $14.95

Data Cellar—Over (68) records per disk. Random access. Menu prompted. Alpha numeric sorts. Easily tailored to your needs.
C64 only. Disk required $19.95

SEND CHECK OR MONEY ORDER TO:
SOFT CELLARS, INC.
828 RUE ROYAL SUITE 535
NEW ORLEANS, LA. 70116

ADD $2.00 for disk version

Vic, 64 are Commodore trademarks

The Banner Machine™

For the **Commodore 64** (4 extra fonts available). For the **VIC-20** with 24K memory (3 extra fonts available). • Use on any Gemini or Epson MX with Graftrax or the FX and RX printers. Also Commodore 1525E and Banana with the C-64. • Menu-driven program operates like a word processor. • Makes signs up to 13" tall by any length. • Makes borders of widths up to ¾". • 8 sizes of letters from ¾" to 8" high. • Proportional spacing; Automatic centering; Right and left justifying. • $49.95 Tape or Disk (Specify computer equipment)

For the Commodore 64:

Space Raider An amazing arcade simulation. Your mission is to destroy the enemy ships. $19.95

Super Roller Challenging dice game. Sprite graphics and sound. Yahtzee-style rules of play. $14.95

Microbroker Exciting, realistic and educational stock market simulation. $34.95 Tape or Disk

Preschool Educational Programs ABC Fun; 123 Fun; and Ginger the Cat with: Addition and Subtraction, Number Hunt, and Letter Hunt. All programs have bright color, music, and action. Each $14.95

Formulator A scientific calculator for tasks which require repetitive arithmetic computations. Save formulas and numeric expressions. $39.95

Sprite Editor The easy way to create, copy, alter, and save up to 224 sprite shapes. $24.95

Cross Reference Generator for BASIC programs Locates lines with BASIC words or variable names and allows changes, and more. $19.95

VIC-20 Programs Also Available. Ask for Catalog.

Cardinal Software™
Virginia Micro Systems
13646 Jeff Davis Highway
Woodbridge, Virginia 22191
Phone (703) 491-6502

Commodore 64 and VIC-20 are registered trademarks of Commodore Electronics Ltd.

R.I.S.T. Inc. ANNOUNCES

SPEECH SYNTHESIZER

ORDER NOW **SALE** ORDER NOW

ACT NOW Receive

$59.95 plus $4.00 sh/hd

FREE
Phrase Finder Program
Speech Synthesis Manual

TRS80 Models I, III
Apple II, II+
ZX80/81 TS1000
Com 64, VIC 20

HEAR THE POWER...

Regularly $89.95
Now Only $59.95

Complete or Starter Kits Available

**** 10 Day Money-Back Guarantee ****

Send check or M.O. COD add $1.50 to R.I.S.T. Inc., Dept. GZ 8, P.O. Box 499 Ft. Hamilton Station, Brooklyn, NY 11209 212-259-4934
N.Y.S. RESIDENTS ADD 8¼% TAX

Advertise your program or product for the VIC-20 or 64 here and reach hundreds of thousands of readers.

MEMORY FOR YOUR VIC-20 PLUS A LITTLE EXTRA

32K Dynamic Ram - Same type as Commodore 64
24K Used normally, for full expansion. Blocks 1, 2, & 3
"EXTRA" 8K maps into block 5, normally used for ROM cartridges (Games)
Accesses from BASIC for data storage, only through PEEK and POKE
Accesses in assembler for data, or write your own games, & boot into them normally
The 3K "Block 0" ram has been intentionally left free for the VIC Superexpander
Each 8K block can be switched in or out for compatability with games, or other peripherals.
Powered by your VIC-20 (175 ma typ)
Plugs directly into VIC expansion slot, or motherboard
Complete, assembled, and tested
90 day "No questions asked" money back return
5 year manufacturer's warranty

$109.95

TEMPUS
Dept. G101 832 Brown Thrush, Wichita, KS 67212
Handling charges, add $3.00
Personal checks take 3 weeks to clear, before we ship.
We carry the entire XETEC line for VIC-20 and C64. Write for literature.
MasterCard & Visa — Send card number & expiration date.
VIC-20 & Commodore 64 are Commodore trademarks

2 + 1 EXPANDER

$24.95 (GOLD CONNECTORS)

2 FULL SERVICE EXPANSION CONNECTORS RIGHT ANGLE CONNECTOR: DAISY-CHAIN 2 + 1s TOGETHER
1 SPECIAL SWITCHED CONNECTOR FOR BLOCK ASSIGNMENTS
STORE SEVERAL PROGRAMS ON SAME CARTRIDGE
SCREEN PROMPTS
STORES BASIC OR MACHINE PROGRAMS
RE-USABLE CARTRIDGES

RAM BLOCK $29.95

8K MEMORY EXPANSION CARD. ASSEMBLED AND TESTED
BARE 8K CIRCUIT BOARD $9.95 FOR VIC 20

PLAY OVER THE PHONE OR LOCAL
VIC TO VIC OR C64
NO CONNECT CHARGES

CARTRIDGE PROGRAMMER

$79.95 USER FRIENDLY
WORKS WITH PROG.AID & V-MON
NO MEMORY EXPANSION REQUIRED
LOW COST

GAZETTE SPECIAL
7TH FLEET $29.95

SEND MESSAGES
SINK ENEMY SHIPS
SPY MODE
SET MINES

MIGHTY 5 $49.95

5 FULL SERVICE EXPANSION CONNECTORS
SOFTWARE CONTROL OF BLOCK ENABLE LINES
DUAL RESET SWITCHES: RESET VIC OR MIGHTY 5
(GOLD CONNECTORS) USES SEVERAL GAMES OR RAM CARDS AT ONCE

These and other fine products available from:

QUEUE PRODUCTS
(313) 846-6666
BOX 1655 / DEARBORN, MI 48121
— DEALER INQUIRY REQUESTED —

C-64

PCI'S COMPLETE PRACTICAL GUIDE TO THE COMMODORE 64

Where the C-64's owner's manual falls down or leaves off, PCI's Guide to the C-64 keeps you going. Over 200 pages of practical information to help you get the most out of your Commodore.

The Guide gives special attention to graphics and sound capabilities and provides practical, in-depth discussions of programs specifically developed for the Guide.

SPECIAL OFFER: To get your copy early and save $2 off the retail price, send check or money order for $12.95 before January 15 to:

PERSONAL COMPUTERS, INC.
4533 BAILEY AVENUE
AMHERST, NY 14226

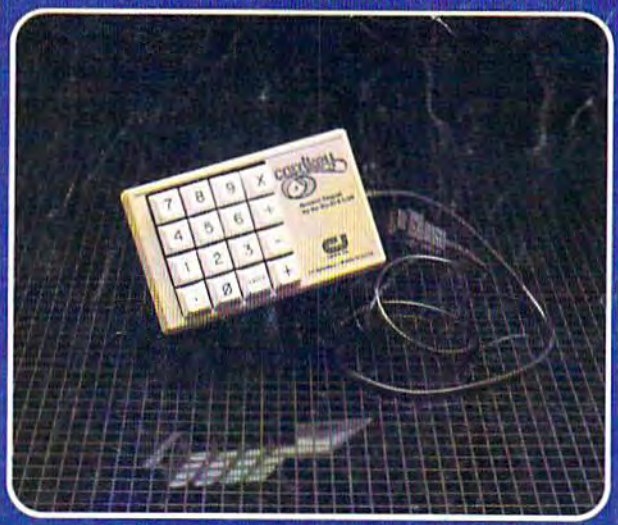
Dealer inquiries call: (716) 832-8800

ADVERTISERS INDEX

Cardco Presents

16 Key Keypad for the VIC-20™ and Commodore 64™

The CARDKEY/1 provides an additional programable 16 keys to either the VIC or 64. This is a separate pad that plugs into the joystick port of your computer. It comes with tape programs (transferable to disk) that allow the user to define the 16 keys as any number, letter or character string value. It additionally allows the keys to be defined as strings. Another function available is the direct calculator mode in which you use your computer as a calculator. For example you press 5 + 5 (enter) and the computer prints 10 on the screen.



The CARDKEY features:

- 16 keys laid out in a calculator format
- All keys are user definable
- Keys can be defined as any character
- Keys can be defined as complete strings
- Optional calculator mode

Suggested retail: $39.95

See us at
COMDEX™
Fall '83
November 28-December 2, 1983
Las Vegas Convention Center
Las Vegas, Nevada

See a complete line of American made Cardco Products at a computer store near you, today.

313 Mathewson • Wichita, Kansas 67214 • (316) 267-6525

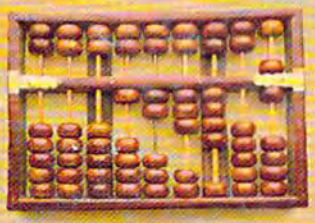
cardco, inc

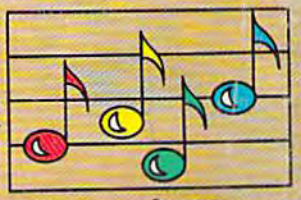
It writes, rates, creates, even telecommunicates. Costs less, does more- the Commodore 64.

When Commodore introduced the 64, the industry suddenly realized that there would be a computer in every home, school and business years before anyone ever dreamed.

That's because Commodore 64 halved the price of high technology: while you can compare the 64's capabilities with those of any sophisticated business PC, you can compare its price with that of an average television.

What can you do with it? Create with its high resolution Sprite Graphics. Add a printer and type with it. Add a disk drive to use spread sheets and other financial programs. Learn and play music through your home sound system on the 64's professional quality music synthesizer.





Add a modem, and hook up with the vast computer networks through your telephone. In short, the Commodore 64 is the ultimate personal computer, at a price you can afford.

COMMODORE 64