PROOF OF PURCHASE

Software Giveaway with Purchase of this Magazine
See back cover

# HOME COMPUTER™ magazine

FOCUSING EXCLUSIVELY ON ● APPLE ● COMMODORE ● IBM ● TEXAS INSTRUMENTS

Vol. 5 No. 3

$3.50 in USA
$4.50 in Canada

## Computerized Budgeting

—A Mighty Ally Helps Slice The Pie

Contains 33 Type-In Programs!

**Featuring A Ready-To-Use Budget Processor for:**

- Visualizing Spending Trends
- Monitoring Expenses
- Planning Purchases
- Boosting Savings

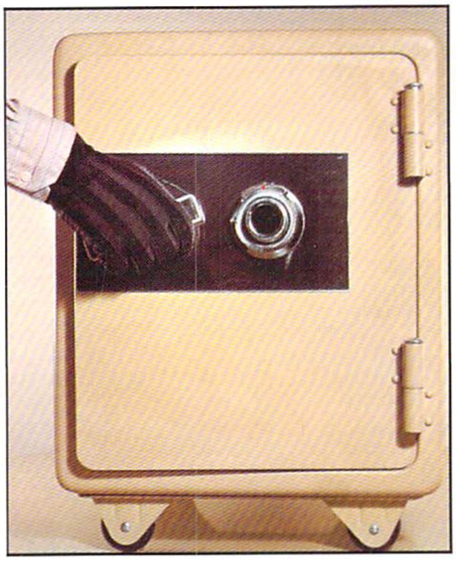
**Special Apple, Commodore, IBM, & TI Software:**

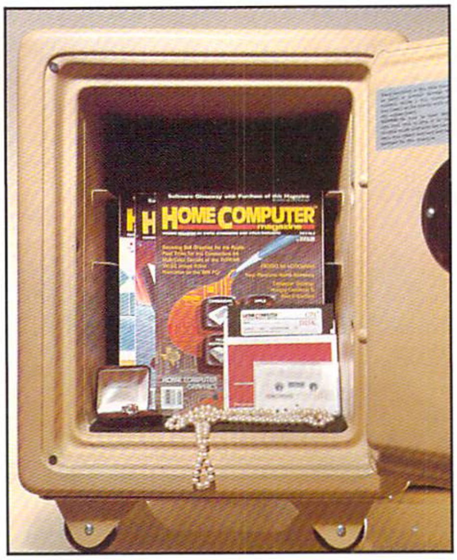
- Honing Geometry Skills

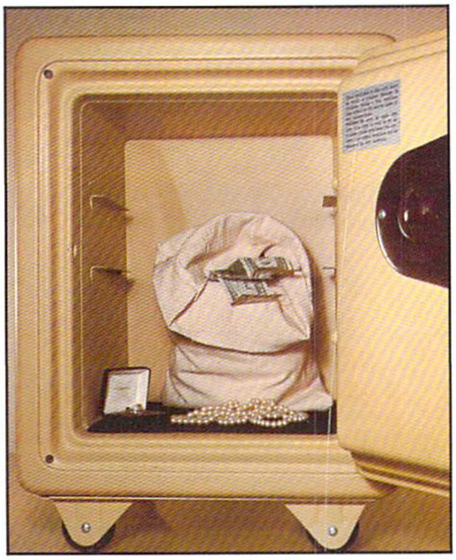
uring
Plant
on
ne

**—Reviews Galore:**

★ Artist & Programmer Tools
★ Typing Tutors & Word Processing
★ Computerized Speech & Scrabble
★ Music & MIDI Link-Up to Home Computing



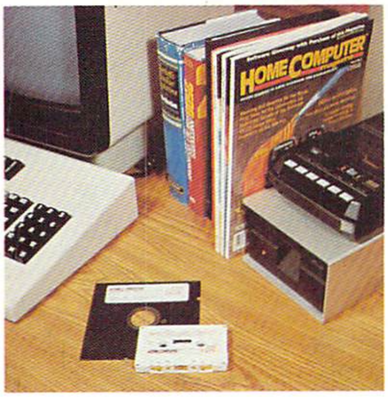




# MISSING ANY VALUABLES?

If you're missing any back issues of HOME COMPUTER™ magazine
you're missing more than you'll ever know . . .

Having each issue of *Home Computer Magazine* readily at hand provides you with direct access to a valuable reference library of home computer knowledge —unequaled anywhere!



A valuable reference library of each *Home Computer Magazine* issue is the One Essential Peripheral™ for your home computer.

Back issues of HCM's program service— ON DISK™ or ON TAPE™ are also available.

ON DISK™ and ON TAPE™ are the convenient, accurate and affordable ways to save hundreds of typing hours.

Collect all the programs from each magazine issue on a ready-to-RUN quality floppy disk or cassette tape available in separate versions for Apple, Commodore, IBM, and Texas Instruments home computers.

## "Safeguard" Your Home Computer Knowledge— Order Valuable Back Issues Today!

To Order, Use Bind-In Card at Center of Magazine.

# HCM BACK ISSUES

. . . and discover what's in store for you—an extraordinary resource value!

**VERSIONS SUPPORTED:**

| Machine | Media |
|---|---|
| APPLE *II* Family (A) | ON DISK™ |
| COMMODORE 64 (C) | ON DISK™/ON TAPE™ |
| IBM PC/PCjr (I ) | ON DISK™ |
| TI-99/4A (T) | ON DISK™/ON TAPE™ |

* = No ON TAPE™ available, even if normally supported
TX = Extended BASIC programs only
PCjr = Available for PCjr only

Apple Owners: Please note that ON DISK™ Media for *HCM* 4.1-4.3 is in DOS 3.3 format only, and all Apple programs beginning with *HCM* 4.4 are in ProDOS format. All programs will RUN on a 64K Apple II + (with Applesoft BASIC in ROM), an Apple *IIe*, or an Apple *IIc*.

Apple & IBM "clone" Owners: Please note that some *HCM* programs may not RUN on your machines, because of differences in hardware and/or BASIC interpreters.

## Issue 4.1:

Premier Issue * Uncle Larry's Fiddle Tunes * Electronic Sheet Music * Music in Mini Memory * PCjr: A Look Inside the Peanut's Shell * 66 Keys to Graphics Success: A Primer for the Commodore 64 * Have No Fear: Assembly Language Won't Byte, Part 3 * Porsches and other Pipedreams: Computer Assisted Savings * 3DIIe: Apple Graphics in Three Dimensions, Part 1 * Biting Into Your Apple * Don't Be A SlowPOKE * Down Memory Lane: Don't let programmable characters gobble up your memory * Easy As Pie: Apple programming for intricate works of art * Microcomputer Accuracy * What is LOGO? * Lyrical LOGO * LOGO Shoots for the Moon: A lesson in structured problem-solving * Product Reviews * Flak Attack * Slots * Meltdown * Challenging the Tower of Hanoi * HCM TECH NOTES: Apple, C-64, IBM, and 99/4A * Product News * Group Grapevine, and much, much more!

**CONTENTS: ON TAPE™ & ON DISK™**

**Flak Attack** (A,C,I,T)
Air-to-ground battle game
**Applesoft 3D** (A)
Apple graphics in three dimensions
**Tower of Hanoi** (A,C,I,T)
An ancient brain teaser
**Saving** (A,C,I )
Computer-assisted savings plan
**LOGO Poet** (A,C*, I )
Recursion frees the poet in your console
**LOGO Apollo** (C*, T)
A lesson in structured problem-solving
**Slots** (T)
An intriguing Las Vegas simulation
**Uncle Larry's Fiddle Tunes** (C, I, T)
Play ten beloved fiddle tunes
**Music Magic** (TX)
"Joy to the World" in harmonious BASIC
**Music Assembler** (T)
Assembly language simplifies composition
**Autosprite** (C)
Routines to keep your graphics lively
**Meltdown** (TX)
Debug the reactor and save the world

## Issue 4.2:

Graphics * Sea of States * San Francisco Tourist * Building Your Character: A Graphics Editor for the VIC-20 * Quick Pixel Tricks: A Graphics Editor for the C-64 * Follow the Bouncing Ball: On the rebound with graphics fundamentals * 3DIIe: Apple Graphics in Three Dimensions, Part 2 * Double Your Color, Double Your Fun: Sprites try on a layered look * Musical Mystery Words * Matrix Muncher * Elementary Addition and Subtraction for the VIC-20 * IBM Animation: Controlling the pallet on the PCjr * Jr. Sounds Off: Access Jr's Special Sound Enhancements * The Electronic Home Secretary * Files in LOGO * LOGO Spans the Generation Gap: A review of Commodore LOGO * FROGO: LOGO Invades the Arcade * Product Reviews * Tablut * Cannibals * HCM TECH NOTES: Apple, C-64, IBM, and 99/4A * Product News * Group Grapevine, and much, much more!

**CONTENTS: ON TAPE™ & ON DISK™**

**Cannibals** (A, C, I, T)
Livingston Stew, I presume?
**FROGO** (T)
A logical LOGO learning lesson
**The Home Secretary** (A, C, I, T)
Address & inventory recordkeeping
**LOGOFILES** (A, C*, I, T)
Access your DATA files in LOGO
**Sea of States** (C, TX)
State Capitals and dive for booty
**Tablut** (C, I, TX)
14th-century strategy revisited
**Matrix Muncher** (C)
Solve unknowns simultaneously
**Graphic Editor** (C)
Pixel tricks create easeful graphics
**Mystery Words** (A, I )
Reading between the treble clefs
**PCjr Animation** (PCjr)
Exploring Junior's graphic modes
**Applesoft 3-D IIe** (A)
Edit your 3-D graphic shapes

## Issue 4.3:

Productivity * Snap-Calc: A Homespun ready-to-use spreadsheet * Bars and Plots: Create colorful graphic charts of your records * Elementary Addition and Subtraction for the 99/4A and C-64: A powerful children's learning tool * Spider Graphics: Spin a colorful web on screen * Convertible for Comfort: Automatically convert your machine-language programs to DATA statements * Programming: The Name of the Game: Designing your own game—a complete tutorial * Colorfun on your VIC-20 * Product Reviews * Binary Forest: Branching out with LOGO * LOGO Flakes: Creative explorations with snowflake designs * Robochase * Cyber-Cipher * Wild Kingdom * Speeder * Boolean Brain * Missile Math * HCM TECH NOTES: Apple, C-64, IBM, and 99/4A * Product News * Group Grapevine, and much, much more!

**CONTENTS: ON TAPE™ & ON DISK™**

**Snap-Calc** (A, C, I, TX)
Home sweet spreadsheet
**Robochase** (A, C, I )
Run from the rampaging robots
**Spider Graphics** (A, I )
Spin a myriad of rainbow filaments
**Boolean Brain** (A, I )
A graphic Adventure inside computers
**Wild Kingdom** (A, C, I, TX)
Flee ferocious felines
**Missile Math** (A, C, I, T)
Launching grade-school arithmetic
**Binary Forest** (A, C*, I )
Branching-out with leafy LOGO trees
**Bars & Plots** (T)
Color your chart factfully
**Cyber-Cipher** (T)
Decode correct color combinations
**Elem. Addition & Subtraction** (C, T)
BASIC preschool arithmetic skill-builder
**LOGO Flakes** (T)
Snowflakes in June? This must be LOGO
**Convertible for Comfort** (C)
Machine Language DATA auto-conversion

## Issue 4.4:

Computer Sports * IIc: The Core of a New Machine * On the Home Court: Computer Sports Simulation * Razzle Dazzle: Quick Graphics Magic for the 99/4A * Simon Sez: Plug in 114 new BASIC commands to the Commodore 64 * Tax Deduction Filer: A complete tax recordkeeping program convinces you that makes tracking of deductions a breeze * Kaleido Computer: Creating a myriad of mosaic designs on your home computer * Multiplan Medium, Part 8 * Have No Fear: Assembly Language Won't Byte, Part 4 * The RS-232 Interface: Understanding Your Link to the Periphery * One for the Money, Two for the Slow—Adding a Second Drive to the PCjr * Missionary Impossible: A Logic Puzzle in LOGO pits you against hungry Cannibals * Product Reviews * Boolean Brain * Stadium Jumping * Market Madness * Elementary Addition and Subtraction: An arithmetic tutor (for Apple and IBM PC and PCjr systems) * HCM TECH NOTES: Apple, C-64, IBM and 99/4A * Product News * Group Grapevine, and much, much more!

**CONTENTS: ON TAPE™ & ON DISK™**

**Boolean Brain** (C, TX)
A graphic Adventure inside computers
**Tax Deduction Filer** (A, C, I, TX)
SAVE-ing with your tax deductions
**Market Madness** (A, C, I, TX)
Exciting Stock market simulation
**Stadium Jumping** (A, C, I, T)
Horsing around an Olympic Stadium
**LOGO Spreadsheet** (A, C*, I, T)
And you thought LOGO was kidstuff
**Missionary Impossible** (A, C*, I, T)
Watch out for Cannibals with LOGO
**Elem. Addition & Subtraction** (A, I )
BASIC preschool arithmetic skill-builder

And There's More On The Next Page

## Issue 4.5:

Building Up Your Software Library * Quiz Construction Set: Create a Quiz or Take a Quiz—a must for students and teachers * Personal Loan Calculator: Find out where your interest lies * Jumping Ahead With Game Programming: A complete game programming tutorial includes a program example * Sketch-64: Joystick graphics with just a flick of the wrist * Simon Sez: New string-related commands explained * Razzle Dazzle: Character manipulation on the 99/4A * Division Tutor: Teaching BASIC math learning skills * Putting The Puzzle All Together: Apple IIc Programming Considerations * Bird Brain * Slither * LOGO Clones: TI Graphics In a Turtle-Shell * Build A LOGO Adventure, Part 1 * Product Reviews * HCM One Liners * HCM TECH NOTES: Apple, C-64, IBM, and 99/4A * Product News * Group Grapevine, and much, much more!

### CONTENTS: ON TAPE™ & ON DISK™

**Bird Brain** (A, C, I, T)
Keep your fishing feathers dry
**Division Tutor** (A, C, I, TX)
Expand elementary math skills
**Personal Loan Calc** (A, C, I, T,)
Find out where your interest lies
**Sketch-64** (C)
Use a joystick to draw graphics
**Quiz Construction Set:**
**Quiz-Make/Quiz-Take** (A, C, I, T)
Complete tutorial with file examples
**Peg Jump** (A, C, I, T)
Learn BASIC game programming
**Slither** (A, C, I, T)
A maze of snake-like proportions
**LOGO Clones** (T)
TI-Graphics in a Turtle-Shell
**LOGO Adventure** (A, I, C*)
Pt. 1: Creating interactive fiction

## Issue 5.1:

Thought Processing: A New Frontier in Home Computing * The Organizer: Store and organize your thoughts * Orbital Defender * Quiz-Print/Quiz-Print Tutorial: This educational enhancement is a tool for use with your Quiz Construction Set (see *HCM* 4.5) * Electronic Backgammon: A modern version of an ancient game of skill * Razzle Dazzle: Screen patterns with graphics characters on the 99/4A * Kors-Elf: An Arcade Typing-Tutor Game * Personal Loan Calculator: Find out where your interest lies * Apple Seedlings: A ProDOS Date-Setting Utility * IBMpressions: Create a beautiful pie chart * Build A LOGO Adventure, Part 2 * LOGO Sailing: A Premier Yachting Event * Simon Sez: Composing music is simple * HCM TECH NOTES: Apple, C-64, IBM, and 99/4A * Product News * Group Grapevine, and much, much more!

### CONTENTS: ON TAPE™ & ON DISK™

**Orbital Defender** (A, C, I, T)
Split-second battle decisions
**Electronic Backgammon** (A, C, I, TX)
Pit your pips against the computer
**Kors-Elf** (A, C, I, TX)
An arcade typing adventure
**The Organizer** (A, C*, I, TX*)
A versatile Thought Processor
**Quiz-Print** (A, C, I, T)
Format printouts of your quizzes
**Apple Seedling** (A)
BASIC utility dates ProDOS files
**LOGO Adventure** (A, C*, I)
Pt. 2: Creating interactive fiction
**Merging Files** (C)
Experienced hackers only!
**Personal Loan Calc** (T)
Find out where your interest lies
**Razzle Dazzle** (T)
*Wormwood* your character graphics
**LOGO Sailing** (T)
Turtles race for the America's Cup
**IBMpressions** (I)
Create a beautiful pie chart

## Issue 5.2:

Number Crunching: The Building Blocks of All Computing * It Figures: An equation calculator that'll crunch your numbers accurately * Evacu-Pod: See if you can rescue all the miners in this challenging space game * Switch 'n' Spell: Electronic anagram brain teasers to puzzle over (for children, and adults) * Laserithmetic: Strut your math skill with this space fantasy edu-game * Organizer Reports: An enchancement to print-out your organized thoughts (see The Organizer *HCM* 5.1) * Razzle Dazzle: Tinker with musical sounds, or Play It Maestro! * What is CP/M?: Learn the Basics of Control Programming for Microcomputers * Apple Seedlings: Sorting out your ProDOS Catalog * Commodore Hornblower: Discover what's inside the Commodore 64's SID chip * IBMpressions: Create 3-D surface drawings in BASIC * Field & Screen: A tutorial for using a Data Base System—correctly * Product Reviews * HCM One Liners * HCM TECH NOTES: Apple, C-64, IBM, and 99/4A * Product News, and much, much more!

### CONTENTS: ON TAPE™ & ON DISK™

**Evacu-Pod** (A, C, I, TX)
Miner rescue in space
**It Figures!** (A, C, I, TX)
A mighty equation calculator
**Laserithmetic** (A, C, I, T)
Blast aliens with your math skills
**Organizer Reports** (A, C*, I, TX*)
Print your organized outlines
**Switch 'n' Spell** (A, C, I, T)
A spelling aid that's fun to boot
**Apple Seedlings** (A)
Sort your ProDOS catalogs
**Commodore Hornblower** (C)
Inside the SID chip
**IBMpressions** (I)
3-D surface drawing in BASIC

**This space reserved for Issue 5.3**

**This space reserved for Issue 5.4**

**This space reserved for Issue 5.5**

# ◄ ◄ ◄ ◄ HCM BACK ISSUES

FOR NEW READERS

# The Plain & Simple Truth About HOME COMPUTER™ magazine

## Chock Full of Valuable Software & How-To Articles Without Filler

Every issue is a software "horn of plenty" with dozens of type-in-and-RUN programs printed in an easy-to-read listings format. Our programs are also available on inexpensive disks or cassettes for those who prefer the convenience of ready-to-RUN software. Step-by-step tutorials round out each issue, providing the solid facts you need without fluff or filler. Thus, each issue functions as an excellent reference work, as well as a valuable software source.

## No Outside Advertising

Freed from the pressures of servicing *advertisers*, we concentrate on serving our *readers*. Each issue provides uninterrupted editorial flow and graphic layouts for better comprehension—plus unbiased product reviews which focus on true strengths and weaknesses, wherever the chips may fall . . . And we don't have to worry about losing advertisers because of publishing software in the magazine that is "too good." Consequently, we can provide the best free software available anywhere.

## Focused on the 4 Hot Home Brands

We are 4 system-specific magazines under one wrapper—not a sprawling, "general interest" publication which attempts to cover too wide a field, only to spread itself too thin. The other side of the coin to this focused approach is the knowledge you gain from being exposed to the many tips, ideas, and techniques we provide for 3 of the 4 systems you may not even have. You'll learn more about your Apple, Commodore, IBM, or Texas Instruments home computer from this one magazine than from a host of more limited sources.

## A Balanced Mix For a Perfect Recipe

In each issue we strive for a perfect balance of productivity, entertainment, education, utilities, and computer literacy—serving the needs of novice and pro alike. Every issue is a full-course meal, with a smorgasboard of tasty dishes for all palates. Whereas other computer magazines may dish out lumps of "editorial indigestion," we serve up a satisfying blend—one digestible byte at a time.

**—Welcome to Our World of Home Computing**

HOME COMPUTER magazine™

**Home Computer Magazine** (ISSN 0747-055X) is published ten times per year by Emerald Valley Publishing Co., P.O. Box 70288, Eugene, OR 97401. The editorial office is located at 1500 Valley River Drive, Suite 250, Eugene, OR 97401 (Tel. 503-485-8796). Subscription rates in U.S. and its possessions are $25 for one year, $45 for two years, and $63 for three years. In Canada and Mexico add $11 per year. Other foreign countries $43 for one year surface mail. Inquire for air delivery. Single copy price in U.S. and its possessions is $3.50, and $4.50 in Canada and Mexico. Foreign subscription payment should be in United States funds drawn on a U.S. bank. Second-class postage paid at Eugene, OR 97401, and Columbia, MO 65201.

POSTMASTER: Send all address changes to **Home Computer Magazine**, P. O. Box 70288, Eugene, OR 97401. Subscribers should send all correspondence about subscriptions to above address.

Address all editorial correspondence to the Editor at **Home Computer Magazine**, 1500 Valley River Drive, Suite 250, Eugene, OR 97401. Unacceptable manuscripts will be returned if accompanied by sufficient first class postage and self-addressed envelope. Not responsible for lost manuscripts, photos, or program media. Opinions expressed by the authors are not necessarily those of **Home Computer Magazine**. All mail directed to the Editor or to the "Letters to the Editor" column will be treated as unconditionally assigned for publication, copyright purposes, and use in any other publication or brochure, and are subject to **Home Computer Magazine's** unrestricted right to edit and comment. **Home Computer Magazine** assumes no liability for errors in articles, programs, or advertisements. Mention of products by trade name in editorial material or advertisements contained herein in no way constitutes endorsement of the product or products by **Home Computer Magazine** or the publisher unless explicitly stated.

| | |
|---|---|
| **Publisher/Editor-in-Chief** | Gary M. Kaplan |
| **Executive Editor** | David G. Brader |
| **Managing Editor** | Walter Hego |
| **Associate Editor** | Wayne Koberstein |
| **Sr. Technical Editors** | |
| William K. Balthrop, Roger Wood | |
| **Technical Editors** | |
| D. Donaldson, Tom Green, G.R. Michaels, Steven P. Nelson, Patricia Swift | |
| **User Group Editor** | Judy Campbell |
| **Assistant Editor** | Dana M. Campbell |
| **Program Translators** | |
| Hendrik Broekhoff, Stephen A. Cordon, Jeff Fund, Paul Klissner,Robert Paschelke, Randy Thompson, Nancy Vendelin Jim Whitaker | |
| **Asst. to the Publisher** | Rhea J. Grundy |
| **Production Manager** | Norman Winney, Jr. |
| **Creative Director** | Gei-Lei Gom |
| **Photography** | |
| Nelson Stevens, K.D. Wainsworth | |
| **Production Assistant** | Rachel Knight |
| **Customer Service** | Tel. (503) 341-1029 |
| **Dealer Sales & Distribution** | Tel. (503) 341-1036 |
| **Main Switchboard** | Tel. (503) 485-8796 |

# Outside HCM

Here it comes to save the day . . . *Budgetron* is on the way! No matter how fancy the pie—from Ma's apple to Sara Lee's rich chocolate cream—someone has to slice it, making sure there's enough to go around. And from *your* house to the White House, computer budgeting may be the mighty hero of the day. So instead of spending straight out of your back pocket, put that pie on your computer screen. In time, your savings could, in fact, be the biggest slice.

# Inside HCM

Is it *work* . . . or is it *play*?

Computers often perform real work by making it *look* like play. Absorbing vast amounts of tedium, they let us humans have all the fun. At its best, computer software generates enjoyment as a way of coaxing out of us just the right information to get the job done. Perhaps the ultimate productivity package of the future will seem more like a video *game*—with an intriguing scenario that turns "user-input" into a series of strategic game "plays."

It's in the same spirit of productive play that we have developed this issue's handy key-in-and-RUN software. For example, take *Budgetron*—a monthly finance calculator with a dollar-wise sword. If you've had trouble slicing your budgetary pie, stop and let our friendly cybernetic hero "hack" your budget into shape. Not only will your pie *reach* farther—but by accounting for every crumb, you may find it to be even *richer* to the taste.

As the pie slices, the world divides into measurable bytes. Playing with basic two-dimensional shapes is one way we learn how to "measure the earth." If learning (or teaching) is your aim, train your sights on *Geometrix*—a tricky computer exercise with both a practical and an educational purpose.

Play often imitates work—and a lot of work comes disguised as play. In this issue, two game programs imitate the workaday world of war and power. *Torpedo Alley* simulates a lively contest of death beneath the waves. And, departing from the usual "Fire when ready, Gridley" format, *Over-Reaction* puts you in charge of a dynamic nuclear energy system where the order of the day is, "Stay Cool."

Among our regular features, *Apple Seedlings* nurtures helpful utilities such as this issue's character graphics editor. Our second *Commodore Hornblower* column continues to build a BASIC synthesizer for the C-64. Sound is also the subject of *IBMpressions*, which in this issue illustrates simple sine waves blending into complex audible patterns. And *Razzle Dazzle* shows how TI-sprites bring depth to an animated sequence.

There's even more work at play in our LOGO program features: *Achilles and the Turtle*; and *Build a LOGO Adventure, Part 3*. We also provide a real programmer's gem in Part 1 of *Speeding Up a BASIC Program*.

Software is our work, but we also take time for some serious play—reviewing a full-range of products for your home computer. In this issue, we key-in on a host of typing instruction packages; talk sense about *Dollars and Sense*, a best-selling financial planner; and take a sharp-eyed view of the *Gibson Light Pen*. Or, if you want to really get your computer off the ground, come fly with us as we reconnoiter *Flight Simulator II*.

Fortunately, our reconnoitering doesn't end up in the clouds. Back down on the ground, we uncover a snappy utility for the C-64, tongue-lash a speech synthesizer for the IBM PCjr, dig up a *Missing Link* to connect the TI-99/4A to a parallel printer, and unscramble a computer version of *Scrabble*. Then, in Part 2 of *The Music of Sound*, we compare the C-64 musical software reviewed in Part 1 to the Casio CT-6000, a professional-level keyboard with similiar sound and music capabilities.

So, to turn you work into play—or to make your play work for you—start turning the pages of *Home Computer Magazine today*.

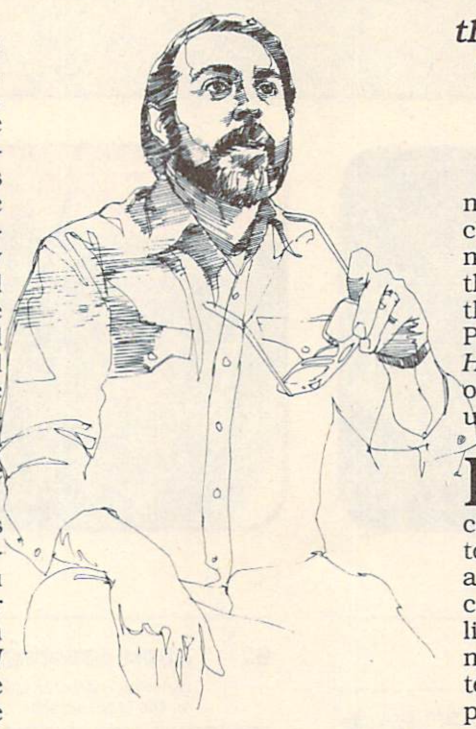
**Until next time, have fun reading, learning, and RUNing**

**HCM**

# On Screen

**By Gary M. Kaplan**
*Publisher & Editor-in-Chief*

*"One day, our program-design staff found itself musing over the idea of doing something very different in the pages of our magazine . . ."*



Keep 'em coming—the "On Screen Feedback" letters, that is. In last issue's column, I asked for your ideas on ways to "fine tune" this magazine so that we can provide more of what you want, and less of what you don't want. Your response has been, as my son would say, "totally awesome!" Poring over the daily sacks of mail has, above all, convinced me of two things: First, I must have been slightly delirious when I invited this mass response upon myself; and second, *HCM* reader feedback in the form specified last issue is undoubtedly an effective way to unleash a staggering tidal wave of creativity, constructive criticism, and fresh new ideas . . .

Even though I asked you to focus your comments exclusively on the *content* of the magazine, many of you didn't. Instead, you also voiced your dissatisfaction with the long wait from the time your order for *ON DISK* or *ON TAPE* is placed, until the disk or tape actually arrives by mail. All along, we assumed that you would favor the trade-off in timeliness versus cost—hence utilization of Third-Class bulk mail (with its inherent delays caused by requisite batching and sorting up front, plus the slower delivery and "third-class" handling that the postal service relegates to bulk mail). Well, as the line goes—we've finally seen the light! Effective immediately, all *ON DISK* and *ON TAPE* media will enjoy speedy First-Class treatment, plus stronger mailing containers to ensure that your software arrives in "first-class" condition (with automatic forwarding upon change of address). To pay for all this, we've been forced to pass along a one-dollar shipping and handling charge on all single-disk and single-tape orders. We're sure you'll soon realize that this small extra cost will make all the difference in the world.

Incidentally, many of you have thanked us for the extra file or template that we occasionally put on our magnetic media when appropriate. We'll be doing more and more of this for "extras" that we can't put in the magazine because of space limitations and key-in problems. One piece of related good news: With our current *ON DISK* and *ON TAPE* Program Subscription service, we're able to absorb the lion's share of the extra first-class shipping charges due to a subscription's much lower one-time transaction processing cost. Thus, we can pass on to you even greater value and savings. I suggest that you take a look at our special "get acquainted" offer found on the multi-color side of the *center* bind-in card. You'll be very glad you did!

Many of our PCjr readers have already written and expressed concern with IBM's decision to cancel further production of their machine. I'd like to take this opportunity to assure all Junior users out there that we will definitely continue our full support of the PCjr. This complete tutorial, review, program, and media support is especially crucial when a manufacturer "orphans" a machine. Our commitment to excellence cannot be altered by a manufacturer's decision—even one the size of Big Blue. So please, spread the word to all your fellow users of PCs, PCjrs, and PC-compatibles that *HCM* is the *only* magazine that can, or is willing to, provide this total user support.

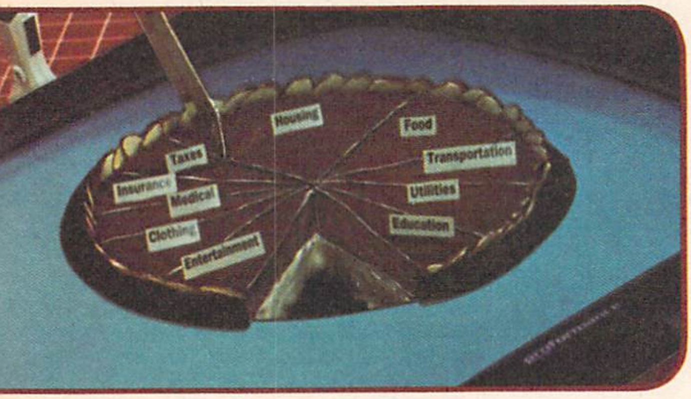
I'd like to now discuss one of the programs included in this issue. We've called it *Over-Reaction*. First, I'd like to state that this program is *not* just a game. To call it "just a game" is like calling a Rolls Royce "just a car" (or like calling *HCM* "just a computer magazine". . . ). So, those of you who tell me you're not interested in games, please take note:

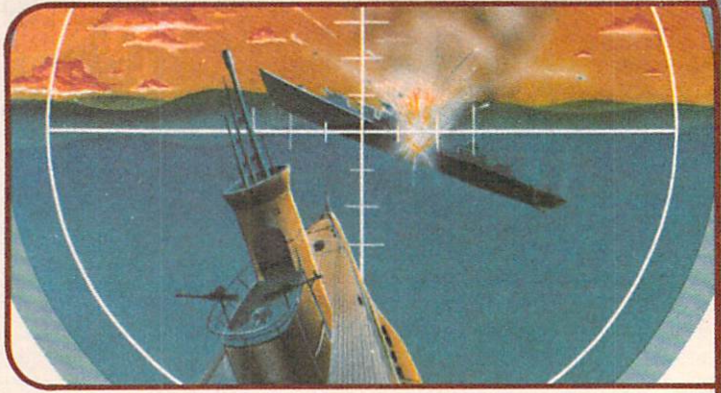
One day our program-design staff found itself musing over the idea of doing something very different in the pages of our magazine—providing a realistic, process-control activity in which our readers could participate in a stimulating learning experience.

Our goal was thus to design a real-time activity that required dynamic strategy, rather than the instant responses and quick trigger-finger demanded by a typical video game. We wanted to simulate the experience of maintaining a complex system—one that is subject to many outside influences beyond the player's control. The perfect scenario eventually suggested itself: a runaway nuclear power plant. In this model, Murphy would have the upper hand. Everything that could go wrong, would. Fortunately, one of our programmers had actually worked in a nuclear plant, and we were thus able to build-in some first-hand realism to boot.

Our first version of the program, however, turned out to be too stable. In fact, our in-house "play-testers" facetiously groaned in unison, "it's *boring* to the core." In subsequent versions, we finally assuaged the snide snickering from our sarcastic staff by introducing more entropy into the algorithm. In short, we made more things go wrong, more often—thus, *compressing time*. The result: an exciting real-time simulation—a "game" that should appeal even to those who have no taste for the usual shoot-em-up video fare. It's one I thoroughly enjoy "playing." I think you will too.

# HOME COMPUTER™ magazine



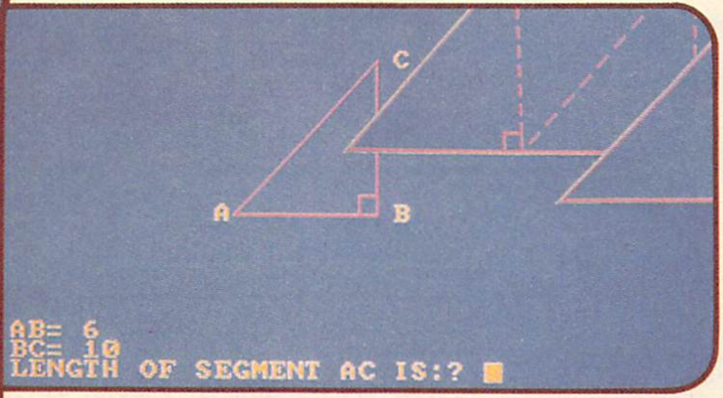


## FEATURES

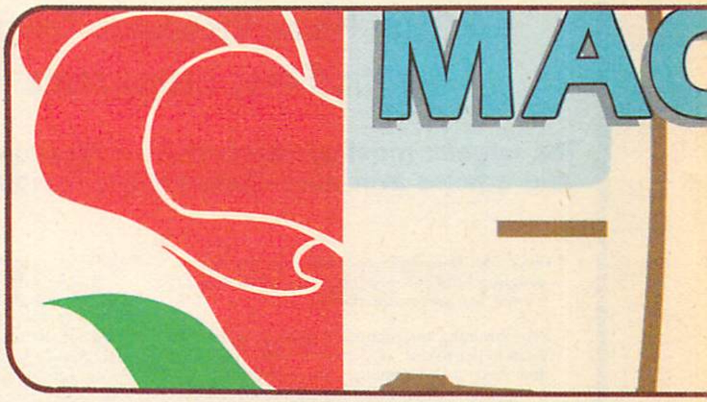
## PRODUCT REVIEWS

# CONTENTS

VOLUME 5 NUMBER 3

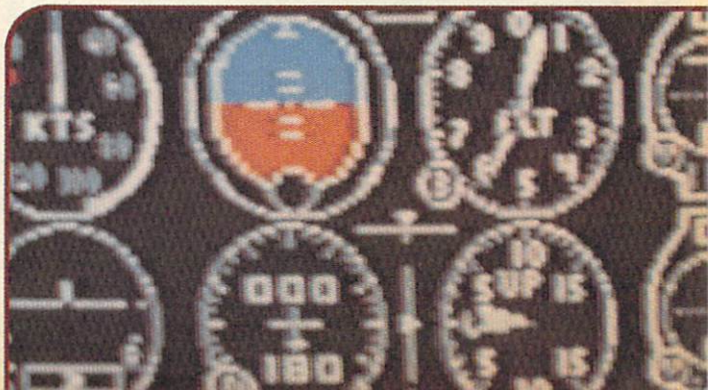


```
AB= 6
BC= 10
LENGTH OF SEGMENT AC IS:? ■
```



## DEPARTMENTS

**Home Computer Tech Notes:**



**HISTORICAL NOTE**
99'er Magazine (founded in December, 1980) was the forerunner of Home Computer Magazine.

# THE BEST OF 99'er™

SUPER CLOSE-OUT SPECIAL for TI-99/4A USERS

## A Giant Home Computer Compendium™ for the Texas Instruments 99/4A

VOLUME 1

**The largest, most comprehensive collection of programs and articles ever assembled for the TI Home Computer**

- Over 200 thoroughly tested key-in-and-RUN programs and sub-programs typeset in a grid format for maximum clarity.
- Programming instruction in 4 languages—learn to use BASIC, Extended BASIC, LOGO and Assembly Language—for everything from record keeping and money management to arcade-quality action games.
- A selection of sensational game software featuring full-color graphics, animation, and sound effects.
- Beyond the owner's manual—tips and techniques for getting the most out of your computer system.
- Computer-Assisted Instruction—The home computer becomes your private tutor.
- Page after page of innovative applications—transforming your computer into a home productivity center.

**Regular (Pre-Close-out) Prices:**

| | | |
|---|---|---|
| Best Of 99'er (Book alone) | $19.95 | + $3.00 shipping |
| Best Of 99'er (Tape Set alone) | $35.00 | + $2.50 shipping |

USE BIND-IN CARD AT CENTER OF MAGAZINE

**SPECIAL OFFER** Buy the Tape Set for ONLY $35. And Get the Book FREE + FREE SHIPPING

PLUS

FREE BONUS WHILE SUPPLIES LAST

ORDER THE BOOK & TAPE PACKAGE AND RECEIVE A Simon's Saucer AND A 99'er Programmer's Guide **ABSOLUTELY FREE!** This Additional $18 gift IS YOURS IF YOU ACT TODAY!

FREE BONUS WHILE SUPPLIES LAST

2
Programming Techniques and Languages
4
LOGO
1
6
Computer-Assisted Instruction
8
Applications and Utilities
3
Inside BASIC and Extended BASIC
THE BEST OF 99'er
FOR THE TEXAS INSTRUMENTS HOME COMPUTER
les & Programs for the TI-99/4A
BEST OF 99'er ON TAPE
Tape 3
Tape 4
Tape 5

## THE BEST OF 99'er ON TAPE

A Choice Selection of 37 Full-length Programs On 5 Quality Cassette Tapes

★ Save Typing Time and Frustrating Key-in Errors.
★ Own the Most Comprehensive Software Library for the TI-99/4A
★ Enjoy Hundreds of Hours of Exciting Computer Activity.

TO ORDER—USE BIND-IN CARD AT CENTER OF MAGAZINE

People Who Know the Home Computer Best™

HOME COMPUTER magazine™

AN EMERALD VALLEY BOOK EVP

**IF CENTER BIND-IN ORDER CARD IS MISSING— MAIL WRITTEN ORDER TO:**
P.O. Box 70288 Eugene, OR 97401
FOR CREDIT CARD ORDERS CALL 1-(800)-828-2212

# Letters to the Editor

### Expiration Issue Clearly Marked

Dear Sir:

Congratulations! Last issue you said your new schedule would have approximately 5 to 6 weeks between issues, and my latest issue (Vol. 5, No. 1) arrived 6 weeks later. After all the shifting around you've done since production stopped on the 99/4A this is a real milestone. Just one comment though. Your recent issues have just had a volume and issue number—no date—and my subscription label gives an expiration date (month and year). This makes it hard to figure how many issues are left in my subscription. Why not continue putting the month on the magazine with two issues each year covering two months, such as June/July and December/January? This would make it easier for your subscribers to keep track of the time on their subscriptions.

Thomas Steinhart
Decatur, IL

*No sooner said than changed, Thomas. If you will look at your present address label, you will see that it no longer has an expiration "date." It now has the volume and issue number of the last issue to which your subscription entitles you. This is the clearest method that we could devise for subscribers, now that we are on a ten-issue per year publication schedule. To avoid confusion at the newsstand, we must presently refrain from putting a date on the cover.*

### Your Days are Numbered

Dear Sir:

In the Letters column of Vol. 5, No. 1, Mr. Donald Mefford asked if anyone knew an algorithm to determine the number of days between two dates. This sort of calculation is quite common in astronomy where Julian Date numbers are used. The Julian Date is the number of days since January 1, 4713 B.C. To determine the number of days between two dates you simply subtract their Julian Day numbers. A subprogram written in TI Extended BASIC to determine the Julian Date follows:

```
  100 SUB JULIANDAY(M,D,YR,JD)
  110 IF M=1 OR M=2 THEN F=YR-1
ELSE F=YR
  120 IF M=1 OR M=2 THEN MF=M+12
    ELSE MF=M
  130 A=INT(F/100) :: B=2-A+INT(A/4)
    :: C=INT(365.25*F) :: DP=INT(30.6001
    *(MF+1)
  140 IF YR<=1582 AND M<=10 AND
    D<=15 THEN B=0
  150 JD=B+C+DP+D+1720994.5
  160 SUBEND
```

The Julian Day is returned in JD. This routine will find JD for any date; if you're only interested in dates this century you have to enter the full four-digit year. The day of the week = (JD + 1.5)/7 where Sunday = 0, etc.

Max A. Shelhorse
Edina, MN 55435

*Say, that's great, Max, but in testing your program we seem to have found a "short-term" bug. We used your suggestion to see how many days there are between January 1st to March 1st of the same year and arrived at 65! Still, it gives very close results even for periods of several hundred yars. Maybe one of you other readers could shed some light on this Julian date mix-up.*

### More on Adding a Drive to PCjr

Dear Sir:

Absolutely delighted with Mr. Brader's article "Adding a Second Drive to the PCjr" (HCM Vol. 4, No. 4). The added drives work like a champ! I added a third drive also, since I had one on hand and it was especially easy because the extra logic was already on board with the piggyback chips. Now I believe my Junior is even better than a PC because of the additional colors available in medium and high resolution, and of course, the more versatile sound.

I had the same trouble initially that Mr. Beifuss reported in his letter (Vol. 4, No. 5), getting an error on bootup. The problem turned out to be that the 74LS10 gate just wasn't able to sink enough current to fully select my Shugart half-height drives. Swapping the chip with a standard 7410 did the trick.

Your readers might be interested to know that the software which comes with Tecmar's jrCaptain add-on memory board enables the PCjr to recognize the new drives without having to make the MODBOOT.BAT file described in the article. Tecmar's CONPCJR.EXE program does the job, and perhaps it can be purchased without purchasing the memory board (although I highly recommend the board).

Keep up the good work.

Ron Sutherland
Vienna, VA 22180

*Thanks, Ron, for the appreciation shown and the information on the Tecmar CONPCJR.EXE program. For those readers who had not been aware of the construction article and would like to get a copy, write or call the HCM subscription line and ask for the HCM Vol. 4, No. 4 back issue for $3.95. Also, our PCjr Second Disk Drive Kit is still available for $49.95. This kit contains the special cable and the two ICs mentioned in the article. With the copy of the article and the kit of parts, all that is needed is an external IBM-compatible disk drive.*

### Delay Loop Trick for the C-64

Dear Sir:

Simon's BASIC contains a convenient function, PAUSE, which allows the programmer to specify a pause in the program for a specified number of seconds. In Commodore 64 BASIC, a pause is often created with a FOR . . . NEXT loop.

Using a FOR . . . NEXT loop to create a delay has two disadvantages. First, unless you use the technique frequently enough to have a feel for it, you must experiment to find the proper range for your loop counter. Second, if you use one of the compilers now on the market, you will need to redetermine all of your loop counters, since the compiler makes the loops run faster.

There is another way to create a delay in the program for a specified number of seconds, without the bother of experimentation, and which will be unaffected by compiling the program. Set the value of S in line 500 equal to the desired delay in seconds:

```
  . . . program ready for pause
  500 S=30:GOSUB 900:REM PAUSE 30
    SECONDS
  . . . more program after pause
  . . . END
  900 T=TI+60*S:REM PAUSE
    SUBROUTINE
  910 IF T=TI THEN RETURN
  920 GOTO 910
```

Since this technique makes use of the internal clock, rather than the time required to execute a FOR. . .NEXT loop, it requires no experimentation and is immune to a compiler. And, if desired, you could even insert a GOSUB between 910 and 920 for a routine which does something during the pause.

Perhaps your readers would find this technique useful.

Jack Ryan
El Dorado, AR 71730

*Pretty tricky, Jack! You notice we didn't "delay" too long in publishing this little routine. In fact, some readers may consider it "just in time."*

### Locating a TI Disk Manager 2

Dear Sir:

In Vol. 5, No. 1 of *HCM* there was a letter from James A. Canter, who was looking for a Disk Manager 2 cartridge. After reading the same article ("2 for TI"—Vol. 4, No. 4), I attempted to locate a Disk Manager 2 cartridge and after calls to numerous dealers and to Texas Instruments, this is what I found.

There are only three ways to get a Disk Manager 2 cartridge:

1. Try a user group (which you suggested).
2. Buy another disk controller card (with DM 2).
3. Send your old Disk Manager to Texas Instruments.

Texas Instruments said that if we send in our old Disk Manager cartridge, they will upgrade it to a Disk Manager 2 for a fee of $14.25 plus state tax plus $2.00 shipping/handling.

They did ask that we call in advance before sending the cartridge to them. The number to call is 1-800-842-2737.

I hope this can be of help to James and anyone else wanting to upgrade their system.

Edward T. Stack
Edwardsville, IL 62025

*For those of you who are not familiar with this solid-state cartridge for the TI-99/4A, it allows you to format and catalog double-sided disk drives (assuming that you have a double-sided disk drive connected to your system). The software in Disk Manager 2 is also set up to allow double-density recording, but the standard Texas Instruments disk drive controller card does not allow this. Some of the third-party disk controllers on the market may allow double-density recording. Watch for reviews of this third-party hardware in HCM in the near future. By the way, Edward, Texas Instruments does not require that you send the old cartridge.*

*Continued on next page*

### How To Buy An Apple

Dear Sir:

I am a soon-to-be owner of a computer. I know very little about the computer field. Words like interface, CP/M, extended memory cards, peripheral cards, buffers, hard and floppy disks, disk drives and more confuse me terribly!

I wish to buy an Apple IIe, a printer, one or two disk drives, a voice synthesizer, and have ProDOS, 128K, also a color monitor and surge protector.

I'd like to know how it would all fit together. Do I open the computer? If so, how and what tools would I need, if any, to put it all together? Do I need interfaces, cards, and other things along with the actual named products?

As you can probably see for yourself, I am really confused.

I'm writing to you because your magazine seems to be the best. I've bought other magazines yet none compares to yours. I know you will give me honest opinions on cost and quality—a trait your magazine solely seems to carry.

Dorene Pope
Niagara Falls, NY 14304

*Judging from your letter, Dorene, we recommend that you buy your Apple computer from an Apple dealer in your hometown, and let him set up the computer configuration as you require. Often when purchased this way, you will get a package discount (something to ask about). When making the decision to purchase a system, try to get price quotations on the package from as many dealers in your area as possible (including the length of warranty on each item and the cost of any service policies). Once you have the entire picture, you can make an informed purchase decision. After you select a dealer, it is advisable to consider several alternatives. Your request for 128K, a printer, etc. may mean that the Apple IIc, with its many peripheral functions already built-in, is perfect—and cheaper! Much depends upon your particular application. We've found that AppleWorks, for example, works well on the IIc (see our review in HCM Vol. 5, No. 2). Your choice of a printer should be made with your needs in mind. If you will only be doing word-processing, a letter-quality printer might be the right choice. If, however, graphics is your cup of tea then a dot-matrix (like Apple's Imagewriter) or even a color printer might be more what you want. These are just some of the considerations that you should discuss with the dealer you select.*

### In Search of King's Quest for PCjr

Dear Sir:

Enclosed is my payment for a one-year subscription to a superb magazine. I have purchased the last two issues and find your magazine extremely interesting. Since our family recently bought a PCjr, we have read your articles extensively and programmed a few of the games. Incidentally, your game reviews and tech articles catch the interest of my children, which reflects that your magazine is geared for all ages.

We found your review of King's Quest in the Vol. 4, No. 5 issue so favorable that we tried to buy the game. The store that specializes in software tried to order it for us, but to no avail. They informed us that the game was only available for the IBM PC. They claimed their distributor called Sierra On-Line to confirm this. Can you verify this? The article in Vol. 4, No. 5 indicated PCjr.

Also, in our programming of your game Bird Brain, we have a problem while playing. During the course of the game we get an illegal function call in line 640 which we typed correctly. Can you help?

In Stadium Jumping, on Grand Prix level we are unable to play because of an illegal function call in line 510. Again, can you help?

Nick Tomasi
Defiance, OH 43512

*Kings Quest was designed specifically for the PCjr and can be used on the PC if it has the special color graphics adapter and a color monitor. We recommend that you contact your closest IBM Product Center for availability. As to the two program bugs you mention—these appear to be key-in errors, as we have never had the difficulties you describe. You must realize that just because the error message names a line, it is not necessarily where the error occurs. For example, if one of the variables in line 640 of Stadium Jumping were a zero, the illegal function call would result. To save yourself a lot of time solving these pesky key-in bugs, we suggest you get the programs ON DISK—the small price you pay can save you hours of aggravation.*

### Making the Reset Button Work

Dear Sir:

I've noticed in your latest magazine that a couple of your readers had installed a reset button on their C-64 and were having problems listing and running their programs after depressing the reset button.

This little program I picked up may possibly be of some help to your readers.

Load and run this program first. Then type in your program. Should it become necessary to hit the reset button, enter the SYS 49152 and you should be able to list and recover your program.

```
10 AD=49152:FORI=0TO21
20 READ D:POKE AD+I,D:NEXT
30 DATA 169,8,141,2,8,32,51,165,24
40 DATA 165,34,105,2,133,45,165,35
50 DATA 105,0,133,46,96
60 PRINT"TO EXECUTE,USE SYS";AD;
   ":CLR"
```

Bill Asher
Detroit, MI

*Thanks Bill. This is a good addition to the reset button Tech Note. Because other folks have written in saying that they could not recover their data, your little program will help immensely, and will make the use of the reset button more "user friendly." One change that some of you might want to make though concerns the starting address. Setting the variable AD in line 10 to equal 53225 would decrease the possibility of this routine conflicting with other machine-language programs. Because machine-language programs "crashing" is a major cause of needing a reset button, storing this program at 53225 might be safer.*

### Tension Sensed Over Review

Dear Sir:

Calmpute is the outgrowth of 11 years of research, development, and production of biofeedback equipment by Thought Technology.

The Calmpute sensor *accurately* monitors changes in skin conductivity—a measure responsive to the activity of the body's sympathetic nervous system which is responsible in part for heart rate, blood pressure, peripheral temperature, and adrenaline responses to stress.

Unlike electromyographic monitoring which provides information solely about the specific muscles being monitored (in the case of RELAX, the frontal region of the forehead), Calmpute detects physiological changes related both to physical and mental stress. The anomolies you reported ("high responses to neutral words such as "wood" and "door") are indeed responses *you* produced, perhaps in frustration observing such simple words or because your mind wandered to something provocative. Another aspect involves "the perception of relaxation." One might report feeling relaxed, however his/her mind may not be relaxed. For example, simply watching the feedback trace on the screen going towards a stressful reading often produces stress in the novice user.

The point I wish to stress is that to develop relaxation skills requires practice.

To assist users, Sunburst Communications has developed a new more comprehensive manual and an optional 6-week stress management program in a 150-page book which they will offer with Calmpute in June 1985.

As well, we have rewritten Calmscope and Calmbar to provide color feedback, to print and save the displays, and have developed an interface to connect our temperature, heart rate, electromyogram, blood volume pulse, and respiration monitors to Calmpute.

Dr. Hal. K. Myers
President
Thought Technology, Ltd.

*Thank you, Dr. Myers, for the update on the Calmpute product which we reviewed in Vol. 5, No. 1 of Home Computer Magazine. If our comments on your product caused you to upgrade it—making the package more useful and a better value—we're really glad.*

### TI'ers in Great Britain Miss Ads

Dear Sir:

Congratulations on yet another wonderful edition of *HCM* Vol. 4, No. 5. I have been a subscriber for two years. I have the magazine posted from my nearest stockist (450 miles away). The content as usual was excellent. In many ways TI'ers in England miss the ads because it was our only way to obtain information as to what and where anything was available for the 99—in particular, add-ons (disks, etc.)—because the supply of just about everything for the 99 in the UK has dried up (tears and more tears!!).

Could you ask any readers if they can come up with any circuit diagrams and lists of parts for stand-alone 32K expansion or any other hardware because it seems our only method left is to build them ourselves.

Could you also give me information as to how I can receive software on cassette from your issues, as we buy our magazines in England. Hope you can help.

Malcolm Pryke
Newcastle Upon Tyne
ENGLAND

*Malcolm, we will send you the information via post regarding purchasing HCM software and cassettes in the United Kingdom. Plus, we'll send some recent back issues of our sister publication, Home Computer Digest, which contains information on merchandise from U.S. vendors of TI-related wares. We believe that purchase from U.S. sources to be a better solution for you than building from scratch.*

## Apple Snap-Calc SNAFU?

Dear Sir:

I have a slight problem with the SNAP-CALC and SNAP-FIX software. I am using an Apple IIc with a single disk drive, and therefore am not able to convert SNAP-CALC from DOS 3.3 to ProDOS. When I received the SNAP-FIX update on a ProDOS disk, I looked on the back side of the disk for the DOS 3.3 version, as the instructions said, and found nothing. I am therefore not able to get the enhanced version of your spreadsheet program.

Now that I have my complaining out of the way, I'll get to the important stuff. I love the mag! Your format and concept are unchallenged by the others in the publishing field, and your coverage of technical articles seems to correlate perfectly with the questions that I have. You always seem to include the most informative pieces in the Letters to the Editor as well.

I never miss an opportunity to recommend your magazine to my friends who also are interested in computers, and am considering the possibility of giving a gift subscription of the magazine and the ON DISK to my children's school. Keep up the good work—I look forward to every issue.

Ron G. Reist
Kalamazoo, MI 49007

*Sorry, Ron, that your disk didn't have the SNAPFIX file on it. The oversight has been corrected in more recent releases of the ON DISK 4.5. In addition, an even more recent update file (which includes all the changes that were in the SNAPFIX file) has been included on the back-side of the ON DISK for Vol. 5, No. 2 entitled CALC.FIX. Your idea of presenting a subscription to your children's school is an excellent one. We hope that other readers also consider this as well, because of the difficulty that many schools have in finding enough money to purchase software for their students' use.*

## Using More Memory with 2nd Drive

Dear Sir:

After overcoming several component problems I was able to complete the installation of the second disk drive on my PCjr. All is well when I use my Junior as the standard 128K machine, but I have been unable to get the additional memory to work with the second drive. I can use one or the other, but not both at the same time.

My PCjr is up to 256K. Can you tell me what code I have to change in order to take advantage of the additional memory and the second disk drive?

Thank you for your time and consideration.

David J. Calabretta
Havertown, PA 19083

*David, it's difficult to give you an answer to your question without knowing which manufacturer's memory add-on you are using with the PCjr. But most manufacturers do supply some software that allows configuration of the memory. In general, this software should be run after the system has been booted as a dual-disk system and before application software is run. For related information see the letter entitled "More on Adding a Drive to PCjr" on page 11.*

## TI Wiring Diagrams Available

Dear Sir:

Reynaldo Rivera (Letters to the Editor, *HCM* Vol. 5, No. 1) can obtain wiring diagrams for his TI-99/4A from:

Howard W. Sams and Co., Inc.
4300 West 62nd Street
P. O. Box 7092
Indianapolis, IN 46206 U.S.A.

Their toll-free telephone number for orders is (800) 428-7267. Ask for SAMS COMPUTERFACTS-of-the-month Set #CF1 Folder CC2 for the TI-99/4A Model PHC004A.

This folder contains diagnostic procedures, parts lists and plenty of other information for true hardware junkies.

I found my copy of this folder at a local electronics store amid other SAMS folders on Apples, IBMs, Commodores, and various printers and disk drives. So SAMS should be able to provide technical information to any of your readers.

Keep up the good work on the magazine, and I would appreciate a review of Turbo Pascal for the TI-99/4A.

Jon Ruth
Boulder, CO 80303

*That is really good information for the TI "hackers" who like to dig around inside the hardware. Thank you, Jon. By the way, Sams refers to the packet as Product #8901 for $19.95. Regarding a review of Turbo Pascal for the TI-99/4A, it is our understanding that no such product exists. Turbo Pascal by Borland International runs on MS-DOS machines (such as the IBM PC, PCjr, and the Texas Instruments Professional Computer), CP/M, and CP/M86 machines, but Borland does not presently plan on producing any products for the TI-99/4A home computer.*

## Ask and You Shall Receive

Dear Sir:

I have a Commodore 64 with disk drive, cassette, printer, and TV. We have various programs already—games, educational, home, and programming.

I bought the computer for all the above reasons, but mainly for educational use. I have three boys, ages 8, 10, and 12. I am interested in an educational program that deals with daily decisions of growing kids. Such decisions as are presented by word problems. But not ones like, if two trains are on the same track going a certain speed, when will they meet? I'd like one that deals in quantity problems. One that can combine math, measurement, choice, and logic.

I would appreciate it if you could let me know if there is such a program available.

You had a program in an earlier issue that dealt with a bills budget called "Cash Flow." Do you have a Commodore version of this program or something similar?

Dennis Rodgers
Bartonville, IL 61607

*Dennis, first may we suggest that you turn to Geometrix in this issue and see what your boys think of that as an educational program. We feel pretty proud of it ourselves. In answer to your question regarding the cash flow program which first appeared in Vol. 2, No. 10, we do not at this time have a Commodore version of the program. But we do recommend that you look at the program, Budgetron, in this issue and see if it will satisfy your needs for controlling your cash flow.*

## Apple One-Liner Error?

Dear Sir:

I found an issue of your magazine by accident. I purchased an Apple IIc the day before Christmas in hopes of enriching my family's life. So far, I've had nothing but expensive lessons with very little interest from my family and a nagging wife. I was utterly shocked at the cost of the commercial software—$58 to $60 per item—that's crazy!

I purchased your magazine, Vol. 4, No. 5 and attempted the one-liner on page 34 for the Apple. All my kids were around me anxious to see me do something creative. When we went through the program twice, item for item and then ran the program, nothing happened. My screen showed # of sides, and if I pressed any key, I would get a SYNTAX ERROR. I could not remedy the program.

In spite of everything my family and I are anxious to become involved in computers.

Michael Wlodavczak
Bay City, MI 48706

*Michael, the only keys that are allowed in that one-liner program are number keys. If you*

*Continued on next page*

press any key other than a number key, you will get a syntax error and the program will stop. A one-line program just doesn't leave much room for error checking. By the way, take a good look at what you are holding in your hand, Michael. We think you'll soon discover that HCM's program content is the "quintessential software bargain" in the industry.

### Using A Serial Printer on PCjr

Dear Sir:

I am writing in response to Ed Quenzer's letter in Vol. 5, No. 1. I too have moved from a TI-99/4A to an IBM PCjr and am using my Smith Corona TP-1 serial printer connected to the serial port on Junior. I initially had some problems with the cabling, but getting the port to work was not a problem. The answer lies in the DOS 2.1 program. The serial port on my system is opened with the following AUTOEXEC.BAT file that I have included on all my system disks:

```
MODE COM1:60,,,,P
MODE LPT1:=COM1
DATE
TIME
```

This file configures the output to COM1 (the serial port) for the TP-1 printer, and redirects all output to LPT1 (the IBM printer port) from LPT1 to COM1, and then brings up the date and time prompts. Other than the cable, no other changes were necessary to get my system running.

I find that your magazine is ideal for a user like myself, and it has been very useful to me in making the transition from one system to another. With the demise of the PCjr-specific magazines, there is no other source of good, pertinent information concerning problems like the above. Keep it up.

Gordon K. Jones
Lake Forest, IL 60045

*That's good information to pass along to other folks who are trying to use the serial port for a printer. Thank you very much, Gordon. Just an added note: the first line which states MODE COM1:60,,,,P contains the specifications for the set-up of the serial port. The 60 represents a baud-rate of 600 (the first two digits are the only significant ones in this parameter). The commas make the operating system use the defaulted parameters for the parity (E), the number of databits (7), and the number of stopbits (1). The P at the end of the statement indicates that the asynchronous adapter is used for a serial printer. Consult your PC-DOS manual to learn more about how the MODE command can be used to redirect printer output to RS232 devices.*

### Locating Commodore Printer Ribbons

Dear Sir:

In your Vol. 4, No. 5 issue, a Mr. Hugh Valliat wrote on the problems of the replacement of the ribbon for the Commodore printer, MPS 801.

Radio Shack DMP-110, Part No. 26-1283 is a perfect replacement ribbon for the printer. Thought I would pass this information along. In closing, I would like to say, keep up the good work—*HCM* is the best computer magazine out today.

Joseph J. Forte
Philadelphia, PA

*That's great information, Joseph. We've also noticed that Radio Shack carries many useful items for the home computer user, including spare parts for the TI-99/4A home computer.*

### Multiplan Solves Again

Dear Sir:

Your letter published in Vol. 4, No. 5 from John R. Johan asked for software that would look up data in tables. I too have done machine tool estimating, so I know what he wants. Microsoft Multiplan has a table LOOKUP function which I use on the C-64 and the TI-99/4 for estimating income taxes. Multiplan is excellent for estimating, if the jobs are repetitive enough and do not vary in format. One could make up a spreadsheet for each of 5-10 different situations. I entered a tax table on the spreadsheet, and positioned it where it will not print out. When required, values are looked up in it and included in the calculation.

The disadvantage of this is that if the table is large so that the percentage of free space figure in the bottom right of the screen is less than 70%, the program execution slows annoyingly. The solution is to run Multiplan on a computer with larger memory, like the IBM PC or PCjr.

Another, more cumbersome possibility is to have the tables on external worksheet files with each entry given a name like INFO12, INFO13, etc., on external sheets with names such as POWER, COST, and DISCOUNT. You have to manually enter the external COPY command unless you establish permanent links. Data in permanent links is loaded each time a worksheet is loaded, so you wouldn't want to load all of them on your active sheet or it would fill up your memory.

In summary, Multiplan will do the job if each estimate is not too unique in format and the tables are not too large.

Cal Lamoreaux
Shelbyville, MI 49344

*Very interesting! Thanks, Cal. Multiplan—and one of our knowledgeable readers—comes through again with another solution to a sticky problem.*

### Thinkjeting the TI Home Computer

Dear Sir:

I read with interest your review of the HP Thinkjet printer in the Vol. 5, No. 1 issue of *HCM*. Having purchased a Thinkjet six months ago at a discount through my company's buyer plan, I am thoroughly satisified with its performance. As you correctly reported, use of the special short-fiber paper is a must. In my opinion, its small footprint (9" x 11"), speed, quiet operation, and quality print (11x12 dot matrix) make the Thinkjet perfect for home use and well worth the price.

I did, however, experience a little difficulty interfacing the Thinkjet to my TI-99/4A. The solution turned out to be a parallel cable with special electronics—the same required for interfacing a Smith Corona TP-1 to the 99/4A. I also found that I had to open a file with a fixed 81 length in order to stop the Thinkjet from splitting and printing my text on two lines. I hope this information is of some help to other Thinkjet-99/4A owners—it took me two cables and two weeks to solve.

B. Caldwell
White Oak, PA 15131

*Several of our readers have written expressing the same problem that you uncovered in interfacing the Thinkjet to the TI-99/4A. We are therefore very happy to publish your solution.*

### BASIC Understanding on the Apple

Dear Sir:

We have had and enjoyed our Apple IIe for over a year now, and while my older son and I live with the word processing and my husband values his VisiCalc, I was delighted with your latest issue of *HCM* (Vol. 4, No. 5). I know just enough BASIC to understand what is going on, but I certainly don't have the know-how to create long, involved programs. So I typed in your "Division Tutor" and my daughter (fourth grader) was delighted—it was perfect for her. We have also "made" a quiz and "taken" it with equal success. [See "Quiz Construction Set" in *HCM* Vol.4 No.5.—Ed.] I want to thank you for remembering that computers can do something else besides balance budgets. Please keep up your imaginative programming geared toward the "at home" market.

Mrs. Katharine Kohudic
Sumerduck, VA 22742

*Thank you, Katharine. You can be assured that we will continue to maintain a balanced editorial and program content for home users.*

### A Tough Act to Follow . . .

Dear Sir:

As the owner of an IBM PCjr, I am pleased to see that your magazine is providing useful how-to articles for this home computer, such as that dealing with the addition of a second disk drive. Now that you have supplied PCjr owners with a cost-effective method of overcoming the single-drive limitation, I would like to suggest that you address two other areas in regard to expanding the PCjr's capabilities.

First, is it possible to expand the PCjr's memory by replacing any of the banks of 64K chips inside the enhanced PCjr or on the IBM memory expansion "sidecars" with 256K chips? These chips have decreased significantly in price in the past few months, and their use would result in a lower average cost per K of RAM relative to the alternative of multiple-expansion sidecars. As this modification (particularly the internal substitution which superficially appears the most attractive) is likely to require some changes to both memory circuit boards and power supply, it would appear to be a good candidate for a separate how-to article.

Second, can you suggest any alternative uses for the modem slot inside the PCjr? As the serial port and an external modem can be used for communications, it might be more cost-effective to utilize the internal slot for other purposes.

Thank you for your assistance.

Paul E. Tubb
North Vancouver, BC, CANADA

*Challenges, challenges . . . always challenges. Thanks for the ideas, Paul. While we're investigating, perhaps some of our readers can share ideas on how to accomplish these projects. Has anyone actually produced a home-brew memory addition for PCjr? Perhaps they have actually used the external modem slot to accomplish this sort of thing. Let's hear from all you PCjr hardware hackers.*

### TI Printer With C-64 Revisited

Dear Sir:

A Vol. 4, No. 4 *Home Computer Magazine* Letter to the Editor mentioned "Using TI Printer With C-64." Can you tell me how to connect this system or set it up?

I have a TI printer, C-64, and Cardco +G parallel adapter. Will this setup print Commodore graphics?

Thank you for your help.

Craig Clark
Gresham, OR 97030

*First, Craig, you must remove the top of your TI printer and take out the serial interface card, then replace the cover carefully. Also, you should check to see that the DIP switches in your printer, and in the Cardco adapter, are set according to the specs given in the appendix of the manual that comes with it. Now, the parallel port located on the left-side rear can be connected directly to the Cardco parallel adapter and, yes indeed, you can print the Commodore graphics on that TI printer.*

### Following the KISS Principle

Dear Sir:

Regarding your "Apple Seedlings" Dating Game in Vol. 5, No. 1, remember to KISS while dating. Anders Nereim was on the right track using powers of two, but lines 420 through 460 may be replaced by:

```
420 NN=YR*512+MM*32+DD
430 A=INT(NN/256)
440 B=NN-A*256
450 DAY=A: YEAR=B
```

Line 210 no longer needs the DIM statement.

Elsewhere in the same issue, the need for a date "ordinal" in MMDDYY format was expressed. Here's a TI Extended BASIC function for such a problem:

```
10 DEF DATE=365*YY+DD+31*
   (MM-1)+((MM>2)*(INT(.4*MM+
   2.3)))+INT((YY+(MM<3))/4)-INT
   (.75*(INT((YY+(MM<3))/100+1)))
```

Nice things happen after this. . .

If your program places values in DD, MM and YY, then the day of week (Sat., Sun.,...etc.) is indexed by D when you execute the following:

```
100 D1=DATE/7 :: D=INT(7*(D1-INT
   (D1))+.5)
```

Warren A. Hall
New Hartford, NY 13413

*Thanks, Warren, for the information on the "ordinal" function—it works great. However, we found that your Apple Seedling update must be overlooking something. The variable names chosen above do not agree with those in the program. Plus, even if they are changed to agree, it still doesn't POKE the proper values into the ProDOS memory areas. Any Apple programmers out there who have been able to shorten the Set The Date program?*

### Snap-Calc Columns Do Not Align

Dear Sir:

When I received my first issue of *Home Computer Magazine*, (August 1984), I laboriously typed in the IBM PCjr SNAP-CALC program. I loved every minute of it and it worked to my satisfaction. It is carrying many of the programs I needed it for, including 1984 IRS taxes.

When Vol. 4, No. 5 issue came, I updated the program and was very pleased that it also worked very well. But, as I started to print my output, I saw that I have lost the relationship of the column numbers to the columns themselves. The only difference that I can find in my copying of the lines of the update is that line 890 calls for a one-bar straight vertical line, which I do not have on the Junior, and I used the vertical line, which was in the original printing.

Please tell me where the error is so that I can get back the original relationship of column numbers to the columns themselves. If you have sent out a disk with the update on it, I did not get one as of this date. Thank you very much. It's a great program.

Edwin C. Mutzer
Bradenton, FL 33507

*That one-bar straight vertical line is accessed by simultaneously pressing the* [ALT] *key and the* [ *(bracket/brace) key at the same time. Once this is entered correctly, you should have no problem at all, Edwin.*

### Clones Will Not Run ProDOS

Dear Sir:

I recently purchased a disk (Vol. 4, No. 4) for the Apple II computer. I have a Franklin Ace 1200 computer. Since I do not have ProDOS on my machine, how can I transfer all the DOS 3.3 programs to another disk? Right now the disk is useless. I can't boot it since ProDOS seems to cover every program on the disk. Any suggestions?

Is there any way, since you are going to use ProDOS, whereby two disks for the Apple Family can be produced and sold (one with ProDOS and one without)?

Robert Hose
Stockton, CA 95209

*Apple ProDOS is set up to work with an Apple II computer with 64K of memory and the Applesoft BASIC language in ROM. An Apple IIe with two disk drives and ProDOS utilities can be used to transfer ProDOS files to a DOS 3.3-formatted disk. Sorry, but it is impractical at this time for us to produce two Apple formats.*

### Building an MS-DOS Boot Disk

Dear Sir:

The first thing I would like to say about your magazine is that it is by far the best magazine I have read. The programs are very good, the articles interesting, and the reviews have helped me purchase some good software for my PCjr. I have written in to other computer magazines, but to no avail. My questions were not answered. I have been doing some intense programming lately and have run across a problem that is hindering me. How can you insert a disk after a system reset and get a designated program to run by itself? It's like when you turn on your computer and don't want to load in DOS, switch disks, and then load in the next program. I just want it to run as soon as the computer's drive is activated.

Thomas Dahbura
Hagerstown, MD 21740

*To boot your own programs automatically upon system power-up, Thomas, you must put it on a "bootable" disk. You can create this disk from a blank disk by placing the MS-DOS system master disk in the drive and using the following MSDOS command: FORMAT A:/S The operating system will then prompt you to put an unformatted disk in the drive and will FORMAT it for you. Once you have done this, copy the program file onto this disk using the MS-DOS COPY command. Finally, create an AUTOEXEC.BAT batch file on this same disk with the name of the program file you wish to boot. See the Home Computer Tech Note for IBM in Vol 4, No. 3 for details on Batch files. The next time you start the system with this disk in the A: drive, your program will boot automatically.*

### Use More Computer ID's in HCM

Dear Sir:

May I suggest you extend identification symbols of the computers applicable when writing the "Home Computer Product News?"

Also, on the Letters to the Editor, I feel that most readers are only interested in reading about what applies to their equipment. I find I waste time looking for items that concern my equipment.

Donald H. Farr
Coquille, OR 97423

*Yes, Donald, we'll consider what you say regarding the Product News column. As to the Letters to the Editor, we feel that the cross-pollination of ideas is very worthwhile, and we receive a lot of positive reader comments indicating that they enjoy reading about other user's experiences with different machines.*

HCM

# BUDGETRON

**by Scott Williams**
*and the HCM Staff*

***With this program in hand, the only hard part about making and following a budget is resisting the temptation to spend . . .***

This year you've resolved to become more practical and start planning and saving early for your vacation. The planning part is easy—Bermuda in late fall for two weeks. Being practical about saving is more difficult. If you could make up a budget that would reasonably limit your expenses and allow you to save $1700 between May and October, you could be on the beach sipping tropical drinks the first week of November. Plus, when you returned you could adjust your evidently successful budget to plan for new purchases, or relax it and enjoy a more liberal spending style.

*Budgetron* is just the tool to help you easily set up a custom home budget that allows you to estimate and track your monthly expenses. You may then multiply the net savings (or net losses) of one month to extrapolate your budgeting results over a series of months. Graphs and reports of your budget results can be displayed in a variety of ways.

Budgeted expenses may be listed in up to 16 of your own categories. For each category, you must enter a value which is the maximum amount of money that you wish to spend each month for items in that category. Actual expenses can then be itemized under each category and totaled for comparison against your budgeted values. Category 1 is used exclusively for income. (See Figure 1 for our sample budget. Actual expenses are indented.)

**The Main Menu**
1) WORK WITH BUDGET
2) REPORTS
3) BAR CHART
4) FILES
5) EXIT PROGRAM

NOTE:
In this text, we've color-coded the hierarchy of menus and submenus for clarity.

### 1) WORK WITH BUDGET

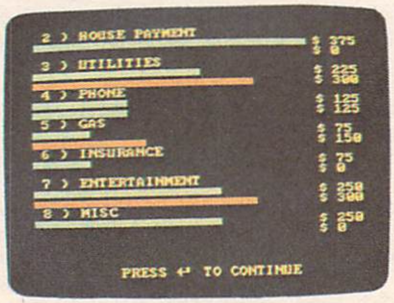
This is the first option that you will need to choose to set up a new budget. Selecting it will take you to the WORK WITH BUDGET menu screen:

1) BUDGET CATEGORIES
2) INCOME AND EXPENSES
3) EXIT

### Budget Categories

You should select this option first to set up your budget categories. Selecting it will display this next menu screen:

1) INITIALIZE BUDGET
2) ADD A CATEGORY
3) CHANGE A CATEGORY
4) EXIT



```
2 ) HOUSE PAYMENT                 $ 375
                                  $ 0
3 ) UTILITIES                     $ 225
                                  $ 300
4 ) PHONE                         $ 125
                                  $ 125
5 ) GAS                           $ 75
                                  $ 150
6 ) INSURANCE                     $ 75
                                  $ 0
7 ) ENTERTAINMENT                 $ 250
                                  $ 300
8 ) MISC                          $ 250
                                  $ 0

          PRESS ↵ TO CONTINUE
```

*The Bar Chart option from the IBM version of Budgetron shows a comparison of budgeted (green) and actual (orange) expenses for eight categories of a monthly budget.*

### Initialize Budget

If you already have a budget in memory and would like to start over again with a new budget, select this suboption. It will erase all of the existing budget categories, leaving the program ready for a new set. The actual-expense items are not changed by this option. If you wish to also erase the expense items in the budget, you must use the Initialize Items option under Income and Expenses. Do not use Initialize Budget if you don't want to lose whatever information is in memory. You will be asked to confirm your selection of this option before it executes.

### Add a Category

By selecting this suboption, you will see a list of existing categories, if there are any, and will be asked to enter a new category description. If there are more categories than the program can display on one screen, the program prompts you to either press the (SPACE BAR) to view more categories, or press (ENTER) or (RETURN) to get the prompt for entering new categories. After entering a description for the new category, you will be asked to enter its budget value. Remember, this budget value is the maximum amount that you have decided to spend on items in this category for one month. After entering the budget value, you will be asked if you would like to add another category. If so, press Y and the categories will be listed again.

### Change Category

After selecting this suboption you will see a list of the categories you have created so far. Select the category that you wish to change by entering its number. You will then be prompted to re-enter both the category description and its value. If you attempt to change category 1 (INCOME), however, you will be allowed to change only the value. You can't change the income category description.

## Income and Expenses

Select this option from the WORK WITH BUDGET menu when you wish to work with actual expense or income items in the budget. A menu screen will display the following options:

1) INITIALIZE ITEMS
2) ADD ITEMS
3) CHANGE ITEMS
4) EXIT

### Initialize Items

As mentioned earlier, you should use this suboption only if you want to erase all expense and income entries from the budget so that a new budget can be started. This option is generally exercised in conjunction with the Initialized Budget option, but it can be used alone.

### Add Items

This suboption will allow you to enter your actual expenses under each category, or an income value in category 1. You may enter as many items under each category as your system's memory will allow.

Select a category from the list displayed by entering the number beside it. You will then be asked to enter a description for the item. After entering a description, you will be asked for the item's value. For example, under our sample budget's UTILITIES category, we entered the description PHONE and a value of $60 when we paid our phone bill. After entering a value, you will be asked whether you would like to add another item. If you reply by entering Y, the list of categories will be shown again for your selection. Otherwise, you will be taken back to the main menu.

### Change Items

A list of all of your categories will appear when you choose this suboption, though they may not all fit on

FIGURE 1

Sample Monthly Budget

| | | |
|---|---|---|
| $$ INCOME $$ | 1500 | |
| PAYCHECK | | 1500 |
| | | 1500 |
| HOUSE PAYMENT | 400 | |
| HOUSE | | 400 |
| | | 400 |
| UTILITIES | 240 | |
| PHONE | | 60 |
| ELECTRIC | | 45 |
| WATER | | 35 |
| GAS | | 100 |
| | | 240 |
| INSURANCE | 100 | |
| CAR | | 40 |
| HOUSE | | 60 |
| | | 100 |
| CABLE TV | 25 | |
| BILL | | 25 |
| | | 25 |
| FOOD | 195 | |
| GROCERIES | | 75 |
| DINE OUT | | 100 |
| | | 175 |
| MEDICAL | 40 | |
| DENTIST | | 40 |
| | | 40 |
| AUTO | 75 | |
| GAS | | 50 |
| MAINTENANCE | | 60 |
| | | 110 |
| ENTERTAINMENT | 51 | |
| NEWSPAPERS | | 7 |
| MAGAZINES | | 12 |
| MOVIES | | 12 |
| MISC. | | 20 |
| | | 51 |
| CLOTHES | 50 | |
| JEANS & SHIRT | | 50 |
| | | 50 |
| MISC. EXPENSES | 40 | |
| HAIRCUT | | 18 |
| | | 18 |

one screen. Press the (SPACE BAR) to see more, (ENTER) or (RETURN) to select an item to change. When all of the items have been listed (or if you change an item description or value), you will be asked whether you would like to make another change.

### 2) REPORTS

Selecting this option will take you to another menu screen to select the report type:

```
1) ALL CATEGORIES AND ITEMS
2) MONTHLY SUMMARY
3) PRINT CHART
4) EXIT
```

These report options allow you to print to the screen or to a printer. When the screen is used to output a report, you are occasionally asked to press (ENTER) or (RETURN) to continue to the next screen.

### All Categories and Items

This report will generate a formatted listing of every category, and all expenses or income items assigned to each category. Below each category's list of itemized values is the total value of expenses in that category. That is the number which should be compared to the value next to the category description on the first line of each category listing. This option is illustrated in Figure 1. A Monthly Summary Report is always printed following this report.

### Monthly Summary Report

This report is printed at the end of the other reports, but it can also be printed by itself with this option. This is a short report which prints the total budgeted amount and total expenses. They are compared to the income total, and the difference is calculated. As you can see in Figure 2, our income exceeds our expenses by $291 in this first month, and we actually managed to save 7 more dollars from what we had budgeted. At this rate, in six months we will have saved $1746! By sticking to our budget, we can count on spending early November in Bermuda.

FIGURE 2

```
              Budget Summary

   BUDGET INCOME:                 1500
   ACTUAL INCOME:                 1500

   ACTUAL EXPENSES:               1209
   CASH BALANCE:                   291

   BUDGETED EXPENSES:             1216

   DEVIATION FROM
   BUDGETED EXPENSES:                7
```

### Print Chart

This option will print a simple chart for each category. The chart is printed horizontally, with the letter B repeated in a bar to indicate budgeted amounts. If the expenses for a category are within the budget, an E is printed in bar form for the expenses bar. Otherwise, if the expenses are more than what was budgeted, asterisks are used for the bar.

### 3) BAR CHART

This option allows you to visually evaluate how well you are doing with your budget. After a brief pause while the computer calculates the totals, a bar graph of your budget and expenditures will be displayed for each category. The top bar for each category indicates how much you allotted for it in your budget. The bottom bar shows your actual expenses for the month.

### 4) FILES

Here you may load and save your budget information to disk (with all 4 versions) or tape (Commodore 64 and TI-99/A versions only). The steps involved in saving or loading your data file will vary from machine to machine. Check your system's manual.

### 5) EXIT PROGRAM

Before the program halts, you will be given one last chance to return to the main menu. A message will warn you that any data in memory will be lost if you stop the program, and you will be asked to reaffirm that you want to exit. Press Y to stop the program, or N to return to the main menu and continue without losing data.

The Apple *II* Family computers have a large number of ROM (Read Only Memory) routines which can be accessed from BASIC with the CALL command. Two of these routines are used in *Budgetron*. The HOME command allows you to clear the entire screen, but if you want your program to clear a single line from the screen, then CALL -868 is just what the doctor ordered. This accesses a machine-language routine contained in the system's ROM. The routine will clear the current screen line of text from the cursor position to the right edge of the screen. This is most useful when returning to an INPUT statement because the first entry was illegal. On the *II*e and *II*c you can do this at your keyboard by pressing (Esc) E.

A second routine, CALL -958 will clear all of the screen to the right and below the current cursor position. All text above, and to the left of the cursor will not be affected. This command is extremely useful, and it saves a considerable amount of time and programming effort. If you have a data-entry area, or options displayed at the bottom of the screen, you can easily clear them without having to clear the entire screen.

A third call to -936 is not used in this program. This call will erase the entire screen just as the HOME command does.

The Commodore 64 version of this program utilizes a data-compaction algorithm to place in a single string all of the information for an item, such as its category, description, and value. Each string is made up of four parts. The first two parts occupy only one character each. The first part, category, is the category number with which the item is associated. It is a single character. Subtracting 100 from the ASCII code of this character will yield the category number. The second byte of the string contains the length of the description. Subtracting 200 from the ASCII code of this character yields the number of characters in the description. The value of the item follows the description in the string. The length of the description—derived from the second character in the string—is then used to locate where in the string the value for the item resides.

Format for string:
Category # / Description Length / Description / Value

The items are all stored in the A$( ) array, and the data for the array is constructed in lines 1480 through 1500. The variables used in the construction are as follows: ZZ = item number; S$ = item description; C = category; V = item value.

The reporting features of the IBM PC and PCjr version are very handy. You can either print a report to the screen for a quick review, or get a hard copy of it on the printer. Fortunately this can all be handled by the same print statements. The IBM PC and PCjr allow you to open a file which outputs to the screen just as if it were a printer. However, this does not work if you want to use the LPRINT statement to output data to the system's printer. The program needs to open a file with either SCRN: to send output to the screen, or LPT1: to send it to the printer. After opening the file, you can print and the system doesn't care whether the data is going to the printer or to the screen.

Ever notice the arrow used in many of the continue prompts that stand for the (ENTER) key? It is constructed from two of the IBM's standard graphics characters. In this program, we have assigned these two characters to the variable ENT$.

```
ENT$ = CHR$(17) + CHR$(217)
```

This presents a problem on the PC, however, because the graphics characters with ASCII values above 127 are not available when using graphics screens 1 or 2 (this may vary depending on the type of graphics card you are using). These characters are always available on the PCjr. To solve the PC problem, we have designed a graphics string which is drawn with the DRAW command. Whenever the shape is needed (only on the graphics bar-chart screen in this program), you can simply draw it on the screen. The string is defined in line 1390 and then used in line 1630. ENTDR$ contains the graphics string.

When developing a program which prints a report of some form—such as this one for the TI-99/4A—you may often have the need to print that same form to the screen as well. That way you can get a quick review of the data before committing to a hard copy. In addition, if you are developing the software to be marketed, or if it is intended for use by other people, you may want to attract the largest audience possible. But many potential software buyers may not own a printer, and would not reap the benefits of the printed report. This means you would have to write a completely different section in the program to handle screen output as well as printer output.

A little-used and seldom-discussed function of the TI-99/4A computer may be able to save you hours of frustration and headaches: Just open a file with channel 0. When the report gets ready to print, you can provide an option which asks for output to the printer or the screen. You may then set a variable to 0 if the screen is selected, or to some other file number if the printer is selected. You will need a little logic to branch around any OPEN or CLOSE statements if the channel variable is set to 0 (this is illegal for those statements). In this program the variable F is used. When you print to the file, use the variable after the # symbol instead of a number:

```
PRINT #F:"THIS GOES TO THE SCREEN IF F=0"
```

**HCM**

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.**

### ***Budgetron*** **(Apple *II* Family)**
### **Explanation of the Program**

| **Line Nos.** | |
|---|---|
| 100-200 | Program header. |
| 210-610 | Initialization and main menu. |
| 620-1780 | Menu selection screens. |
| 1790-2400 | Work with the budget. |
| 2410-3080 | Work with items. |
| 3090-3150 | Select a category. |
| 3160-3680 | Print reports. |
| 3690-3780 | Display categories. |
| 3790-4090 | Number-entry routine. |
| 4100-4220 | Calculate totals. |
| 4230-4310 | Turn on the printer port. |
| 4320-4410 | Get input for menu selections. |
| 4420-4590 | Error routine. |
| 4600-4770 | Delete routines. |
| 4780-4990 | File-name-entry routine. |

### ***Budgetron*** **(IBM PC & PCjr)**
### **Explanation of the Program**

| **Line Nos.** | |
|---|---|
| 100-210 | Program header. |
| 220-310 | Initialization and main menu. |
| 320-490 | Work with budget. |
| 500-660 | Work with items. |
| 670-780 | Category routines. |
| 790-820 | Display prompts for another. |
| 830-1000 | Change-items routine. |
| 1010-1090 | Delete routines. |
| 1100-1650 | Reports. |
| 1660-1830 | Files routines. |
| 1840-2140 | Misc.-input routines. |
| 2150-2200 | Print-categories routine. |
| 2210-2230 | Exit-program routine. |
| 2240-2290 | Data for menu screens. |
| 2300-2360 | Error routine. |

### ***Budgetron*** **(C-64)**
### **Explanation of the Program**

| **Line Nos.** | |
|---|---|
| 100-200 | Program header. |
| 210-360 | Initialization and main menu. |
| 370-650 | Menu selection screens. |
| 660-1070 | Work with the budget. |
| 1080-1540 | Work with items. |
| 1550-2030 | File routines. |
| 2040-2200 | Display bar chart. |
| 2210-2730 | Report routines. |
| 2740-2860 | List-categories routine. |
| 2870-2940 | Data-entry routines. |
| 2950-2960 | Beep-and-pause routine. |
| 2970-3020 | Exit program. |
| 3030-3310 | Misc.-input routines. |
| 3320-3330 | Move-cursor routine. |
| 3340-3460 | Delete routines. |

### ***Budgetron*** **(TI-99/4A)**
### **Explanation of the Program**

| **Line Nos.** | |
|---|---|
| 100-200 | Program header. |
| 210-670 | Initialization and main menu screens. |
| 680-1460 | Work with the budget. |
| 1470-2250 | Work with the items. |
| 2260-2640 | Display-the-chart routine. |
| 2650-3530 | Reports routines. |
| 3540-3880 | Load and save data routines. |
| 3890-3970 | Routine to initialize expenses. |
| 3980-4010 | Key-input routine. |
| 4020-4160 | Get totals for each category. |
| 4170-4320 | Validate numeric entries. |
| 4330-4360 | Time delay with a beep. |
| 4370-4420 | Routine to exit the program. |
| 4430-4440 | Graphics character data. |
| 4450-4710 | Delete categories and items. |

# Over-Reaction

by HCM Staff

*A nuclear reactor is raging out of control; the world is on the brink of war—earthquakes, riots, and sabotage are the order of the day—and you are the one at the controls . . .*

*Over-Reaction* is a game that simulates the operation of an electric-power-generating nuclear reactor. It's the player's job to keep the reactor operating smoothly, producing a steady supply of electricity at a frequency of 60 cycles per second (Hz). If the frequency deviates by more than 2 Hz (below 58 or above 62), the reactor automatically "trips" (shuts down), ending the game.

The reactor's power output is controlled by 24 fuel rods set up in a 4-row by 6-column grid; each rod is designated by a letter (except in the Apple version, which uses numbers). The grid is divided into four quadrants—each has a temperature gauge which monitors the average temperature of the quadrant. By pressing the proper key, you will push in a rod—this tends to raise the temperature around the rod. By pressing the same key with the shift key depressed, you will pull out the rod, lowering the temperature. The overall temperature of the entire grid controls the reactor's steam pressure, which drives a turbine to produce electricity.

## Built-In Variations

As the steam pressure fluctuates, so does the frequency of the electricity you are producing. It is therefore your job to keep a constant steam-pressure level. If the temperature in one quadrant drops too low or increases too high, you will hear a warning buzz. If it goes outside of a safe range, it will cause an automatic shut-down or trip—costing you points and ending the game until the reactor can be repaired.

You must also deal with the effects caused by changes in load demand. If the load demand increases, it tends to pull down the system, and drop the frequency. If the frequency begins to rise unexpectedly, it probably means that the load demand has dropped, and you will need to decrease the overall temperature of the reactor to compensate. You will find that as in real life, the system won't react to your control immediately—it takes time for your actions to take effect.

**Photo 1.** *This screen from the Apple version shows the normal operating conditions of the reactor. Note the use of numbers rather than letters to indicate the control-rod drives.*

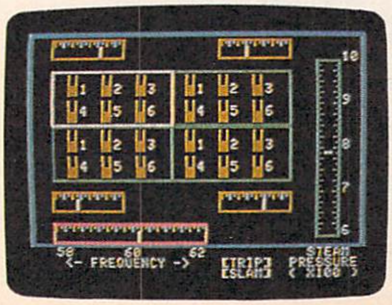


**Photo 2.** *This screen from the C-64 version shows a broken temperature gauge. Notice the temperature variations in each quadrant. Try to maintain uniform temperatures.*

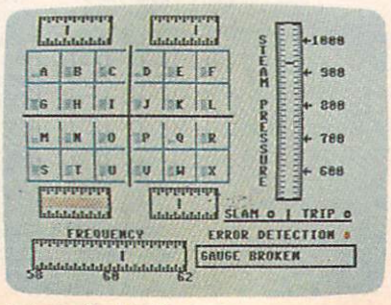
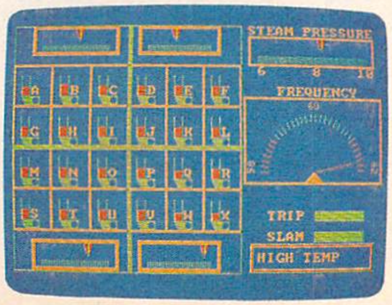


**Photo 3.** *This screen from the IBM version shows a high temperature warning. Notice that the frequency gauge is dangerously high. Check the 4 quadrants for high temperatures and adjust the control rods accordingly.*



## Controlling the Power Output

The difficulties you will encounter vary according to skill level:

1) Normal operation
2) Faulty systems
3) Earthquake country
4) Armageddon

In the first level, you merely have to watch for minor temperature fluctuations and small changes in the load. This mode is best for training, so that you will be prepared to handle the higher levels. Experiment with raising and lowering the control rods—watch how long it takes for your action to cause the reactor to change. Once you feel like you can control the power output, you are ready to tackle the higher levels.

> *"You will find that the system does not react immediately . . . it takes time for your actions to take effect."*

The second level introduces more problems. The temperature here can fluctuate more radically, and the load can increase or decrease as well. Also, any one of your four temperature gauges can break at any time, making it impossible to monitor the temperature of a quadrant. Photo 4 displays an example of some fluctuations. When a warning sounds, you must be prepared to take corrective action where needed.

At level 3, you're faced with all of the problems that can occur in level 2, plus steam leaks caused by earthquake tremors. Here you'd have to increase the temperature to compensate. Photo 5 illustrates what can happen if you don't react fast enough to correct a problem. In this example, the overall temperature of the reactor dropped low enough that the frequency level fell below 58, causing the reactor to trip off-line.

Finally, at level 4, a nuclear war is imminent—riots, sabotage, and even more drastically fluctuating loads will plague you. At this level, be prepared to quickly make adjustments to changes that you see on the gauges. But remember, it takes a little time for the reactor to "react" to your manipulations of the control rods. Before you play the game, review *all* of the accompanying photo examples (regardless of which machine is indicated) to familiarize yourself with some of the different problems that can arise during the reactor's operation.

One final tip for being more successful at the control panel: learn to identify and listen for audible warnings—steam leaks, high temperature warnings, gauges breaking, etc. And, if you should realize that you have overreacted, or that you just can't bring the frequency levels into the safe range and a meltdown is inevitable, fear not. Just push the slam button ((FUNCTION) 1 on all but Apple systems, where it is the (ESCAPE) key). This will keep your score from being lowered by a forced shutdown of the system.

The original, Apple *IIe* version of *Over-Reaction* was designed for use with shifted alpha characters (lowercase). Unfortunately, the Apple II+ does not have this capability. Thus, the Apple *II* Family version of this program is controlled in a slightly different manner. You can select a quadrant with the keys A, B, C, and D. The active quadrant will be highlighted by a color different from those not active; it will stay active until you select another quadrant. There are six control rods in each of four quadrants in the control panel. You can access the rods in the active quadrant by pressing the numbers 1 through 6. Pressing the unshifted numbers will push in the rod. If you press a shifted number, the rod will be pulled out.

The Commodore 64 contains an extremely versatile set of built-in graphics characters, and this program takes full advantage of them. The temperature, steam pressure, and output frequency meters use these characters to position their needles. Two strings are used to contain the desired characters: L$ is used for the horizontal needles of the temperature and frequency meters; the string U$ is used for the steam pressure meter. Each string contains all possible needle positions that can occur within a character. When the meter setting is evaluated, the position of the character desired is calculated and extracted with the MID$ function. The character is then printed on the screen as the new meter position.

The IBM PC and PCjr version of *Over-Reaction* uses two methods to display the meter needles on the control panel. The temperature and steam pressure indicators both use a sliding-scale meter which moves horizontally, both left and right. The needle is displayed using the PUT command. This command allows you to display a graphics shape that was previously saved into an array using the GET command. The PSET option is

***Photo 4.*** *This screen from the C-64 version shows power fluctuations—two temperature gauges are broken. Fluctuations will make the frequency level behave erratically, so try to anticipate "hot" spots.*

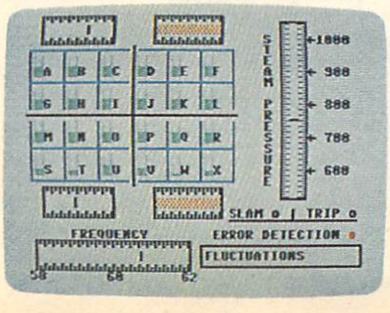


***Photo 5.*** *This screen from the IBM version displays a bad-frequency warning. The frequency level is extremely low; too low—the reactor has tripped. At this point, the game ends and you can try again next time.*

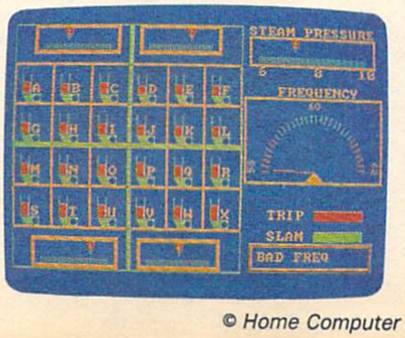
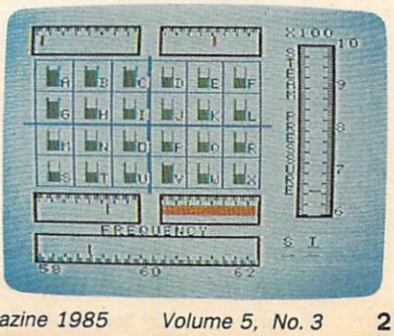


***Photo 6.*** *This screen from the TI version shows normal reactor operations, but notice the faulty temperature gauge. Try to keep the frequency as close to 60 Hz as possible.*



used with an area for the needle which extends to the left and right of the needle itself. In doing this, the needle will erase its old image every time it is moved. This method produces much less "flicker" than the default method of erasing the old image with the XOR option. The frequency meter looks like an old-fashioned analog meter, with a long needle sweeping across a semicircular dial. The needle produced here is drawn with the LINE command. The program will redraw a new line only if a calculation shows that the needle position has changed. NOTE: the (CAPSLOCK) key *must not* be activated before and while you run the program. Also, do not hold down a key after the acceptance beep—it may cause the keyboard buffer to overflow, which may result in erratic program behavior.

The TI version of *Over-Reaction* takes advantage of the 99/4A's ability to redefine character graphics, producing what appears to be a high-resolution motion of the indicator needles used in the the control panel's instruments. Only four characters were needed to define the horizontal meter needle, and four more for the vertical meter needle. A calculation determines which character is required for a given position in the meter. The character is then placed on the screen using the HCHAR command.

Eight characters were defined for the control rods. There are sixteen positions in each control rod, requiring that two characters be used. These algorithms can easily be employed in a multitude of applications where the relative position of a value is more important than the value itself. As you may know, it takes quite a while to display messages or numbers on the screen by using TI BASIC without scrolling the screen. Although this program doesn't burn up the track with speed, you will nevertheless be impressed with the amount of activity going on, and the speed with which the program executes.

### *Over-Reaction* (Apple *II* Family) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-200 | Program header. |
| 210-230 | Relocate program above hi-res screen. |
| 240-350 | Initialize the program. |
| 360-390 | Get skill level. |
| 400-570 | Draw the control panel. |
| 580-630 | Initialize the variables. |
| 640-790 | Main control loop. |
| 800-940 | Select-problems routine. |
| 950-1110 | Start-of-game messages. |
| 1120-1370 | End-of-game messages. |
| 1380-1420 | Controlled shutdown—slam. |
| 1430-1480 | Control updating gauges. |
| 1490-1520 | Meltdown routine. |
| 1530-1560 | Update steam gauge. |
| 1570-1600 | Update frequency meter. |
| 1610-1720 | Scan keyboard to control the reactor. |
| 1730-1780 | Update rods & block temperature. |
| 1790-1800 | Draw a box routine. |
| 1810-1820 | Routine to flash the screen. |
| 1830-1970 | Program DATA. |

### *Over-Reaction* (C-64) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-190 | Program header. |
| 200-390 | Initialize program. |
| 400-450 | Skill-level option screen. |
| 460-700 | Start-of-game messages. |
| 710-1210 | Draw the reactor control panel. |
| 1220-1250 | Initialize variables for a new game. |
| 1260-1390 | Main control loop. |
| 1400-1530 | Determine types of problems. |
| 1540-1620 | Scan keyboard. Reactor control. |
| 1630-1660 | Position the reactor rod graphics. |
| 1670-1740 | Update the block temperature. |
| 1750-1810 | Trip and slam routines. |
| 1820-1910 | Update temperature gauges. |
| 1920-1960 | Update steam pressure meter. |
| 1970-2000 | Update frequency meter. |
| 2010-2080 | Error-condition routines. |
| 2090-2440 | End-of-game messages. |
| 2450-2460 | Move-cursor routine. |
| 2470-2680 | Program data. |

### *Over-Reaction* (IBM PC and PCjr) Explanation of the Program

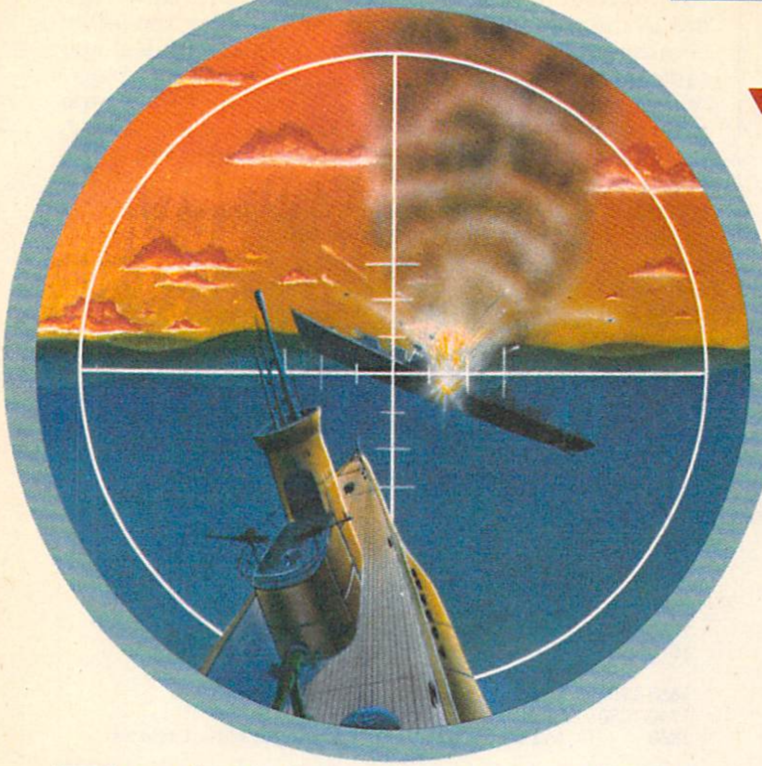
| Line Nos. | |
|---|---|
| 100-220 | Program header. |
| 220-340 | Initialize the program. |
| 350-430 | Main control loop. |
| 440-470 | Branch to problems for skill level. |
| 480-500 | Temperature meter malfunctioned. |
| 510-520 | Cooling system malfunctioned. |
| 530 | Control problems for levels 2 and 3. |
| 540 | Steam leak develops. Pressure drops. |
| 550-600 | Scan the keyboard. Reactor control. |
| 610-650 | Move rods in or out. |
| 660-720 | Update the block temperature meters. |
| 730-790 | Update steam pressure meter. |
| 800-850 | Update frequency meter. |
| 860-1010 | End-of-game messages. |
| 1020-1150 | Get skill level. Display messages. |
| 1160-1310 | Draw the control panel. |
| 1320-1350 | Routine to scan the keyboard. |
| 1360-1400 | Coordinate the rods. |

### *Over-Reaction* (TI-99/4A) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-200 | Program header. |
| 210-310 | Select skill level. |
| 320-890 | Initialize program. |
| 900-1140 | Main control loop. |
| 1150-1410 | Select problem routine. |
| 1420-1650 | Start-of-game messages. Determine skill level and problems. |
| 1660-1980 | End-of-game messages. Replay option. |
| 1990-2090 | Slam routine. |
| 2100-2210 | Update temperature meter. |
| 2220-2290 | Update steam-pressure meter. |
| 2300-2370 | Update frequency meter. |
| 2380-2660 | Key input routine-position rods. |
| 2670-2770 | Update block temperature. |
| 2780-2800 | Key-scan routine. |
| 2810-2840 | Routine to update graphics position. |
| 2850-2980 | Character graphics & color DATA. |
| 2990 | DATA for meter position array BP( ). |
| 3000-3070 | Screen display format DATA. |

HCM

For your key-in listings, see HCM PROGRAM LISTINGS Contents.



# TORPEDO ALLEY

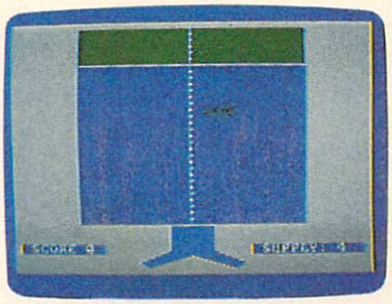
**by William K. Balthrop**
*HCM Staff*

***War has traveled to the far reaches of the Earth—even to the depths of the sea. How long will you survive in this underwater game of cat and mouse?***

From deep in the tumbling sea, your submarine rises to peek above the surface. A single warship enters from the edge of your viewing field. As it approaches the vertical crosshair at the center of your scope, you issue the order: *Fire Torpedo!* You see the wake of the first shot climb along the center line, diminishing in the distance as it nears its deadly rendezvous. Finally, a flash of flame and following thunder announce the demise of yet another enemy vessel. But the next ship to appear zig-zags around your second torpedo, and swings around on a collision course as its guns pump out deadly salvos and depth charges are made ready. There's no time to waste—*Dive! Dive!*

*Torpedo Alley* is a game of timing and wits. The object of the game is to sink as many enemy ships as you can with your limited supply of torpedoes. But watch out—the enemy can shoot back. Fortunately, at any time you may submerge and take evasive actions.

But submerging has its drawbacks as well. Your sub is not fast when fully submerged, and the enemy ship drops depth charges. You must try and maneuver your sub around the charges or you will be blown-up. If you survive the depth charges, you can resurface and take another stab at the ships with your torpedoes.



*Screen photo taken from C-64 version.*

### Fire Away

Start the game by deciding whether to use joysticks. If you do use joysticks, the following actions will result:

**Main Screen - Up Periscope**
**Left**—Move sub to the left.
**Right**—Move sub to the right.
**Down**—Dive; down periscope.
**Fire button**—Fire torpedo.

**Second Screen - Submerged**
**Left**—Move left.
**Right**—Move right.
**Down**—Move down.
**Up**—Move up.

If you select the keyboard option, you can consult your machine's Control Capsule for the proper movement control keys.

After you select the joystick or the keyboard, the screen will clear and the main game screen will appear. This screen displays the view of enemy waters as seen through your periscope. Occasionally, an enemy vessel will cross your path. You must try to judge the precise time to fire your torpedo, based on the ship's distance, and its direction of travel. The destroyers are aware that you are in the area, so most of them will be navigating a zig-zag pattern. This makes them a more difficult target because distance is harder to judge.

The enemy ships will fire at your sub if you allow them to pass beyond the center of your periscope. You can turn your sub either left or right to try and line up the ship by using either the keyboard controls or the joystick. If you are unable to get off a good shot before the enemy ship reaches the other side of your periscope, your only hope of avoiding a hit from its main armament is to dive.

When you dive, the screen will redraw, showing your sub diving below the surface of the water. A ship will then start traversing the top, dropping depth charges. Resurfacing after out-maneuvering all of the depth charges is automatic; at this point, you needn't do anything. After you surface, the main screen will display again, and you will be ready to fire your torpedoes at the next ship.

Every time you are hit by the main armament from the enemy's ships, some of your torpedos become damaged, lowering your supply. When your supply of torpedoes reaches zero, or when you are destroyed by depth charges, the game is over. Your score and percentage of hits versus shots taken will be displayed. You will then be given a chance to play another game.

### CONTROL CAPSULE
*Torpedo Alley*

| KEY | FUNCTION |
|---|---|
| *Surface Mode:* | |
| J | Move periscope left. |
| K | Move periscope right. |
| M | Dive. |
| [Space bar] | Fire torpedo. |
| *Submerge Mode:* | |
| I | Move sub up. |
| J | Move sub right. |
| K | Move sub left. |
| M | Move sub down. |

The secret to good animation on the Apple computer lies in the use of the XDRAW command. Like the DRAW command, XDRAW places shapes defined with a shape table onto the high-resolution graphics screen. XDRAW will reverse the color of the screen wherever the shape table indicates that a pixel should be plotted. By XDRAWing at the same location twice, you in effect erase the shape drawn by the first XDRAW. This method leaves the contents of the screen undisturbed after the shape is erased, making animation possible.

One drawback to this method of animation is that, if not done properly, the shape you are drawing can flicker—an undesirable effect when doing animation. This problem can be improved by remembering the last position of the shape and erasing it just before drawing the shape at its new location.

### *Torpedo Alley* (Apple *II* Family)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-210 | Program header. |
| 220-380 | Initialize program variables. |
| 390-410 | Load shape table and sound routine. |
| 420-490 | Display periscope control room. |
| 500-540 | Main control loop for the periscope; part 1 of game. |
| 550-580 | Place a ship in the periscope. |
| 590-720 | Scan keyboard and joystick. |
| 730-840 | Move the ship in the periscope. |

| Line Nos. | |
|---|---|
| 850-990 | Move the torpedo when fired. |
| 1000-1040 | Hit the ship with a torpedo. |
| 1050-1140 | End of the game. Option to play again. |
| 1150-1560 | Submerged game routine; part 2. |
| 1570-1650 | Ship fires and hits your sub. Lose supply of torpedoes. |
| 1660-1750 | Sound-effects routines. |
| 1760-1850 | Shape-table data. |
| 1860 | Machine-language routine for sound effects. |

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.**

### CONTROL CAPSULE
*Torpedo Alley*

| KEY | FUNCTION |
|---|---|
| *Surface Mode:* | |
| S | Move periscope left. |
| D | Move periscope right. |
| X | Dive. |
| [Space bar] | Fire torpedo. |
| *Submerge Mode:* | |
| E | Move sub up. |
| S | Move sub right. |
| D | Move sub left. |
| X | Move sub down. |

The Commodore 64 is capable of producing very good animation with the use of sprites. Sprites are movable objects which can pass behind or in front of screen character graphics. These sprites, however, are not able to move on their own. You must track each sprite and POKE its new location into memory. This method has its good and bad points. The fact that you have to move each sprite manually can really start to eat up processor time if you are moving very fast or are moving more than one object. On the plus side, however, you always know exactly where each sprite is—no guesswork is involved. You may see the game slow down a bit when more than one sprite is moving on the screen, such as when the ship is dropping depth charges on your sub.

### *Torpedo Alley* (C-64)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-200 | Program header. |
| 210-300 | Title screen. |
| 310-430 | Initialization. |
| 440-980 | Display playing screen. |
| 990-1030 | Main control loop. |
| 1040-1110 | Start and move the ship. |
| 1120-1250 | Scan keyboard and joystick. |
| 1260-1380 | Move islands across screen. |
| 1390-1460 | Ship shoots back. |
| 1470-1540 | Fire torpedo at ship. |
| 1550-1610 | Check for a hit. |
| 1620-1660 | Ship hit by a torpedo. |

| Line Nos. | |
|---|---|
| 1670-1800 | Dive routine. |
| 1810-2060 | Main control loop for submerged portion of game. |
| 2070-2090 | Explode depth charge. |
| 2100-2120 | Hit by depth charge. |
| 2130-2260 | Surfacing routines. |
| 2270-2280 | Select an island shape for display. |
| 2290-2360 | End of game. Option to play again. |
| 2370-2380 | Move-cursor routine. |
| 2390-3110 | Sprite shape data. |
| 3120-3160 | Sound data. |

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.**

### CONTROL CAPSULE
*Torpedo Alley*

| KEY | FUNCTION |
|---|---|
| *Surface Mode:* | |
| Left arrow | Move periscope left. |
| Right arrow | Move periscope right. |
| [ENTER] | Dive. |
| [Space bar] | Fire torpedo. |
| *Submerge Mode:* | |
| Up arrow | Move sub up. |
| Right arrow | Move sub right. |
| Left arrow | Move sub left. |
| Down arrow | Move sub down. |

The IBM PC and PCjr have a number of graphics-drawing commands which are used in games requiring a lot of animation. The DRAW command is capable of creating intricate designs on the screen with a minimum of overhead. (For complex animation though, this command has its limitations.) Once a shape is drawn with this command, you can store the shape in an array using the GET command. With the shape in an array, you can easily put it back on the screen anywhere you like with the PUT command. The default mode of the PUT command is XOR. This option will cause the shape to inverse the color of any pixel on the screen that comes in contact with a pixel in the shape. This allows the program to erase the shape by simply drawing at the same location again.

### *Torpedo Alley* (IBM PC & IBM PCjr)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-220 | Program header. |
| 230-320 | Initialize program graphics and variables. |
| 330-370 | Set up for start of a new game. |
| 380-420 | Draw the periscope control room. |
| 430-470 | Main control loop for periscope. |
| 480-510 | Ship hit by torpedo. |
| 520-560 | Ship fires back and hits you. |
| 570-620 | End-of-game routine. |
| 630-670 | Scan keyboard. |
| 680-890 | Submerged game routine. |
| 900-980 | Move the islands and adjust ships speed. |
| 990-1120 | Fire and move the torpedo. |
| 1130-1200 | Move ship through periscope. |
| 1210-1250 | Input routine and score-display routine. |

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.**

### CONTROL CAPSULE
*Torpedo Alley*

| KEY | FUNCTION |
|---|---|
| *Surface Mode:* | |
| S | Move periscope left. |
| D | Move periscope right. |
| [ENTER] | Dive. |
| [Space bar] | Fire torpedo. |
| *Submerge Mode:* | |
| E | Move sub up. |
| S | Move sub right. |
| D | Move sub left. |
| X | Move sub down. |

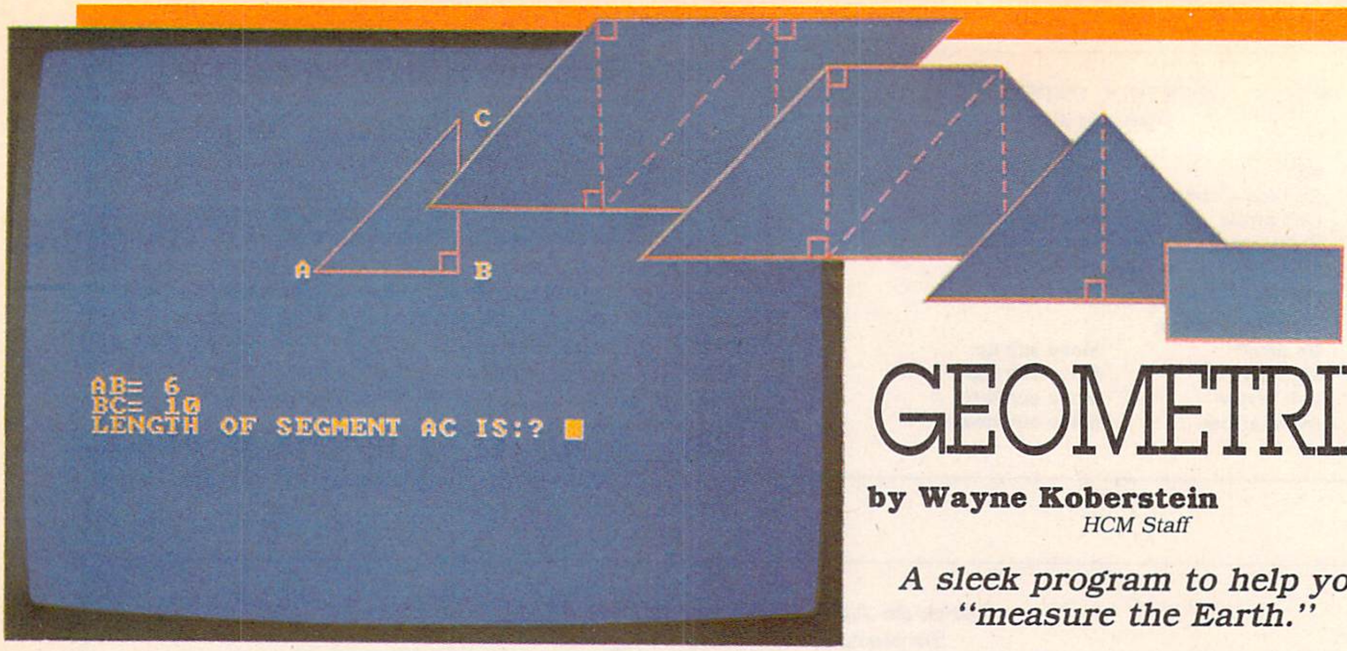
Animation on the TI-99/4A is easy with the use of sprites. With their automatic motion, you can have the program set them off on a course while it works on other tasks. You can even change the shape of a sprite while it's in mid-motion—this is how this program simulates the torpedo and its trail of white water getting smaller as it moves farther away. A loop which starts in line 660 reads the shapes from a DATA statement as they are needed. Because the sprites get their shape from redefining the shapes of normal characters, the CHAR command is called. This method is not as fast as first defining all of the shapes needed, and then reassigning a different character to the sprite. However, when a large number of graphics characters are required for either the animation or other parts of the program, this is the only method available.

### *Torpedo Alley* (TI-99/4A)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-190 | Program header. |
| 200-260 | Display title & option screen. |
| 270-380 | Display the main playing screen. |
| 390-410 | Main control routine—part 1. |
| 420-440 | Place a ship in the periscope. |
| 450-500 | Scan for input. |
| 510-600 | Check ship and island position. |
| 610-730 | Fire torpedo at the ship. |
| 740-800 | End-of-game routine. |
| 810-860 | Select one of four island shapes. |
| 870-920 | Dive routine. Submerge below water. |
| 930-960 | Main control routine—part 2. |
| 970-1020 | Move sub on screen 2. |
| 1030-1130 | Update depth charges. |
| 1140-1200 | Sub surfacing routine. |
| 1210-1280 | Ship shoots back at you. |
| 1290-1480 | Program data for graphics. |
| 1490-1510 | Explosion sound effects. |
| 1520-1550 | Update ships motion. |
| 1560-1570 | Subroutine to create the sonar sound. |

XB *Torpedo Alley* requires TI Extended BASIC.

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.** HCM



```
AB= 6
BC= 10
LENGTH OF SEGMENT AC IS:? ■
```

# GEOMETRIX

**by Wayne Koberstein**
*HCM Staff*

***A sleek program to help you "measure the Earth."***

How high is that tree? How long is that rafter? How much fence do I need to keep these animals enclosed? How many square feet in this oddball-shaped yard? These are the kinds of questions that long ago prompted humans to place a mental grid over their picture of the real world. They used this method of measuring the earth to build pyramids, navigate, and to create a whole abstract mathematical system based on simple two-dimensional shapes—a system the Greeks called geometry. But the practical reasons for this theoretical base are still with us today: We still need to "measure the Earth."

*Geometrix* is designed to administer problems dealing with the basic shapes of plane geometry. It is both an educational and a practical tool. Four different types of shapes are used by the program: rectangles, right triangles, parallelograms, and trapezoids. Each shape has its own unique set of mathematical relationships expressed by a given formula. In this text, we provide the basic formulas for rectangles and right triangles. Parallelograms and trapezoids are combinations of these first two shapes—so to solve these more complicated shape problems, you will use the skills that you've already learned from the simpler exercises. In working with each problem, you may use a small calculator for the arithmetic involved; but you will also have to employ your powers of logic to arrive at a correct answer. The computer's job is to supply random problems and check for errors.

Although the shapes that the program displays appear to be abstract, they have a real practical value. These shapes occur throughout nature, and in man-made materials and objects. Trees and seashells, as well as buildings, cars, and telephones, are all examples of complex structures made up of simple geometric shapes. With *Geometrix*, you can discover how these basic shapes serve as building blocks for more complicated forms. And, conversely, you will learn how to see intricate objects and relationships in terms of their basic shape components. The sidebar "Geometrix at Home" provides some "real-world" applications for the shapes and formulas included in this program.

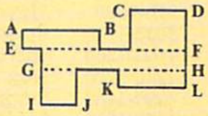
## Getting With The Program

For each basic shape, there are three levels of difficulty. Each succeeding level supplies a more complex problem, often requiring you to solve several smaller problems in order to come up with a final answer. (An increase in difficulty also occurs as you move to the more complex shapes: parallelograms and trapezoids.) These exercises may at first seem almost *too* simple—but the lower-level problems have a purpose that will become

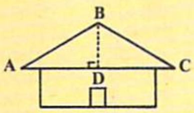
## *Geometrix* At Home

All of the basic shapes we investigate with this program have their applications in everyday life. Here are just a few examples:

**1. Rectangles**—Often when we deal with a rectangular surface, we need to find the area of that surface. Suppose you have an odd-shaped yard covered with grass (or wheat, etc.). As with our drawing, the boundary of your yard takes many 90-degree turns. How do you find the area of such a seemingly complicated shape? By refering to the Level 3 exercise for rectangles, you can see that one shape may consist of many component rectangles—as indicated by the dotted lines in this drawing. Simply figure the area of each rectangle and add up the total.



**2. Right Triangles**—In nature, triangles are fundamental—they abound everywhere. Although not all triangles are *right* triangles, any triangle can be analyzed as a right triangle by drawing a perpendicular line from its base to its apex—as represented in drawing 2 by the dotted line, BD. This line—often called the *height*—divides any triangle into two right triangles. A carpenter uses this feature in computing the length of rafters for a roof frame. If you see the "gable" end of a house as a triangle like the one shown here, the rafters would correspond to the segments AB and BC. Knowing the width of a given side (either AD or DC) and the height from roof-base to roof-top (BD) is sufficient to determine the rafter length.



apparent as you progress. These earlier exercises merely pound the stakes for the more complicated constructions.

The computer will check each answer, which must be rounded to the second decimal place. If you give three incorrect answers, the program supplies the correct solution. After completing a given exercise, you have a chance to enter your own variables—thus creating your own problem, which the computer will check for errors. Because each exercise uses a specific shape, each variable has a certain range of legal values. Entering an illegal value will cause illogical results—such as negative numbers—or other unexpected effects on the program. Part of the exercise entails figuring out what the legal range should be in advance of actually entering a value.

All of the basic shapes as they appear on the screen remain constant—they do not change their apparent dimensions as the stated values change. Each screen figure serves not as a literal representation of the dimensions given in a specific problem, but only as reference. This places even more emphasis on logic, rather than visual measurement or guess.

1) Rectangles
2) Right Triangles
3) Parallelograms
4) Trapezoids

### Main Menu

The main menu will allow you to choose the type of shape you would like to work with.

After selecting a shape, you must select one of three difficulty levels (Level 1 is the easiest). The following is a breakdown of each shape, and the problems that each level will supply.

For clarity, let it be understood that two capital letters placed together represent a line segment between the points indicated by each letter. For example, AB represents a line segment between point A and point B. Also, the number values used in each problem are "generic" units; you could substitute any standard measuring unit and apply it to any problem.

## 1) Rectangles

Level 1

This problem presents a simple square. A square is defined as having four sides, all being the same length, with the angle of the four corners being 90 degrees wide. The problem is to find the area of the square, with two sides known, using the basic formula:

AREA = AB * BC Given AB and BC

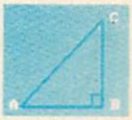
Level 2

This problem is just like that of Level 1, except that rectangle sides do not have to be of equal length. Level 2 also uses the same basic formula and the same "givens" as Level 1.

Level 3

This level introduces a more complex shape: two rectangles put together to form an upside down L shape. Four of the possible eight line lengths in this shape are given. You must find the area of the shape by analyzing it in terms of its basic components.

## 2) Right Triangles



Level 1

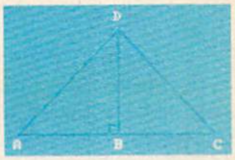
Right triangles are defined as having only three sides, with one of the angles equal to 90 degrees. Right triangles have certain properties which make calculating unknowns simple. This first level is fairly easy—you are given the two sides adjacent to the 90-degree corner: AB and BC. You must find the area of the triangle, using the basic formula:

AREA = (AB*BC)/2

Level 2

This problem uses the same shape as the one in Level 1, and asks that you find the length of the hypotenuse AC, which is the side opposite the 90-degree angle. To solve this problem, you can use the formula contained in the classic Pythagorean theorem on the relationship between the hypotenuse of a right triangle and its other two sides. Here we express this formula as:

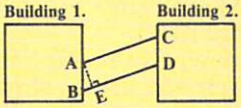
AC = √(AB^2 + BC^2) Given AB and BC.



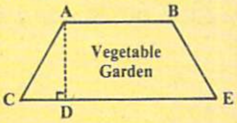
Level 3

This level is only slightly more complicated than Level 2; however, the shape has been changed to a vertically-symetrical

**3. Parallelograms**—As long as we're up on the roof, let's look at how differently shaped roof surfaces require different tactics to, say, figure the amount of material required to cover them. Suppose our carpenter has to build a roof over a walkway between two buildings. These structures sit parallel to each other; but the doorways connected by the walkway are not directly across from one another. Looking directly down on the walkway, we see that its roof forms a parallelogram. (To keep it simple, we'll say this is a flat roof.) How much area, then, does this roof cover? If you know the width of the walkway (AE), and the length of one side (AC), it is relatively easy to calculate its total area.

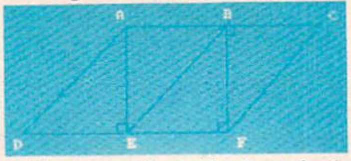


**4. Trapezoids**—Although it sounds like they might come from outer space, trapezoids actually have many down-to-earth applications. This basic shape, like the others, occurs often in all kinds of construction. But, for variety, let's consider a garden that has to fit in a space that is narrower on one end than on the other—effectively, a trapezoid. How much area can our gardener cover with vegetables in this space? Refering to the Level 1 exercise, we need only to know the length of the narrow side (AB), the long side (CE), and the height (AD).



These examples are only suggestions of what can be done with basic geometric shapes in everyday circumstances. Perhaps you can imagine—or will encounter—other similiar situations. You may even turn some of these examples around to exercise other aspects of what you've learned with *Geometrix*. For instance: What if you had to build a structure with trapezoidal sides? How would you calculate on paper the lengths of the bracing members in the side frames? Look at Trapezoid Levels 2 and 3 for some ideas on this. HCM triangle, which divides into two right triangles—each a mirror image of the other. Use the basic formula from Level 1 to solve the problem, given AB, BD, and AC.

### 3) Parallelograms



Level 1

A parallelogram has four sides. Opposite sides are always parallel with each other, and have the same length. Levels 1, 2, and 3 all require you to solve for area. To do this, you need to know the length of one side—in this case, the segment AC—and the distance to the opposite side—here, the "height" AE (a line 90 degrees from the known side to the opposite side). For Level 1, you are given AC and AE. Use the basic formula:

AREA = AC x AE

Level 2

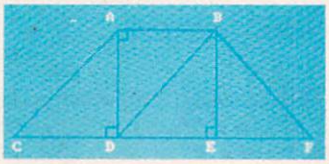
This problem is quite a bit more complex. It requires you to solve several problems to arrive at the correct answer. The unknown link in solving this puzzle is the height of the shape, segment AE. If you can find the length of that segment, you can use the formula from the first level to arrive at a solution. You are given AD, DF, and AB.

> *". . . you will learn how to see intricate objects and relationships in terms of their basic shape components."*

Level 3

This problem, like the exercise in Level 2, is more complex—requiring you to solve several problems before the final answer can be obtained. You are given EB, BF, and DE.

### 4) Trapezoids



Level 1

A trapezoid is defined as a shape with four sides, two of which are parallel. This shape poses more complex problems than the previous three. Because of the difficulties involved in solving such problems without getting into trigonometry, a few additional clues have been supplied for Levels 2 and 3. In Level 1, you must solve for the total area, given AB, AD, and CF.

Level 2

To solve this problem, you should know that the angles at points C and F are both 45 degrees. This still makes for a very challenging problem. Solve for segment AD, given AB and CF.

Level 3

This last problem is a real brain-twister. Again, it is necessary to supply additional information. Here we merely state that the two angles at C and F are equal to each other. Find segment AC, given AB, DB, and CF.

One of the biggest problems that Apple *II* programmers face is the inability to place standard characters on the hi-resolution screen. But, a number of alternatives are available to the programmer.

One method would be to create a machine-language routine (see "Apple Seedlings" in this issue) which assists in graphics development and manipulation. However, you must know machine language, or have access to a program which can manipulate graphics for you. All of this can be a worthwile effort if the task at hand is very complicated. For small or limited jobs though, it may be easier to rely on the DRAW and XDRAW commands in conjunction with shape tables.

In this program we needed to recreate the letters A through H for display on the high-resolution screen. These characters are used to label the line segments of the geometric shapes. Lines 2110 and 2120 contain the shape table data. The first two values indicate how many shapes are in the table. Following this will be two bytes for each shape—these indicate where in the shape table the shape data starts. By knowing this, you can easily find the shape data for each of the eight shapes.

Two commands can access these shapes and place them on the high-resolution screen for you. The DRAW command will place the indicated shape at the location you specify, replacing any graphics which may already be on the screen under the shape. The XDRAW command is most commonly used for animation because of its special ability to reverse the color of the screen at every point where the shape is drawn. This simplifies animation because you can draw at the same location twice, reversing the screen colors where the shape draws again, and reverting them back to their original state before the first shape was drawn.

A number of short routines from line 1710 through 1960 draw the geometric shapes and display the letters which mark the line segments.

The Commodore 64's BASIC language can be difficult to work with, though in many ways it is very forgiving. The INPUT statement is one example. In a "clean" program, it should be so difficult for average users to make a fatal mistake (one that crashes the program or causes an error) that they couldn't make one if they tried. This is perhaps the most difficult aspect of programming—making sure that all "end-cases" are checked.

End cases are the limits within which the program must operate to perform properly. It could be as simple as making sure a user can't crash the program by pressing a key that was never meant to be pressed, or doing a complex edit check on a user's typed input in search of a legal response.

One end-case check which is often overlooked in BASIC programs is in the entry of numeric information. If you use a numeric variable with the INPUT statement, inputting anything other than a legal number will bring the cryptic message "REDO FROM START." How does one avoid this?

Instead of using a numeric variable with the INPUT statement, use a string variable. When string variables are used, virtually anything can be entered. If you need to have a numeric value from the input, then you can convert the string characters into a numeric value. The VAL function does this for you, and it is much more flexible with numeric variables than the INPUT statement. The VAL function will scan the list of characters in a string and convert as many as it can into a number until it runs into an illegal character. This means that if you accidentally typed 123R, the VAL function would return the value 123. Any other numeric characters after the R would also be ignored—e.g., 123R456 would return a value of 123.

# HCM ONE-LINERS

Here they are . . . the best of the one-line programs that we have received since printing the second "*HCM* One-Liners" column in *Home Computer Magazine* Vol. 5, No. 2. Although many interesting programs were submitted, we have selected what we felt were the best four (one for each brand of computer covered in our magazine) of those that arrived prior to this issue's press date. If you have not yet submitted your masterpiece, it is not too late! As long as we keep getting great One-Liners written in any computer language, we'll keep filling this page for you. Our prize winners this issue will each receive a check for $50 for sharing their ideas with our readers.

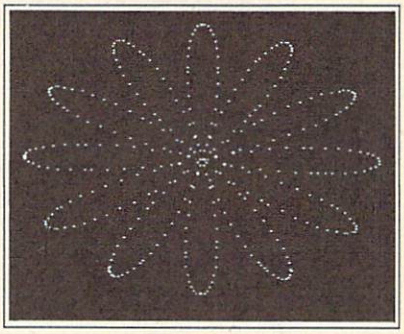
## Polar Coordinates
**[Applesoft BASIC**
**on the Apple *II*e, *II*c]**

Dear Sir:

After reading your magazine, I decided to enter a one-liner of my own. This one-liner draws 10 different polar coordinate curves. They are referred to in some calculus books as leaved-roses. Each odd-numbered pattern shows that same number of leaves; each even-numbered pattern shows double that even number of leaves. It takes about 8 minutes to finish all 10 curves.

James R. Klenke
Whitewater, KS 67154

```
1 TEXT : HOME : FOR B =
 1 TO 10: HGR2 : HCOLOR
= 3:C = 0:D = 360 + 360
 * ( INT (B / 2) = B / 
2 AND B > 5): FOR A = C
 TO D:Z = COS (A * B):
X = 139 * COS (A) * Z 
+ 139:Y = 95 - 95 * SI
N (A) * Z: HPLOT X,Y: N
EXT A,B: TEXT : HOME
```



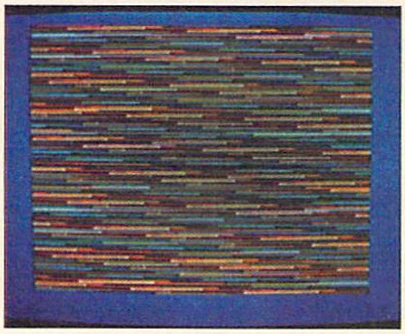
## "It's Alive!"
**[Commodore 64 BASIC**
**on the C-64]**

Hi There HCM,

Here's a one-liner that will make your screen come alive. By pressing down various keys, you can create a variety of effects. The only way to exit from this frantic routine is to press the (RUN/STOP) and (RESTORE) keys simultaneously.

Thom Randall
Roseburg, OR 97470

```
1 ?"☞SHIFT CLR☜":S=828:
P☞SHIFT O☜S,238:P☞SHIFT
 O☜S+1,33:POKES+2,208:P
OKES+3,76:POKES+4,60:PO
KES+5,3:SYSS
```



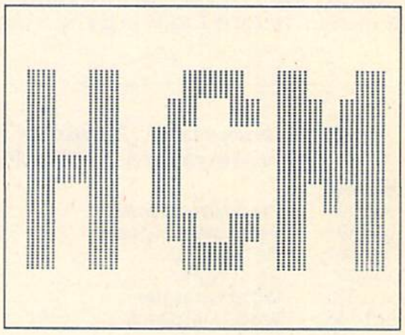
## Message Banner
**[BASICA on the IBM PC,**
**Cartridge BASIC**
**on the IBM PCjr]**

Dear Editors,

Included is my one-liner contest submission for the IBM PCjr (it also RUNs on the PC). As you can see, the program prints a banner on a line printer. Thanks for the great coverage of the PCjr! [NOTE: You can input up to 255 characters for your custom banner.]

David J. Bohlke
Manchester, IA 52057

```
1 SCREEN 1:COLOR 1:CLS:
INPUT"<ENTER> BANNER ",
M$:FOR A=1 TO LEN(M$):C
LS:PRINT MID$(M$,A,1):F
OR C=0 TO 7:FOR I=1 TO 
3:FOR R=7 TO 0 STEP -1:
A$=STRING$(9,POINT(C,R)
+32):LPRINT A$;:NEXT:LP
RINT:NEXT:NEXT:NEXT
```



## Tabling Multiplication
**[TI Extended BASIC**
**on the TI-99/4A]**

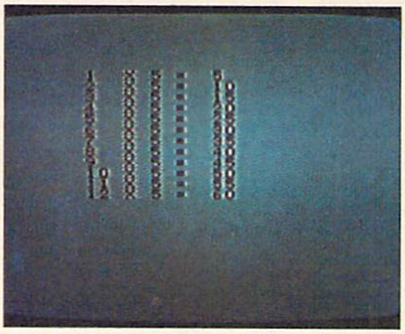
Dear Editor,

I read with interest your call for one-liners in *HCM* Vol. 4, No. 5. The ones in that issue were most interesting, and made me think.

This is my entry for your contest: When you RUN this program, input a number at the blinking cursor. A table with this number and multipliers 1-12 will be displayed with the answers in columnar format.

Raleigh H. McQueen
Savannah, GA 31419

```
1 ACCEPT AT(1,1):B :: F
OR A=1 TO 12 :: DISPLAY
 AT(A+3,5):A;TAB(9);"X"
;TAB(10);B;TAB(12+LEN(S
TR$(B)));"=";TAB(14+LEN
(STR$(B))));A*B :: NEXT
A :: GOTO 1
```



All One-Liner submissions are subject to the same publishing criteria as Letters to the Editor (explained in the magazine's masthead, page 6). If you have written a great One-Liner in any language on any computer covered by *HCM*, send it addressed to: Letters to the Editor, 1500 Valley River Drive, Ste. 250, Eugene, OR 97401. You too may win a cash prize and be immortilized in print!

Have you ever noticed the little arrow symbol on the (ENTER) key of your PC or PCjr keyboard? You may also have noticed that quite a few programs use this symbol to represent that key rather than spelling out the word "RETURN" or "ENTER." Two built-in graphics characters resident on the system allow you to build this shape for use within text: ASCII characters 17 and 217. In a program which uses this symbol many times, it is sometimes useful to place these two characters into a string variable with a name which is easily understood:

```
RETRN$ = CHR$(17) + CHR$(217)
```

This works very well as long as you're in Text mode or if you have a PCjr. But if you are using a PC in one of the graphics modes, this string may not work properly, depending on the color graphics adapter card that you are using. (Some graphics adapter cards do not support the built-in graphics character set; the PCjr does support it). If your program has that problem, you still can draw in the arrow shape as we did in *Geometrix*. It requires a little more effort and planning than the method just described, but the results are just as satisfying.

First, set up a string variable which contains the DRAWing commands:

```
ENTDR$ = "E3DG2DE2DGDFU2ER6U3LD2LU2"
```

Before you can DRAW this shape, you need to determine where on the screen it should appear. There are a number of ways this could be done. One method involves adding another command to the beginning of the DRAW command string mentioned earlier. The BM command causes the pointer for the DRAW command to move to the coordinates specified without drawing a line. Another method (the one used in this program) is to simply use the PSET command to position the DRAWing cursor before DRAWing the shape.

One problem programmers still face in displaying graphics on the TI-99/4A is the lack of an expedient way to get graphics characters on the screen in large quantities. Innumerable algorithms have been designed to help this process—some quite complex, and some quite simple. In the case of this program , the simplest way is not only the most efficient, but also the quickest.

The one requirement for this method's success is that all of the graphics characters must be confined to those characters accessible from the keyboard; you will need to be able to type their unaltered counterpart into data statements. A FOR-NEXT loop can then be used to READ the data, one line of graphics at a time, and print it to the screen using the PRINT statement. If you are using Extended BASIC, you could speed up this process even more by using the DISPLAY AT statement.

To simplify the design of such a screen full of graphics, a piece of graph paper may be necessary. Then you can map out the position of each character on paper, so that when you enter the characters into the DATA statements, you will be able to count spaces and repetitions of characters (to ascertain their proper alignment with the rest of the screen). This may reduce your screen-development time by 50 percent or more.

The DATA statements which contain the screen format are in lines 4390 through 4470. In this program, we have left the upper-case (capital) letters alone so that they may be used as the labels for the line segments of the geometric shape. The lower-case letters and other characters have been redefined to create the geometric shapes. In these few lines of DATA are the formats for seven geometric shapes. To designate the end of the graphics for a screen, a null item is used in the data. A null item occurs when two commas are placed one after the other.

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.**

**HCM**

### *Geometrix* (Apple *II* Family)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-200 | Program header. |
| 210-230 | Relocate program. |
| 240-340 | Get options. |
| 350-720 | Rectangles. |
| 730-1030 | Right triangles. |
| 1040-1360 | Parallelograms. |
| 1370-1700 | Trapezoids. |
| 1710-1960 | Draw shapes |
| 1970-1990 | End-of-problem routine. |
| 2000-2100 | Key-scan and input routines. |
| 2110-2120 | Graphics data. |
| 2130 | Text-messages data. |

### *Geometrix* (C-64)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-200 | Program header. |
| 210-360 | Get options. |
| 370-760 | Rectangles. |
| 770-1130 | Right triangles. |
| 1140-1510 | Parallelograms. |
| 1520-1940 | Trapezoids. |
| 1950-2080 | Check and get the answer. |
| 2090-2150 | Next problem menu. |
| 2160-2640 | Draw shapes. |
| 2650-2660 | Time-delay loop. |
| 2670-2690 | Continue routine. |

### *Geometrix* (IBM PC and PCjr)
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-230 | Program header. |
| 240-320 | Get options. |
| 330-480 | Rectangles. |
| 490-620 | Right triangles. |
| 630-800 | Parallelograms. |
| 810-980 | Trapezoids. |
| 990-1000 | Input routines for answers. |
| 1010-1020 | Check answer. |
| 1030-1040 | Next problem menu. |
| 1050-1130 | Draw shapes. |
| 1140 | Key-input routine. |

### *Geometrix* (TI-99/4A)
### Explanation of the Program

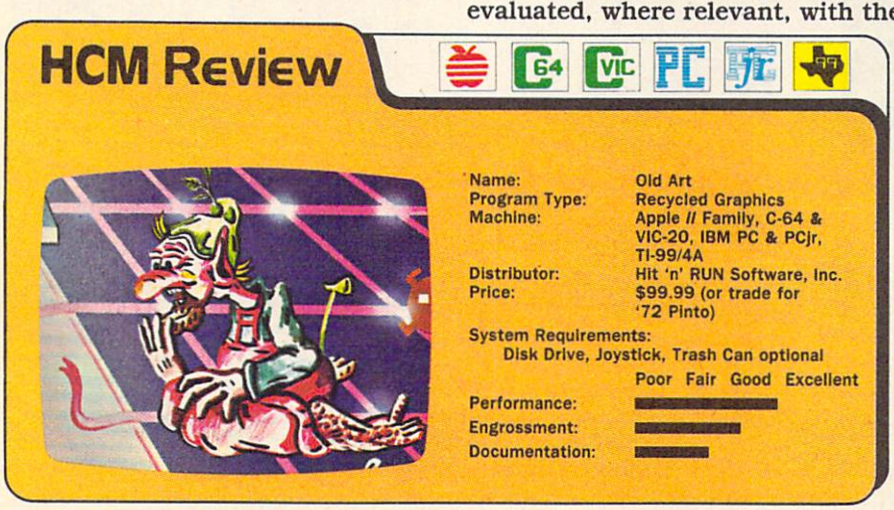
| Line Nos. | |
|---|---|
| 100-200 | Program header. |
| 210-290 | Initialize the program. |
| 300-460 | Get options. |
| 470-1340 | Rectangles. |
| 1350-2140 | Right triangles. |
| 2150-2860 | Parallelograms. |
| 2870-3710 | Trapezoids. |
| 3720-3970 | Misc. input routines. |
| 3980-4020 | Next problem menu. |
| 4030-4060 | Time delay and beep routine. |
| 4070-4100 | Single-key input routine. |
| 4110-4470 | Control and do shapes. |

# HCM Review Criteria

Each month, *Home Computer Magazine* (*HCM*) reviews products designed for the Apple *II* Family, Commodore 64 and VIC-20, IBM PC and PCjr, and Texas Instruments 99/4A computers. *HCM* reviews take a detailed look at the quality, utility, and value of commercially available packages for these machines. Because our publishing charter forbids accepting outside advertising, we strive to make the scope and content of our review pages shine with a unique blend of humanistic frankness and objectivity.

Not only will you find all relevant information for making a wise purchase decision, but in some special cases we also provide nuggets of compu-prestidigitation.* For example, we frequently include essential documentation not furnished by the manufacturer. Additionally, each issue of *HCM* tries to review at least one outstanding product—a "Diamond in the Rough"—which, because of company size, marketing clout, or for some other reason, has not received the attention it deserves.

**At the beginning of each review, a review-at-a-glance box provides the user with an instant assessment of the product. Each item will be evaluated, where relevant, with the criteria below.**



**HCM Review**

Name: Old Art
Program Type: Recycled Graphics
Machine: Apple *II* Family, C-64 & VIC-20, IBM PC & PCjr, TI-99/4A
Distributor: Hit 'n' RUN Software, Inc.
Price: $99.99 (or trade for '72 Pinto)
System Requirements:
Disk Drive, Joystick, Trash Can optional

| | Poor | Fair | Good | Excellent |
|---|---|---|---|---|
| Performance: | | | | |
| Engrossment: | | | | |
| Documentation: | | | | |

*** Performance—**
How well the product performs as intended; how well it takes advantage of a specific machine's capabilities; how well it responds to the user's commands; how effectively the graphics, sound effects, music, or speech are integrated with the software.

*** Engrossment—**
Whether the game or activity has that intangible quality that holds players on the edge of their seats while the hours tick by unnoticed.

**OR**

*** Ease of Use—**
The degree to which a user can interact with the product without outside help: the ease and effectiveness of error-handling features; whether the actual reading level of the activity is appropriate for the suggested audience.

**OR**

*** Ease of Set-up—**
How well the product design facilitates easy installation.

*** Documentation—**
The quality of the printed matter that comes with the product; whether the instructions are clear and comprehensive; whether the machine configuration requirements are spelled out. Information such as how to load a program, use the keyboard, and restart an activity contributes to the documentation rating, as do tips on performance peculiarities.

**Products may also be evaluated in the following areas:**

*** Flexibility—**
Can the product be adapted to the specific needs of the users?

*** Cost/Benefit—**
Is the product worth the user's investment in time and money?

*** Necessity—**
Is the product a solution for which a problem already exists?

*** Originality—**
Is it unique in concept, or simply a "me too" product?

*** Longevity—**
The "Boredom Factor." Does the program sustain interest?

*** Rewards—**
Are the audio-visual rewards motivating and appropriate?

*** Concept Presentation—**
Are the concepts presented clearly, logically, and in depth?

*** Special Effects—**
How does quality of sound and visual effects rate? Do they enhance or detract from the product or learning process?

## Attention Software Authors & Peripheral Inventors:

### * WANT TO BE DISCOVERED? *

### Home Computer Magazine Wants To Give You A Chance!

We are looking for home computer products that have not received the attention they deserve. Each month, we will be singling out one such package for special review. If you have a unique commercial product of exceptional quality—but your advertising and promotion budget has not allowed you to capture major media attention—we want to see it. We will consider reviewing any product that meets our high standards.

We are an Equal Opportunity Reviewer!

In order to qualify for possible review, your product must:

1. Currently be available for purchase to readers of this magazine.
2. Make a unique and important contribution to the home computer industry.
3. Be of outstanding merit, quality, and value.
4. Be consistent with the type of machines and products we normally cover.

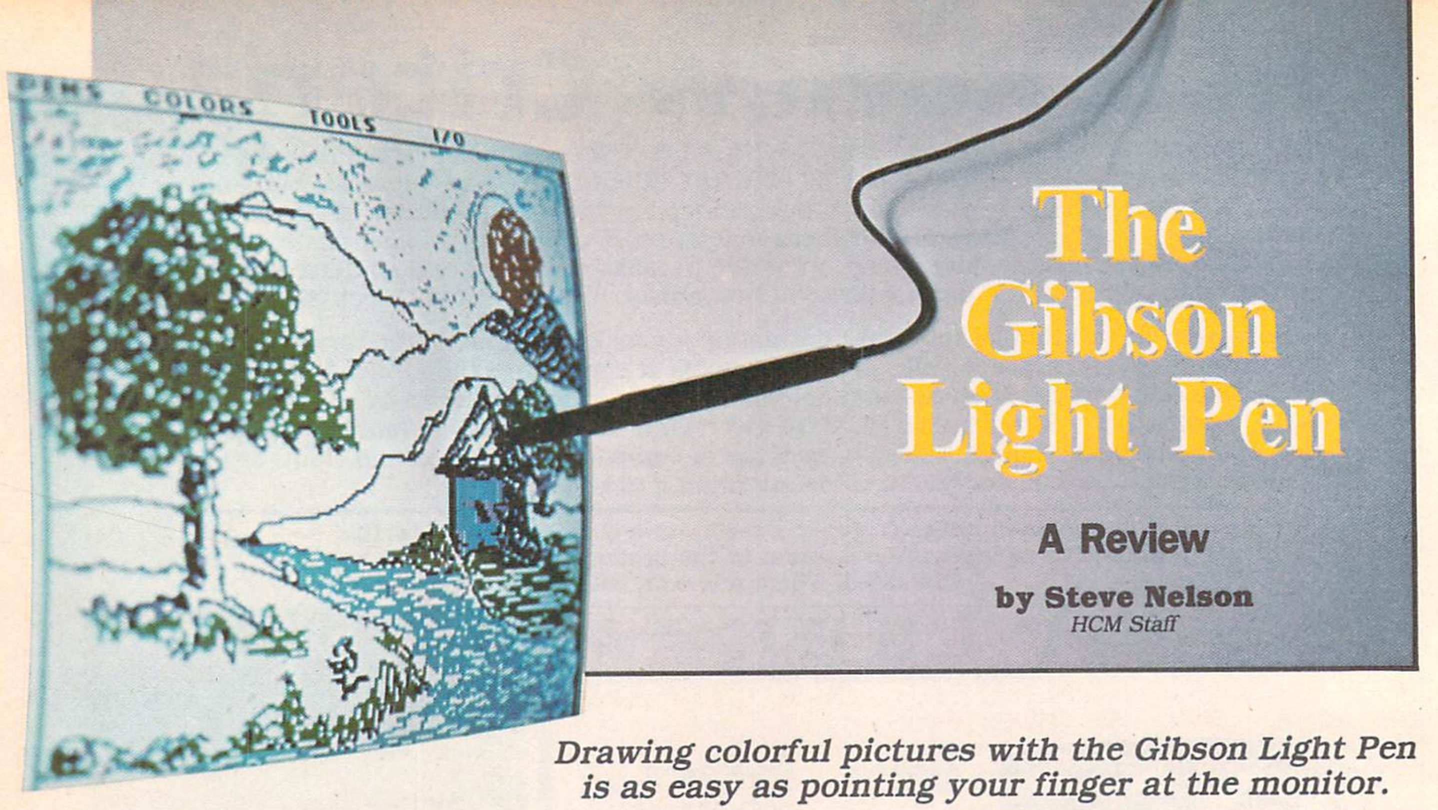
If you feel that your product qualifies, mail it to:

Home Computer Magazine
Attn: Editorial Submissions
1500 Valley River Drive, Suite 250
Eugene, OR. 97401

We reserve the right *not* to reply to each inquiry, so please do *not* contact us except to request return of your product. If you want your product to be returned, please include sufficient return postage.

*Compu-prestidigitation (kóm′• pū • prĕs′• teh • dĭ • jeh • tā′• shŭn) —*n* 1. The magical quality of unexpected comprehension that results from presenting technical information about computers in a lively, entertaining, visually attractive and easy-to-understand format. 2. The magical tricks that make a computer sing, dance, and do all sorts of wonderfully useful things.

# The Gibson Light Pen



## A Review

by **Steve Nelson**
*HCM Staff*

***Drawing colorful pictures with the Gibson Light Pen is as easy as pointing your finger at the monitor.***

Take one Apple *II*e or II+ computer, add a color TV or monitor, some specks of light, and voilà—instant art! The *Gibson Light Pen* is one of the easiest ways for a person to utilize the graphics potential of an Apple computer. I mean, what could be simpler? Just point the pen at the screen and watch your drawing develop.

The *Gibson Light Pen* comes with two disks: Penpainter, and Pendesigner. The Penpainter disk has four programs on it: *Penpainter*, *Penanimator*, *Pentrak*, and *Penmusician*. Setting up is very easy—just plug the card into slot number 7 and run the wire out the front of the computer over the keyboard. If slot number 7 is already occupied, you can insert the card in any other slot as long as you attach it (with the short jumper wire included in the package) to the appropriate integrated-circuit pin.

### Painting

Once this is done, boot the disk and view the *Penpainter* introduction screen. Calibrating the pen to the monitor—recognizing the position of the pen on the screen—is a simple process: point the pen at a small horizontal and vertical lines on the screen and the program automatically calibrates the pen for you. The calibrating process uses the video synchronization signal and the known position of the calibrating lines in the hi-res memory to give the pen's software a reference point. To subsequently determine where the pen is pointing, the software uses this reference point plus two other pieces of information: (1) the timing of the sync signal, and (2) the time at which the sensor is activated by an illuminating pixel.

*"The Penpainter program uses pull-down menus similar to those on the Macintosh to access the options."*

You are next presented with a blank screen with an option bar across the top. The *Penpainter* program uses pull-down menus similar to those on the Macintosh to access these options. You have your choice of Pens; Colors; Tools; and the I/O option, which lets you save pictures to disk, retrieve them, and access the printer.

Under the Pens option, you have your choice of 14 ways to draw and create on screen. *Penpainter* follows the standard format of most graphics programs—you can draw freehand; create squares, circles, lines, etc.; mirror; and use fill colors and patterns. *Penpainter* lets you choose from 6 colors, including black and white. The Patterns option offers 75 preset patterns, and the option to change them or even create your own pattern. While in the *Penpainter* program, you can use any of these colors to either draw with or add color to specific areas of your picture. One annoyance, though, is not being able to edit your picture with the *Penpainter* program. You must save it to disk, boot up the *Pendesigner* program, and then call up your picture before you can access the editing functions.

### Animating

*Penanimator* gives you the ability to produce an animation sequence using up to 20 frames. Each frame can be accessed individually, using the previous frame as a template for making your changes. The program works in a Zoom mode (similar to the Fat-Bits option on the Macintosh's *Macpaint* program). The blank screen has a grid pattern of minus signs on it, and you produce your drawing by aiming the light pen at the minus that you want to turn on. This method of fat-bit drawing is an excellent way to make controlled changes. The actual animation works similar to a motion picture, cycling the different screens at approximately 10 frames per second, producing a very smooth animation effect. Penanimator is easily one of the best products of its kind that I have ever seen, and is my favorite part of this light-pen package.

### Designing

If you are an experienced Applesoft programmer, you may want to add light-pen graphics to your programs. *Pentrak* is a machine-language program for just this purpose. Included in the manual is a *Pentrak* reference guide with 27 commands and their definitions.

|  |  |
|---|---|
| Name: | Gibson Light Pen |
| Program Type: | Graphics tool |
| Machines: | Apple *IIe*, *II*+ |
| Distributor: | Koala Technologies Corp.<br>3100 Patrick Henry Dr.<br>Santa Clara, CA 95050 |
| Price: | $249 |
| System Requirements: | 64K, color monitor |

|  | Poor | Fair | Good | Excellent |
|---|---|---|---|---|
| Performance: | ■ | ■ | ■ |  |
| Ease of Use: | ■ | ■ | ■ |  |
| Documentation: | ■ | ■ | ■ |  |
| Cost benefit: | ■ | ■ |  |  |

The second disk, *Pendesigner*, is the heart of the *Gibson Light Pen* system. With this program, you have access to the powerful editing options which let you fashion your picture into the masterpiece you always knew it would be. Unfortunately, the *Pendesigner* program does not draw in color, so when you have finished drawing your picture, you must save it to disk, boot up *Penpainter*, and call up your picture before you can add color to it.

*Pendesigner* is primarily intended for precision drawing like drafting, designing business charts, layouts, etc. With it, you can draw complex designs, use the editing options to correct errors, and finalize your picture. The *Pendesigner* program does not use pull-down menus like *Penpainter*—rather, it has a menu screen that is superimposed over the drawing screen. You can hide it—clearing the drawing surface—or bring it back by pressing the space bar. From this menu screen, you can choose the Drawing options (Freehand, Line, and Box) or the Tools options (Edit, Grid, Cursor/Crosshairs, Frame, Black or White, Negate, and Card Images) and use them to either create a new picture, or edit an old one from the *Penpainter* program.

### Cleaning Up Your Creation

*Pendesigner*'s edit function allows you to change portions of your drawing pixel-by-pixel by using the Zoom mode. Just point with the light pen to the portion of the drawing that you want to edit, and press Z for a close-up view of your edit point. The grid option helps you draw perfect horizontal and vertical lines on the screen by forcing all lines and box corners to stay on the grid points. Cursor/Crosshairs helps you to see exactly where the drawing point is on the screen by giving you a choice of cursor styles. One style is a white cursor which is superimposed over the point where you are drawing. I found this to be too large for my taste. The other type of cursor is Crosshairs, which shows exactly where your pen is pointing.

> *"The Patterns option offers 75 preset patterns, and the option to change them or even create your own pattern."*

The Frame option is a very useful tool, allowing you to "frame" a portion of your drawing and copy it, move it, erase it, even shrink or stretch it. By using the Black or White and Negate options, you can reverse the screen—make black white, and white black.

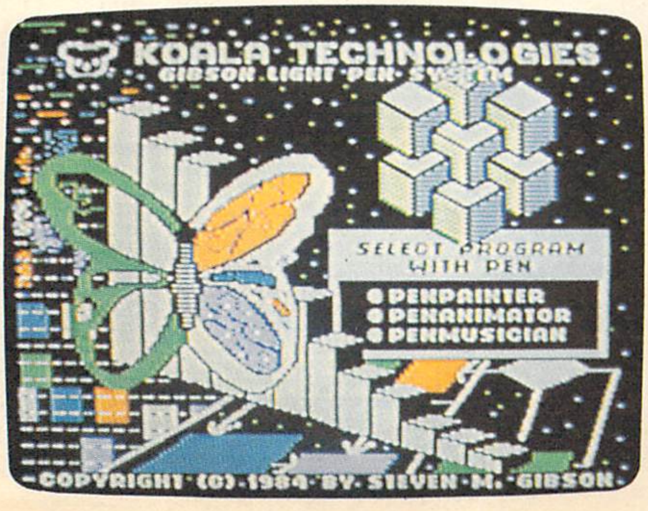
The Card Images option lets you store stock images that you may wish to use over and over. The program stores them in a sort of electronic index card file, which you can access to add them to your drawing page at any time. With this many powerful and useful editing functions, you can clean up even the messiest of drawings.

### A Little Bit Awkward

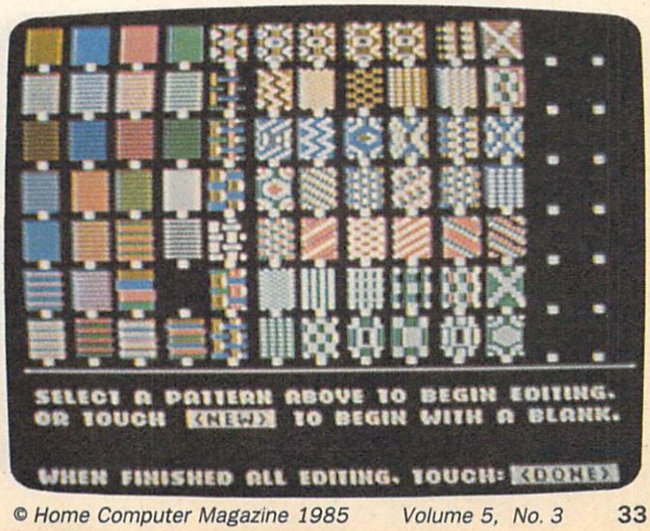
Drawing with the *Gibson Light Pen* is both easy and difficult. It's easy because you just point the pen and the line tracks where you are pointing. It is difficult because it's awkward to draw in a vertical position and it takes some time to get used to turning the pen off before the line goes beyond where you want it. Once you become familiar with the way the pen works, it is easy to control. I found that it worked best when I didn't touch the pen to the screen, but used it like a pointer, or an extension of my finger.

The response of the cursor to the pen is phenomenal—you almost can't outrun it. One of the functions under the Tools option is Inertia. Using this, you can adjust the response of the drawing point to the light pen to suit your style. One problem I noticed, however, is that when you set Inertia for the fastest response, the drawing point is difficult to control when attempting to draw slowly or precisely.

*A colorful title screen introduces the Light Pen.*



*This menu provides a wide choice of fill patterns.*



SELECT A PATTERN ABOVE TO BEGIN EDITING.
OR TOUCH <NEW> TO BEGIN WITH A BLANK.

WHEN FINISHED ALL EDITING. TOUCH: <DONE>

This brings up an old familiar problem with computer graphics. Due to the way computers display graphics, you are going to have what is called "blocking"—where the line you draw is zig-zagged, and color spills over into the surrounding colors. This happens because the computer sees the TV or monitor screen as a grid of points (called pixels). Each pixel can be turned on or off and assigned a specific color by the program. The *Gibson Light Pen* does the best it can under the circumstances, but since the Apple *II*'s graphics system is limited to a resolution of only 280 by 192 without special software, it is nearly impossible to avoid these problems (the *Gibson Light Pen* does not access the double high-resolution mode). All this means is that after drawing a picture with *Penpainter*, you will undoubtedly have to use *Pendesigner*'s Zoom feature to clean up the rough spots.

### Making Music

*Penmusician* gives you the ability to sample some of the sound effects that the Apple II computers are capable of producing. You can compose on three full staffs of music (24 notes) at a time. Each note is preset at B and is a whole note. You can change them by pointing with the light pen to the positon on the scale where you want them to be, and pressing the space bar. For instance, if you want to change a note to a C, simply point the light pen to position the note in the appropriate spot of the scale and press the space bar. You can also place a sharp or a flat using the option bar at the top of the screen. After you have placed the notes where you want them and added sharps, flats, natural designations, and rests, you can direct the program to play your composition.

> *"The Frame option is a very useful tool, allowing you to "frame" a portion of your drawing and copy it, move it, erase it, even shrink or stretch it."*

Although you can create some fairly complex jingles, don't plan on selling your synthesizer yet. This program doesn't really do much. It seems more of an added-on feature, just to make the whole package a little more well-rounded. Moreover, this program does not come with a demonstration mode, or a teaching mode, so—if you have never written music before, *Penmusician* will probably be somewhat useless to you.

### Finding Faults

As easy as the *Gibson Light Pen* is to use, it has more problems in addition to those already mentioned. For instance, when using the Pens option, you can select different shapes to insert in your picture. But the program seems to have occasional difficulties getting out of the option you select. This happens in all of the different draw options except for Freehand, but is particularly bad when using the Squares and Circles options. Another problem has to do with placement of the pen. Once you complete a square, and you want to create another one at a different place on the screen, the program will sometimes recall the last point where it was and begin the drawing from that point, rather than the new location.

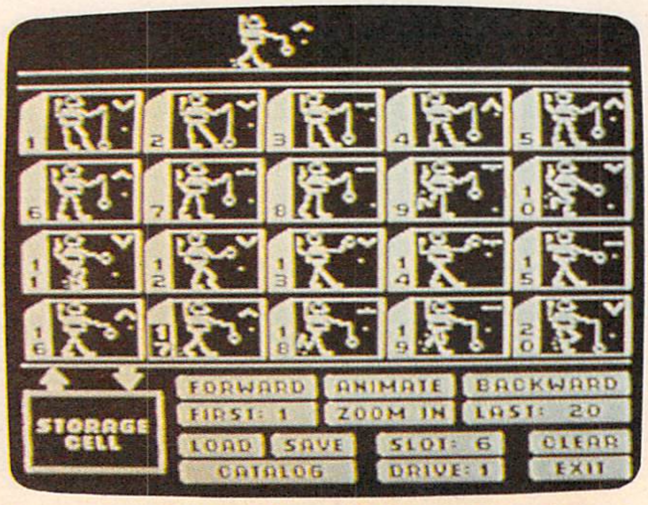
The *Gibson Light Pen* doesn't have a generic fill function that you can use to add color to areas on the screen that are not created by the Shape options. You can use the Patterns option to do this, but it is tricky—you must select a pattern, and then point to the area of the screen that you wish to fill. If you are not paying attention, the pattern will begin filling at the place where you were last pointing with the pen.

And, while I'm on the subject of problems, it is unfortunate that you must swap disks each time you want to edit a *Penpainter* drawing, or add color to a *Pendesign* drawing.

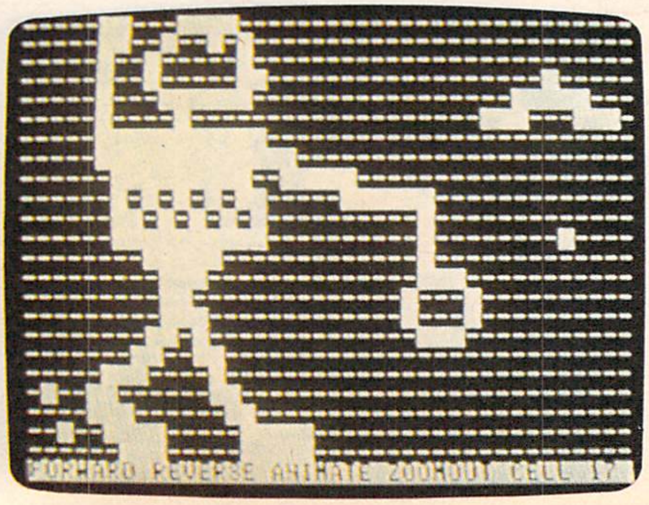
The *Gibson Light Pen* is not cheap, and before spending $249, you should be sure that this product will meet your needs. Several graphics tools on the market will give you a similar ability to draw graphics on the Apple, and are less expensive. [See the review of *Mousepaint* in Vol. 4, No. 4 of *Home Computer Magazine*—Ed.] If you are looking for a product that will aid you in creating computer graphic screens, but are not an artist making your living doing it, then you may want to shop around some. If, on the other hand, you are a serious artist, this product may help you immensely. I know several people who are using the *Gibson Light Pen* professionally and are very pleased with its results.

Aside from these complaints, the *Gibson Light Pen* is a useful and flexible system that is not only practical, but fun as well. The documentation provided is good, installation is quick and easy, and the product's performance is very good compared to other similarly priced graphics systems available for the Apple *II* Family of computers. HCM

*Animation is coordinated on an editing screen.*



*Once the sequence is built, the full-size animation can begin.*

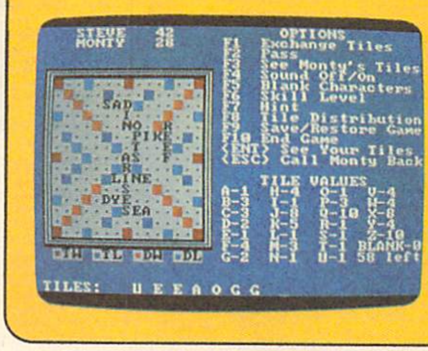


# Monty Plays Scrabble

**A Review by Steve Nelson**
*HCM Staff*

***Stick to the board if you don't want to be bored with this video rendition of an old classic.***

## HCM Review



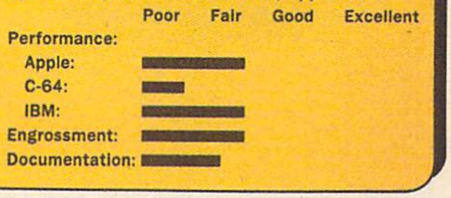
```
STEVE    42          OPTIONS
MONTY    28     F1  Exchange Tiles
                F2  Pass
                F3  See Monty's Tiles
                F4  Sound Off/On
                F5  Blank Characters
                F6  Skill Level
                F7  Hint
                F8  Tile Distribution
                F9  Save/Restore Game
                F10 End Game
                <ENT> See Your Tiles
                <ESC> Call Monty Back

                     TILE VALUES
                A-1  H-4  O-1  V-4
                B-3  I-1  P-3  W-4
                C-3  J-8  Q-10 X-8
                D-2  K-5  R-1  Y-4
                E-1  L-1  S-1  Z-10
                F-4  M-3  T-1  BLANK-0
TW  TL  DW  DL  G-2  N-1  U-1  58 left

TILES:   U E E A O G G
```

Name: Monty Plays Scrabble
Program Type: Game
Machines: Apple *II* Family, C-64, IBM PC, PCjr
Distributor:

IBM & Apple versions:
Ritam Corporation
P.O. Box 921
Fairfield, IA 52556

C-64 version:
EPYX Inc.
1043 Kiel Court
Sunnyvale, CA 94086

Price: $39.95
System Requirements: IBM: 64K, Apple: 48K



Computers have been used to simulate all sorts of things over the years, from airplanes, to aliens—even nuclear reactors. [See *Over-Reaction* elsewhere in this issue—Ed.] For gaming purposes, computers have often been used to simulate an opponent. *Monty Plays Scrabble* is one such game where you pit your skill against the binary brain. Unfortunately, this particular binary brain from Ritam and Epyx is somewhat simple-minded.

When you compete with Monty, you are competing with a large stored vocabulary (from the official *Scrabble* player's dictionary), and the computer's speed to scan through the thousands of possible word combinations. Most of the monitor screen is taken up by the board, and what's left displays an options list, as well as prompts for entering tiles. When the game begins, you have the option of letting Monty go first, or going first yourself. It would be better if the computer flipped a coin or randomly rolled some dice to decide this, instead of making you decide—some people think that the player going first is at a disadvantage.

### Is It A Clone?

*Monty Plays Scrabble* follows the regular *Scrabble* game format pretty closely. You draw 7 tiles, which you use to make words. When it is your turn, you have an unlimited amount of time to enter a word, as does the computer. On the more difficult levels, however, the computer sometimes will take forever to decide. None of the versions let you set a time limit, and I really think one is needed. IBM and Apple versions do allow you to interrupt Monty while he's thinking—making him use his best play "so far."

If you are unable to create a word, you can exchange tiles; however, you do lose a turn if you select this option. You may also opt to pass your turn, look at Monty's tiles, watch Monty think as he is scanning through the letters, or even ask Monty for a hint when you are stumped. You may also save your game to disk, load an old game, and end the game at any time.

### Two Out of Three Are Close

I tested three machine versions of this game, and two of them—the IBM and Apple versions—are virtually identical. The C-64 version, however, is quite different—disappointingly so.

The IBM and Apple programs let Monty challenge any word you make if he doesn't recognize it. You have the option to do the same, and because Monty sometimes has a few problems, you will need it. For example: If you have the word FARM going down, and the word RAT going across, with R as the common letter, Monty may "blow it" when making another word. Let's say Monty wants to make the word AND, using the A in RAT as the common letter. AND is a legal word, but MN is not, so you must challenge the word. You can do this on all versions except for the C-64. This maddening inability to challenge in the C-64 game is a major drawback that essentially ruins the game. I can't believe that this serious a flaw could pass unnoticed. Not only does the program accept any gibberish that the computer comes up with, it lets you input anything as well.

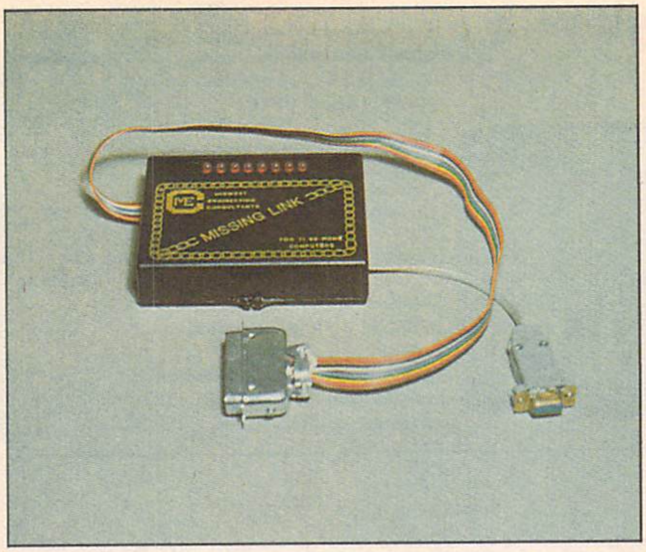
### Check That Word!

Challenging a word on the IBM or Apple versions is a simple process. After you or Monty enter a word and place it on the screen, Monty will display the word's score, and ask you to input Y or N. If you input N, Monty will ask you whether the word is legal. You can then look up the word in a dictionary. Because Monty's vocabulary is finite, he always accepts your decision. This is another real problem. If you are free to keep Monty from using words that *are* legal, and you can *make* Monty accept an illegal word, then you can't really lose if you don't want to . . . The exciting element of bluffing—with its associated loss-of-turn penalties for getting caught or mounting a losing challenge—is therefore missing from this software implementation. And one might rightly argue that it's not really *Scrabble* without it.

All versions have various skill levels, a 3-player option, and the option to let Monty play against himself. And although the documentation is adequate if you've played *Scrabble* before, a set of the official rules should be included for beginners.

I found *Monty Plays Scrabble* to be alternately boring and extremely frustrating because of its previously mentioned limitations. Frankly, I wouldn't feel right about giving these games a positive recommendation. Both the IBM and Apple versions are "usable," but the C-64 version is definitely incomplete without the challenge option; thus, it's really not worth buying. In all fairness, it is difficult to say whether the potential for a really good computer implemetation of this game exists, considering the limitations of the most popular hardware it's been designed to run on. With faster processors, more memory, and larger storage devices, a much better "opponent"—and thus, true-to-life game—would be easier to craft. So, unless this program's software designers can find a way to put more "intelligence" into their present product, potential buyers might be best advised to stay with the original "old-fashioned," cardboard-and-wood version.

HCM



# THE MISSING LINK

## for the TI-99/4A

**A Review by Patricia Swift**

*Parallel printers abound—but if you want to connect one to your TI-99/4A, you may be looking at some big bucks, unless . . .*

In the chain of life, there are many missing links—missing, in that they are not yet found. Computer users encounter these all the time: "If only there were a thing to go from here to over there!" For TI-99/4A users in particular, the missing link has been one that will *inexpensively* connect their computer to a parallel printer.

If your printer has a parallel interface, you could spend a lot of time and money trying to attach it to your TI-99/4A. You could add Texas Instrument's Peripheral Expansion Box and RS-232 interface card (which has a parallel port), and then buy a cable somewhere. But *Missing Link* offers a low-cost alternative.

To use *Missing Link*, you'll need either one of two configurations: (1) a Mini Memory cartridge and a cassette recorder with cable, or (2) an Extended BASIC cartridge, 32K expansion memory, and a disk drive.

### A Good Connection

The *Missing Link* hardware consists of a small plastic box with cables attached. The cables connect to the joystick port on the left of the 99/4A console and to the parallel connector on the printer. *Missing Link* works by sending characters out the joystick port, through the device to the printer. Actually, the "Missing Link Algorithms" send the characters. These algorithms are two assembly-language subroutines, which you can call from your own programs. The MSLINK subroutine sends a string of characters out the joystick port. For example, if you wanted to print the string A$ from a BASIC program, you would say `CALL LINK("MSLINK",A$)`. The VPLINK subroutine is a screen dump program which is even easier to call from BASIC by using: `CALL LINK("VPLINK")`.

The software for the Mini Memory version comes on cassette tape. One side has the "Missing Link Algorithms," which are loaded into Mini Memory so that console BASIC programs can use them. The other side has a word-processing program written in console BASIC.

The software for the disk version includes the "Missing Link Algorithms" in assembly language, and a word processor in Extended BASIC. It also includes Extended BASIC programs for building sprites, for printing certain files via *Missing Link*, and for performing electrical engineering calculations. The 32K expansion memory is required. All assembly language routines are stored on and accessed from the disk.

You will not be able to LIST your BASIC programs via the *Missing Link* in the usual way, because BASIC's LIST command doesn't use the joystick port. *Missing Link* offers some utilities for listing programs to the printer, but they are not as convenient to use as LIST. Documentation with the package described a Mini Memory program for LISTing programs which should have been on the tape, but I couldn't find it.

### Setup: Cassette Version

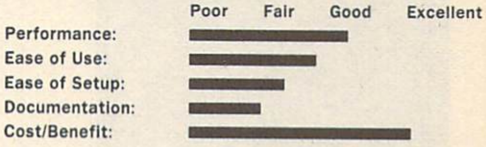
The *Missing Link* package includes some very clear and detailed instructions for setting it up on a system with Mini Memory and cassette tape. I would add one more step: after you hook up *Missing Link*, go into console BASIC and create a short program (all REM's will do). Then `SAVE CS1` to write the program to tape, and answer Y when asked to verify. If you get the DATA OK message, you're set to go. If not, you will have a chance to adjust things until you can record successfully.

After I loaded the "Missing Link Algorithms" into Mini Memory via Easy Bug, I was ready to test the VPLINK algorithm, the one that dumps the screen to the printer. I got an accurate printout of the screen, but all of the characters in the printout had been expanded 200 percent! Undaunted, I continued with the installation instructions by loading the word processing program. I used the word processor to make up a short text and print it out. Happily, my printer was able to print this perfectly—no expanded print problems.

I contacted the program's author about VPLINK, to ask why it was expanding my print while the word processor was acting normally. It happens that ASCII character 31 signals my Okidata printer to use expanded print, and that BASIC uses the same character for its "screen edge" characters at the beginning of its lines. If ASCII 31 means something to your printer, you may find that VPLINK is not useful as shipped.

This brings up an important point about *Missing Link*: you should be prepared to do some fine tuning to make things work perfectly with your printer. If you have a Gemini 10X printer, which is what Midwest Engineering uses, then *Missing Link* will probably work flawlessly right out of the box. But if you have a different printer,

Name: Missing Link
Product Type: Parallel printer interface
Machine: TI-99/4A
Distributor: Midwest Engineering Consultants
P.O. Box 159
Vernon Hills, IL 60061
Price: $30 (cassette) $34 (disk)
System Requirements: Cassette version: Mini Memory, cassette recorder. Disk version: Extended BASIC, 32K memory expansion, Disk drive.



you may want to make some adjustments, and you may need a working knowledge of assembly language to do so.

## Word Processing

The word processing program is the main event in both the cassette and disk versions. When you RUN the word processor, you see an introductory screen which explains the program and then waits for your input. At first I wasn't sure what input was expected, but I soon discovered that the program wanted me to set margins.

The next thing you see is the word processing menu, from which you select options by number. You can add new lines, change existing lines, insert lines, and delete lines. You can replace strings. You can save your text to tape or disk and read it back in later.

Because most printed lines are longer than the 28 columns you get on a BASIC screen, there is no way you'll ever be able to see your text on the screen in its final printed form. The word processor gives you two ways to print through *Missing Link*: either "formatted" or "unformatted." Formatted means that special operators for centering, paragraphs, and continuation are translated to achieve the desired results. In this mode, you can specify that the lines be numbered, or that the un-numbered text be printed single-spaced or double-spaced. You can also print unformatted text.

The cassette version of the word processor has one restriction because it's written in console BASIC: you can't have commas in your text when SAVEing and LOADing files. This is because a comma is a "field separator," which means something special to BASIC. In fact, the only way to put commas in text is to use another character as a placeholder. You could use semicolons as temporary placeholders for commas, and then use a special form of the Search and Replace option to replace all semicolons with commas. (Commas are legal in the disk version because it is written in Extended BASIC.)

> ***". . . you should be prepared to do some fine tuning to make things work perfectly with your printer."***

If your printer has functions that you control by sending it special characters, you can make use of them with *Missing Link*'s word processor. For example, on my Okidata, ASCII 29 means compressed print. *Missing Link* 's documentation (and the *TI BASIC Manual*) shows that the keyboard combination [CRTL] = yields an ASCII 29. Thus, I was able to compress the print by putting a [CTRL] = right into my text. The *Missing Link* documentation lists the proper codes for a Gemini 10X. If you have a different printer, you will need to be familiar with its control codes.

The MSLINK algorithm subtracts 128 from any character with a value over 127 before sending it to the printer. This is not unreasonable, because standard ASCII characters are 7 bit values (127 maximum); but, it means that you will not be able to send control codes over 127 unless you modify the algorithm.

## The Zero Bug

After printing my first document, I did what most people would do: I switched the printer off-line and ejected the page to tear it off. With the printer off-line (but with the power still on), zeroes are fed back to the console. This can fill up the keyboard buffer and produce beeps of protest from the 99/4A. The zeroes (and beeping) stop when you switch the printer back on-line. This happens whenever the *Link* is attached, not just with the word processor. In some situations, you might want to remove the zeroes from the screen before proceeding, but that's the worst-case result. In other words, this problem is annoying but not dangerous. My best solution was to put a page-eject character ([CTRL] L in most cases) at the end of my documents.

## Setup: Disk Version

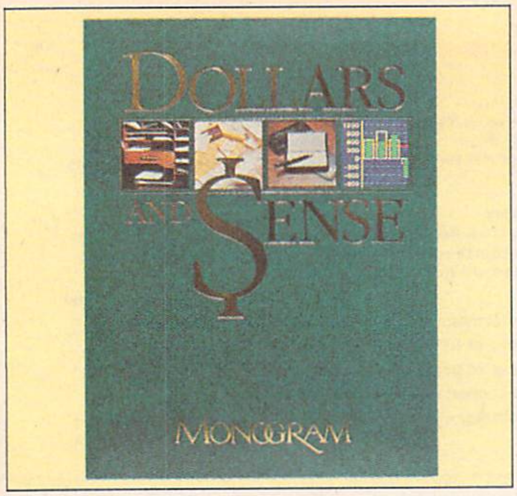
The package includes a prominent "Precautionary Notice" for users who plan to use the *Missing Link* with TI's Peripheral Expansion Box (PEB) hooked up. This notice talks about possible differences in electrical potential between your printer and the PEB. *Missing Link* ties the two devices together electrically, and as such can damage the console in rare cases. To avoid this, connect a wire between any screw on the PEB and a grounded screw on your printer.

No step-by-step documentation for setting up the Extended BASIC version is included. Simply mount the disk before you select Extended BASIC from the 99/4A's master menu, and you will get a list of the programs on the disk. You may select the program you want by number.

## Conclusions

*Missing Link* is a good product which takes some time to learn to use. Its word processor is useful, and the product's low cost makes it very attractive. The package suffers, however, from sparse documentation—instructions that assume the user knows more than is reasonable to expect. (The cassette setup instructions are a notable exception to this statement.) Several documents are included, but they are not well organized. If you have a Gemini 10X or a "plain vanilla" printer, you can probably use *Missing Link* and its word processor as is. If you have a different printer, be prepared to spend some time customizing the software and/or learning to make your printer work correctly with the existing software. This is especially true if you want to make use of your printer's special features.

HCM



# Dollars and Sense

**A review by Dana M. Campbell**
*HCM Staff*

Its name is a cute play on words: *Dollars and Sense*. It has attractive packaging shaded the color of money, and it promises to be easy to use—even if one doesn't know anything about accounting or computers. Best of all, it says it will help me "organize" my money, so that I can figure out where all my dollars are going. I thought "If this friendly looking product can answer a question that's been with me ever since Grandpa gave me my first silver dollar, then by golly that's the package for me!" Since then, however, I've discovered that—after letting $180 of your money get away just to buy this product—you may pay even more in confusion trying to use it.

At first glance, *Dollars and Sense* does appear to be an easy-to-learn "personal financial management program" (as programs of this ilk are labeled). It allows you to set up 120 accounts, or money categories, that you and the program use to: monitor your cash transactions, set up budgets, reconcile your bank statement, make financial reports and graphs, and even write checks. Individual accounts—such as personal checking, auto loan, stocks and bonds, inventory, freight, IRA payments, etc.—can be grouped into larger "Sets" for more specialized use. Three already defined Sets come with the program: Household Accounts, Business Accounts, and Tax Preparation Accounts. It's a good thing they are on the disk, because it's no easy task to figure out how to set up your own from scratch.

> ***"Part of the problem with the tutorial is that it really only provides guided practice on one piece of your financial pie—recording your transactions."***

## Far From Perfect

Of course, the logical thing to do first is to try following the tutorial in the manual that accompanies the program. The Apple *II*e version even supplies a separate demo disk which superficially introduces and explains the menus and processes that occur on the screen. Although the printed tutorial does provide good user-interaction while familiarizing you with transaction entry and editing, the tutorial doesn't go far enough. I wasn't able to extrapolate what had been covered in the tutorial confidently enough to apply it to my own financial situation. Nor could I set up other suggested applications listed elsewhere in the manual. For certain transactions, I was not sure exactly what accounts needed to be defined, and later, accessed, without a lot of trial and error.

Part of the problem with the tutorial is that it really only provides guided practice on one piece of your financial pie—recording your transactions. The Reports and Graphs options are skimmed over, and Reconcile Bank Statement is not even explored here—you must look for it deep in the reference section.

People who are experienced in accounting matters will probably start right in with the program and encounter few problems. Those less experienced in such matters (those most likely to buy a program like this) will necessarily rely on the documentation. But the more I tried to use the documentation, the more trouble I ran into. Key words and explanations are not as well-defined, nor clear and detailed, as they should be. You must constantly flip back and forth through the book to figure out how to set up a Set or work with accounts. More models and examples could have been used to illustrate processes; for instance, showing an entire month of probable financial transactions. Gaps exist in the Apple manual's instructions—prompts and menus do not appear in the same order in the manual that they do on screen. Also, when formatting a new data disk, it is not clear in the manual when the disks should be switched.

## Program Inconsistencies

Some of these deficiencies could have been avoided with the use of adequate screen prompts, but—you guessed it—most of the time you've got to refer back to the manual; a useful prompt pops up only *some* of the time . . . If you get impatient with constantly flip-flopping through the manual, as I did when trying to advance to the next page of a report, you may just start punching keys until you get the desired response.

A quick-reference card listing each key's function would also aid this program, but I can understand why one wasn't assembled: it would be chaos graphically. Many times several keys are used to perform one function—it just depends on which mode you're in. For instance, Q, (Alt) Q, E, or (Esc) at various places in the program will all let you back-out of the program and exit. Monogram should take a hint from Apple's use of across-the-board commands which make *AppleWorks* so easy to interact with. [See a review of *AppleWorks* in *HCM* Vol. 5, No. 2—Ed.]

The manual that accompanies the Macintosh version of *Dollars and Sense* (which was produced later than the IBM and Apple versions) shows some attempt to correct these deficiencies. It's a little more thorough in its

*Why does making sense of Dollars and Sense have to be such an exercise in patience and perseverance . . .*

explanations, and it provides more account examples to work with in the tutorial. The Macintosh program also circumvents the confusion and screen-prompt problems that plague the other versions, merely because of its unique environment. The mouse and the pull-down menus across the command bar at the top of the screen eliminate the need to memorize key-command combinations. In addition, those options or commands that are inappropriate at any given moment are simply "grayed-out" and can't be selected.

Finally, the presentation of this program on the Macintosh screen is delightfully crisp, uncluttered, and pleasing to view. The other versions, in particular the PCjr version, try to cram too much superfluous information onto one little screen. Thus, the current information that you *do* need to see is poorly displayed.

### Does It Ever Get Better?

If you can work your way past these flaws far enough to get some accounts set up on your data disk, you'll discover a few of the better qualities of this money manager.

Monthly budgets may be set up for a fiscal year with starting and ending dates determined by you. When either your data disk becomes full or the year ends, you may want to use an option that will extend all of your accounts, their totals, and other information onto a new data disk.

As you define your accounts, you must identify each one as being either an Income, Expense, Asset, Liability, or Check account. This lets you keep separate, running balances on each of these 5 account types, as well as produce instant balance sheets or reports on your net worth or net income.

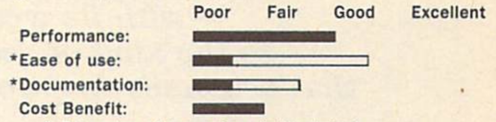
For cash items that pop up irregularly during the year—such as a tax preparer's fee in April or extra expenses for Christmas in December—you can choose the Variable Budget option. For whatever account you're working on when this option is chosen (say, the Expense account TOYS), the program will display the 12 months of the year. There you may enter the extra amount for the appropriate month(s). The total will then be spread out over your monthly budgets for that account.

*". . . probably the best part about Dollars and Sense is the opportunity it offers for speculating on various money strategies."*

When entering transactions, you need to type-in only the first few letters of the account desired in the appropriate field. The program will automatically fill in the rest of the name, or lets you choose from several accounts if there are a few with similar beginnings. In Enter Transactions mode you may also flag items that are taxable or tax-deductible for future reference. Similarly, when reconciling your bank statement you can flag checks that have cleared the bank.

Reducing bill-paying to a minimum of effort is easy with the Automatic Transactions option. Regular transactions like rent and utilities payments only need to be entered once and grouped as a Set. Then reactivate the Set once a month or as often as these accounts must be updated (if updating is even necessary). Every time you need to make those payments, you just have to change the date or check number, for example, and the corresponding accounts are adjusted. The program will handle a maximum of 2500 automatic transactions, in 25 Sets of 100.

| | |
|---|---|
| Name: | Dollars and Sense |
| Program Type: | Financial planning |
| Machines: | Apple *II*, *IIe*, Macintosh, IBM PC & PCjr |
| Distributor: | Monogram<br>8295 South La Cienega Blvd.<br>Inglewood, CA 90301<br>(213) 215-0529 |
| Price: | Apple *II*, *IIe*: $100,<br>*IIc*: $119.95, Macintosh: $149.95,<br>IBM PC/PCjr (with Forecast):<br>$179.95 |
| System Requirements: | Apple: 48K |



* Extension of bar refers to Macintosh

After entering your transactions, you can choose to have your checks printed up automatically. Specially formatted checks can be purchased (from companies listed in the manual) and inserted in your printer. Either one check or a range of checks may be printed out, with or without an address for window envelopes.

### Finally, A Pleasant Payoff

With all of the difficulties encountered in using *Dollars and Sense*, the ease with which reports and graphs can be generated is a pleasant surprise. The Account Year-To-Date Summary reports the current status of each account, and includes yearly totals and the number of transactions saved. The current balance of Asset/Liability accounts and the subtotal of all tax-related transactions for Income/Expense Accounts are also available here. In addition, you may see monthly budget and actual totals for each account, as well as quarterly, annual, year-to-date, monthly, and selected account formats for the Income Statement, Balance Sheet, and Cash Flow reports. All of this, and bar graphs of cumulative budget values and account data, are swiftly available by making a few menu selections.

Aside from helping you to arrange your financial affairs, probably the best part about *Dollars and Sense* is the opportunity it offers for speculating on various money strategies. The example used in the manual examines the possibility of purchasing a boat as a result of a pay raise, and the effect it would have on the Net Annual Budget Balance. Of course, the program's possibilities for forecasting are endless, whether it's making a timely investment for tax purposes or redistributing your income.

Speaking of forecasting, the IBM version includes *Forecast*, a program that helps you estimate and forecast your federal taxes. (For the Apple systems it is available as a separate package that integrates with *Dollars and Sense*.) It allows you to: (1) assign all tax-related accounts to 51 tax categories; (2) make up a personal tax profile detailing your filing status, exemptions, etc.; (3) modify tax parameters that may change from year to year; (4) modify sales and regular tax tables for your profile to reflect yearly and state variations; and (5) estimate your probable liability. The most enjoyable option is being able to set up 5 different tax scenarios for comparing your strategies side-by-side.

The overall quality of *Dollars and Sense* is a toss-up. Where it's good, it's very, very good; where it's bad . . . well, you know the rest. HCM

# The Music of Sound

## **Part 2:** *Music Software for the C-64 vs. Casio's CT-6000 Keyboard*

**by Wayne Koberstein**
*HCM Staff*

***In Part 2, we compare the C-64 sound machine —with its wealth of music software— to the kind of dedicated musical instrument that originally inspired these SID-based programs . . .***

In last issue's "Music of Sound" article, I sorted through 16 software packages that allow home users to turn their C-64 computers into a myriad of different musical instruments—all synthesized by the Commodore sound chip. These products have many things in common—including their basic purpose. But what inspired the creators of these software music machines? Are they striving for some ideal?

In fact, each package is, to some degree, modeled on a new kind of musical device: the synthesizer/keyboard. Having invaded 80's rock and jazz music—totally replacing more "traditional" instruments in some cases—keyboards that can simulate whole ensembles are now beginning to invade the home. First, toy-like "mini-boards" appeared, followed by more up-scale models. Marketed chiefly by Casio Corp., the first units sported tiny keys, preset instrument sounds, and limited rhythm accompaniment. Later models continue to evolve into bigger and better machines. The latest and greatest is Casio's CT-6000, with a suggested retail price of $999.

Sound and music software for the C-64 emulates both of these types of keyboards and, with some packages, more sophisticated synthesizers. At this point, we should ask: How successfully does the C-64, with its impressive hardware and varied software, compare to a dedicated musical device like the CT-6000? Can more advanced programs do even better? And, what does the future ultimately promise for this musical computer?

### Looking Back

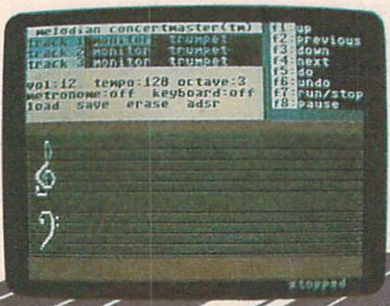
Last issue, we looked at software packages from the Sight & Sound Music, Sequential, Waveform, and Colortone companies which, taken together, represent a great variety of the musical "tasks" possible on one machine: the Commodore 64. (Figure 1 compares the major features of these programs to the CT-6000.)

For example, Sight & Sound takes us from the limited "home organ" of the *Incredible Musical Keyboard* to the nearly full exploitation of the C-64 SID chip in *3001: A Sound Odyssey*. Each package gives the Commodore machine a different face (or "mouth," if you prefer). Especially distinctive are the personalities that Ryo Kawasaki has bestowed on the machine with his two packages: the *Kawasaki Synthesizer* and the *Kawasaki Rhythm Rocker*. His *Synthesizer* opens up the sound chip with interesting preset sounds, special effects, and recording features. His spectacular *Rhythm Rocker* adds "drum tracks" and a user-controlled light show to create a kind of personal discotheque. (Note: Both the *Incredible Musical Keyboard* and the *Kawasaki Rhythm Rocker* have been updated since last issue's review. These additions are reflected in Figure 1.)

*The Kawasaki Rhythm Rocker* both imitates and, in some ways, goes beyond the musical role filled by the (unaided) CT-6000. As a computer program, it can use some of the computer's best features: the video screen, and the ability to store information on a separate medium. However, the *Kawasaki Rhythm Rocker* shares one significant limitation with every other software package that I reviewed in the previous issue: the limited available memory and slow processing speed of the Commodore 64. This means simply that however creative the software, you cannot expect the computer to *play* as many instruments at once as its specialized counterpart—nor to *respond* in such a human-like way to your keyboard "licks."

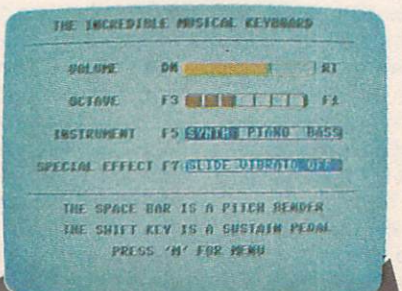
Surely, the CT-6000's major advantage over the Commodore machine is its ability to produce eight separate sounds (or "voices") at a time—to the C-64's three. No

*ConcertMaster from Melodian.*



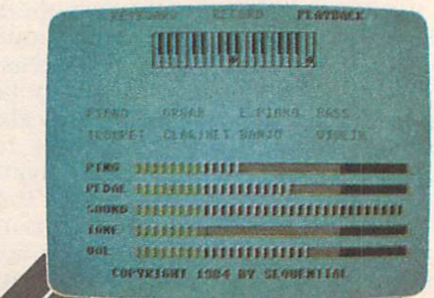
```
melodian concertmaster(tm)   f1:up
track 1 monitor  trumpet     f2:previous
track 2 monitor  trumpet     f3:down
track 3 monitor  trumpet     f4:next
                             f5:do
vol:12  tempo:128 octave:3   f6:undo
metronome:off  keyboard:off  f7:run/stop
load  save  erase  adsr      f8:pause

                                 stopped
```

*The Incredible Musical Keyboard from Sight & Sound*

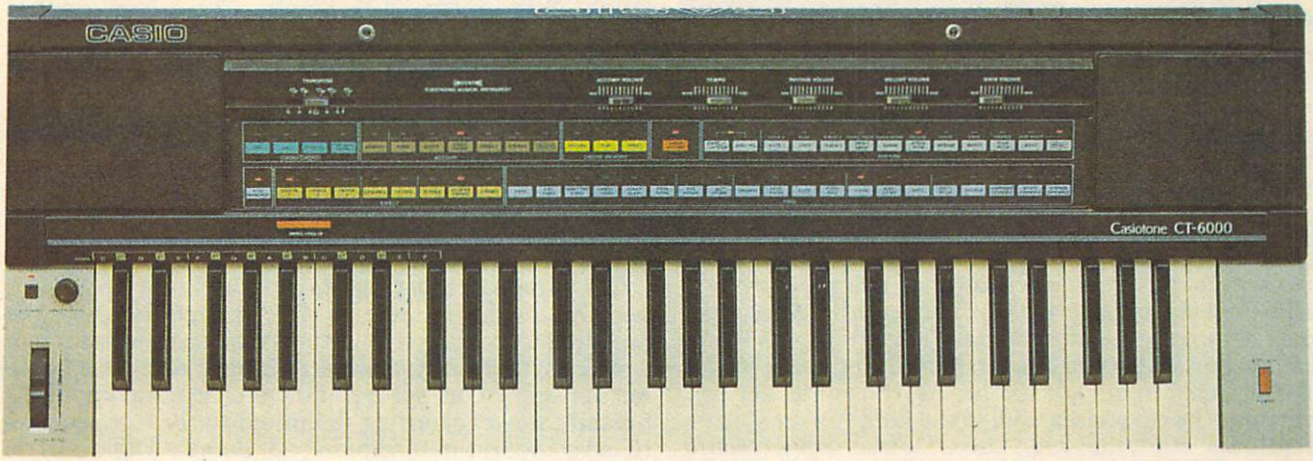


```
THE INCREDIBLE MUSICAL KEYBOARD

VOLUME          DN                  RT

OCTAVE          F3                  F4

INSTRUMENT      F5 SYNTH  PIANO  BASS

SPECIAL EFFECT  F7 SLIDE VIBRATO OFF

THE SPACE BAR IS A PITCH BENDER
THE SHIFT KEY IS A SUSTAIN PEDAL
PRESS 'M' FOR MENU
```

*The MusicMate Keyboard from Sequential*



```
KEYBOARD    RECORD    PLAYBACK

PIANO    ORGAN     E.PIANO  BASS
TRUMPET  CLARINET  BANJO    VIOLIN

PING
PEDAL
SOUND
TONE
VOL
COPYRIGHT 1984 BY SEQUENTIAL
```



> *". . . although the CT-6000 can simulate many different instrument sounds, the C-64 can be many different instruments."*

**Wow!**

When I first listened to the CT-6000, I couldn't help but be impressed with its 8-voice polyphonic sound—or *sounds*. Sitting down at this 61-key touch-sensitive stereo board is an immediate pleasure; it is extremely easy to produce very satisfying music with a variety of well-programmed instrument sounds. Two long rows of touchpad control keys line the top of the keyboard, encouraging experimentation and, well . . . *play*. Throw in the ability to simulate a total of 30 different instruments and 20 different rhythms, to record accompaniment tracks (in memory), to transpose the whole keyboard in semitone steps from G to F#, to mix in a variety of "special effects," even to provide an accompaniment that will "jam" along with you—and you have a music machine that's hard to beat—W.K.

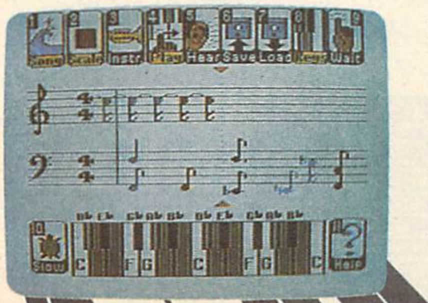
matter what the software package, and no matter how well the C-64's sound compares to other computers, the Commodore machine cannot compete with the superior hardware of the Casio keyboard. A computer's hardware must fulfill many functions, and any one piece of it must share space and cost with other pieces committed to other functions. To use a medical analogy, dedicated units like the CT-6000 are *specialists*, but computers are *general practitioners*.

### It's Programmable

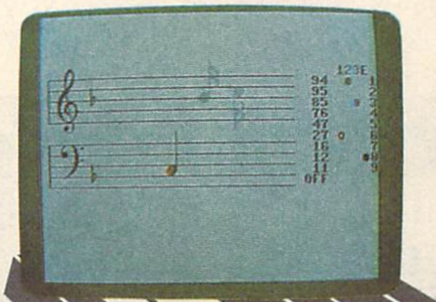
So far, it sounds as if there is *no* comparison to be made between the top-of-the-line Casio board and the C-64, with its growing bevy of music programs. But Commodore owners take heart! Your computer has some intrinsic advantages of its own.

Although I have made a distinction between the C-64 as a computer and the CT-6000 as something else, both are—technically speaking—computers. Both use binary code processed by microcircuits to perform their tasks. Both literally "compute." But their real difference lies in what, as computers, they are designed to do. Casio's keyboard is, as I've said, *dedicated* to producing specific sounds and music. Its circuitry contains hard-coded

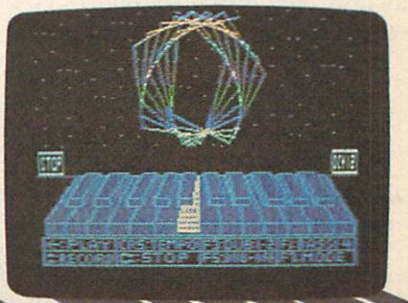
*Colortone Keyboard from Waveform.*

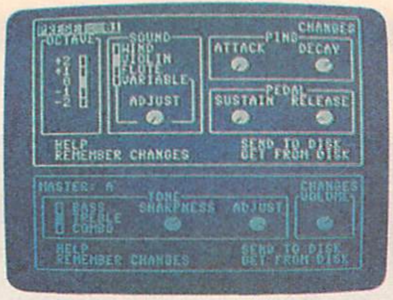


*Music Processor from Sight & Sound.*

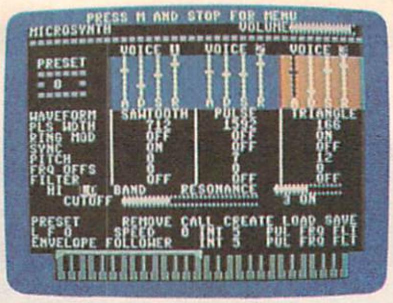


*Kawasaki Rhythm Rocker from Sight & Sound.*





*Sound Maker from Sequential.*



*3001: A Sound Odyssey from Sight & sound.*

programs—called "firmware"—that are either difficult or impossible to change. In contrast, a computer such as the C-64 is *programmable*. It can perform many functions, depending on the software loaded into its RAM. A dedicated keyboard may respond and perform its specific tasks more effectively, but the computer can re-arrange its responses and, in the long run, be a more useful tool. And, although the CT-6000 can *simulate* many different instrument sounds, the C-64 can *be* many different instruments.

## Jamming In Real Time

One major advantage that a specialized keyboard does have over a computer is its ability to perform in *real time*, quickly processing the huge chunks of information that make up its total sound. For the Casio CT-6000, this advantage allows the machine to actually *play along* with you as if it contained a small back-up band. In Super Accompaniment mode, a lower bank of keys triggers specific chords. These chords aren't just *sounded*, they are *played* in a pattern that changes in response to your timing and the notes that you play. A built-in "drummer" (Sync rhythm) keeps time, while adding little breaks and fills to complement both the accompaniment and your playing. The entire effect of the machine responding so *intelligently* in real time is almost spooky.

None of the C-64 music programs reviewed in Vol. 5, No. 2 provide this ability to "jam" in real time. Again, limited memory and sound-producing circuits would make such a provision difficult—but not totally impossible—on the Commodore computer.

Yet, each of the packages reviewed takes an original path to sound and music, adding other features that are not possible with the CT-6000 alone. These include music-printing programs—like *Song Printer* by Sequential and *ScoreWriter* in Waveform's *MusiCalc* series—and sequencing routines that allow detailed manipulation of notes and rhythms. (With the *MusiCalc* series, you can even reconfigure the keyboard to play many different musical scales.) The most ambitious programs not only offer preset sounds, but allow you to create your own *original* sound sets.

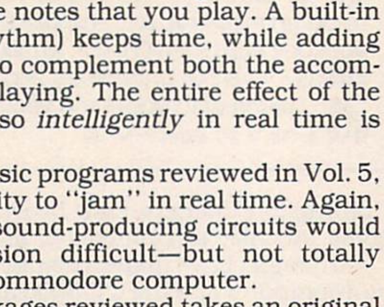
## Recording Revisited

Computers can permanently store binary code in a portable medium, such as disk or tape. Because the essential information that makes up a sound setting or a series of notes consists of this binary code, it is possible to use the computer as a recording instrument. At present, most recording is limited to the "pure" synthesized sound in a sequence of notes. "Digital Sampling," in which the computer measures real sounds 10,000 times a second and translates them to digital code, is still used mostly for speech, rather than music. [See our review of the PCjr Speech Attachment elsewhere in this issue of *HCM*—Ed.] With the CT-6000, recordings remain in memory only—there is no disk drive or cassette player to store them—and when you turn off the machine, you may lose hours of careful work. Most of the C-64 sound programs reviewed in part one of this article allow you to store your recording on disk.

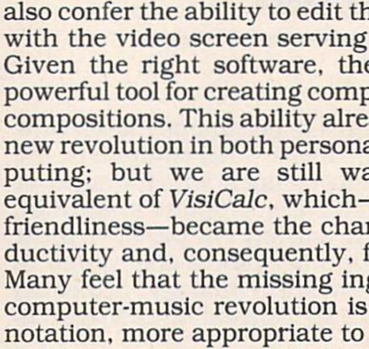
In addition to their *storage* capabilities, computers also confer the ability to edit this musical information, with the video screen serving as an editing medium. Given the right software, the computer becomes a powerful tool for creating complex musical (and visual) compositions. This ability already promises to spark a new revolution in both personal music and home computing; but we are still waiting for the musical equivalent of *VisiCalc*, which—by its clarity and user-friendliness—became the champion for personal productivity and, consequently, for personal computers. Many feel that the missing ingredient in the pending computer-music revolution is a new form of musical notation, more appropriate to the computer medium, and thus more accessible to most computer users.

## MIDI: The Ace Card

Although I have compared the general advantages of each system—the dedicated keyboard versus the Commodore computer and software—the CT-6000 has one more card to play: its built-in MIDI interface. Casio has wisely included this feature as a gateway to the world of personal computers, thus probably extending the keyboard's useful life.



*Kawasaki Synthesizer (playing screen).*



*MusiCalc I from Waveform.*

With the proper software, this interface could be turned into an advantage for *both* systems—giving the CT-6000 the ability to store and edit information on disk, and the C-64 greater sound-producing power. Compositions made with a program like *Song Builder* could, for example, be fed to the CT-6000—where they would play "live" in Super Accompaniment Mode. In fact, MIDI makes possible many different combinations or "patches" between keyboards, drum machines and, yes, *your* home computer. All of this cross-breeding will in turn contribute to the evolution of the ultimate home music system. MIDI will also contribute to the development of the home digital recording studio by adding processing and sampling power to a computer-based system. In short, MIDI can bring more sophisticated instruments into the home computer environment, as it has already brought the ambidextrous home computer into the professional studio.

The features that MIDI could add to the CT-6000 are listed in Figure 1—but currently, there is no software to my knowledge that uses MIDI to connect the CT-6000 to the C-64. [Sequential, however, offers a $99 MIDI cartridge to connect 3 of its synthesizers—the Max, the Six Trak, and the Multi Trak—to the C-64. Also, the Roland company of Los Angeles offers several MIDI products for the Apple and IBM personal computers—Ed.]

## Let's Hear It

Perhaps some of you are the developers who will provide such ground-breaking software. Others may be just awakening to this new world of computer music. Still others may already be among the true believers. Perhaps you have your own ideas on this subject, reactions to these reviews, or more related information that you would like to share with our readers. If so, we would like to hear from you. Please write soon and address any such responses to our "Letters to the Editor" column. And stay tuned for more on the *Music of Sound*.

HCM

**FIGURE 1**

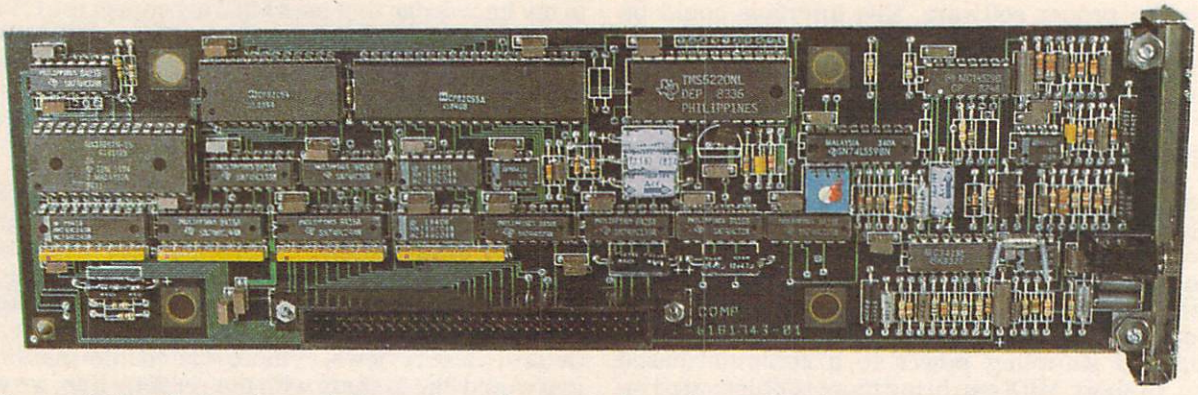
**Comparative Features**
**C-64 Keyboard Software vs. Casio CT-6000**

| | Casio CT-6000 | | Sight & Sound | Sequential | Waveform | Melodian |
|---|---|---|---|---|---|---|
| Features | w/o MIDI | w/MIDI | a Incredible Musical<br>b Music Processor<br>c Kawasaki Synthesizer<br>d Kawasaki Rhythm Rocker<br>e 3001: Sound Ody. | a MusicMate<br>b Sound Maker<br>c Song Builder<br>d Song Editor<br>e Song Printer | a Colortone<br>b MusiCalc 1<br>c MusiCalc 2<br>d MusiCalc 3 | a ConcertMaster<br>b RhythmMaster |
| 8 Oscillators (Voices) | X | X | | | | |
| 3 Oscillators (Voices) | | | a b c d e | a b c d | a b c | a b |
| Preset Sounds | X(30) | X(30) | a (20) b (99) c (21) d (3) e (100) | a (8) b (20) | a (8) b (32) | a (19) |
| Preset Rhythms | X(20) | X(20) | b d (4) | | | |
| "Sync" Rhythms | X | X | | | | |
| Preset Songs | | X | a b c d e | a | a b | a |
| Tempo control | X | X | b c d e | c | a b | a b |
| Polyphonic Play | X | X | a c d | a | b | a |
| Create Preset Sounds | | | b c e | b | b | a |
| Create Preset Rhythms | | X | b c e | b | b | a |
| Auto Accompaniment | X | X | | | | |
| Score Display | | X | a b d | c d | a | a b |
| Record (Memory) | X | X | a b d e | a c | a b | a |
| Record (Disk) | | X | a b d e | c | a b | a |
| Sequencing | | X | b c | c d | b | |
| Touch-sensitive Keys | X | X | | | | |
| Reconfigure Keyboard Scale | | | | | b d | |
| Print Scores | | X | d | e | c | |

# Junior's First Words

## A Review of the PCjr Speech Attachment

**By Roger Wood**
*HCM Staff*



***One of the latest peripherals released for the PCjr is a speech synthesizer. But is it the last word in speech synthesis?***

A child's first few intelligible words mark a momentous event in a parent's life. Likewise, when your own home computer starts to talk, you really begin to listen. The future of computers and their interface to our human world is definitely linked to speech and speech recognition. IBM's *PCjr Speech Attachment* is the first effort with PCjr to bring a part of this exciting field into the home—but it doesn't appear to give the home user enough for the money.

### Installation and Testing

As with all IBM hardware expansion units, the *PCjr Speech Attachment* installs quickly and easily on the right side of the machine, and requires only a flat-blade screwdriver. Three test procedures are detailed in the *Testing Added Options* manual included with the unit. These are added to the regular Test Menu Screen which is accessed by simultaneously pressing (Alt), (Ctrl), and (Ins).

The first test is a series of spoken words, and the second is simply a tone. The third test highlights one of the few exciting features of the unit: the ability to digitally record sound and play it back. You will need to have a 600-ohm microphone with a miniature audio phone plug (3.5 mm) to do this, but it's a feature not available with many speech units. I'll discuss this in more detail below.

If you have a television with an RF modulator, you should hear these speech-unit tests through the TV's speaker. A word of warning, however, for anyone who is using a composite-video, or an R-G-B (Red, Green, Blue) monitor rather than a TV with an RF-modulator: You will need an external amplifier and speaker to use the *PCjr Speech Attachment*. [This is true of using the PCjr's internal sound chip as well. For details on accessing the internal sound chip, see *JR Sounds Off* in *Home Computer Magazine* Vol. 4, No. 2—Ed.] Unfortunately, the *PCjr Sound Attachment* documentation does not explain this, and users unfamiliar with the need for external amplification might easily think they purchased a defective unit.

### Making Junior Talk

Once you've finished with the tests, you'll want to make your computer say what *you* want. This is accomplished entirely through BASIC (either Cartridge or Cassette). IBM supplies a short and easy-to-follow manual which includes several key-in BASIC programs—there is no disk or tape supplied. Although the programs are not long, a typing error when keying-in the DATA statements may cause the computer to get lost while trying to link and return from the speech routines.

First, a couple of short subroutines described as your "speech toolkit" allow you to pass parameters to, and link-up with, the *PCjr Speech Attachment's* BIOS (Basic Input/Output System). The manual is careful to tell you not to " . . . be concerned about how these subroutines work: just type them as follows . . ." I was a bit disappointed at the lack of explanation, but once the routines are keyed-in properly they *are* easy to use.

The other four program segments in the manual all require this toolkit in order to work. I SAVEd the toolkit routines in ASCII format so that I could MERGE them into any program where I wished to use speech. Seven parameters are passed to the linking routine using BASIC variables labeled P1 through P7. As you work through the other routines, the purpose of most of these variables is made very clear—specifying mode, or word selection, etc. Two parameters (P4 and P5) "tell the *Speech Attachment* where in memory to store the recorded sounds and where to get the sounds to play back." The manual goes on to refer the user to the *IBM PCjr Technical Reference Manual* for details—but I found nothing to illuminate me there.

For those with a more technical bent, my limited investigations revealed that these toolkit routines get the seven parameters which are defined in any BASIC speech program. These then use the INTerrupt vector at hex $4D to pass control to the speech BIOS.

This brings me to my major complaint about the manual: the technical section was not really technical enough. The routines are explained fine for use by a novice programmer—but there is nothing to help more adventurous souls learn how to access the speech BIOS routines. In the *PCjr Technical Reference Manual*, a complete assembly-language listing is available, so an ambitious programmer can access the PCjr's normal BIOS routines. An IBM representative told me that there is no such support for the *PCjr Speech Attachment* at present.

### From Memory or Microphone

Two modes are available with the *PCjr Speech Attachment*: (1) Vocabulary mode, and (2) Record/Playback mode. Vocabulary mode is further divided into Foreground and Background. Foreground means that the BASIC program waits for any speech routine to be completed before doing another BASIC operation—such as PRINTing. Background mode allows the BASIC program to continue executing its instructions while the speech is occuring. This could allow an animated graphic scene to be displayed while a speaking commentary describes the action.

Although this would seem to make the *PCjr Speech Attachment* an extremely exciting tool, it has one crippling drawback—the vocabulary has a total of only 196 words and sound effects. Several speech attachments and boards are available for home computers—such as the TI-99/4A's *Speech Synthesizer*, or the *Cricket!* for the Apple *II* Family of computers. [*Home Computer Magazine* has published numerous articles covering the *TI Speech Synthesizer*. See *HCM* Vol. 4, No. 5 for a review of the *Cricket!*—Ed.] Both of these products make it easy to access a vocabulary of 400 to 500 words. With some programming (in the case of the *Cricket!*), or an added *Terminal Emulator II Command Module* (in the case of the *TI Speech Synthesizer*), many words can be phonetically created. At this time IBM has not come forward with any such programmable additions to their *Speech Attachment*. With only 196 words, and no way to program new ones, this product suffers from a severe limitation.

### Recording—Exciting But Primitive

In Record Mode, the *PCjr Speech Attachment* breaks new ground in the field of home computer speech. By attaching a 600-ohm microphone (I used an inexpensive Radio Shack model, catalog #33-986), you can digitally record any sound. When you RUN the *Record/Playback* demonstration program listed in the manual, it records 5 seconds of sound and stores the data in an integer array DIMensioned to 25000—that's 50000 bytes of memory! This gives you an idea of the memory-intensive nature of digital recording.

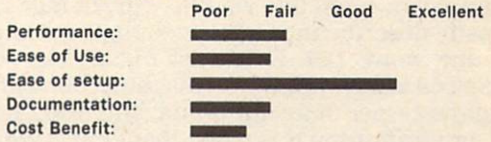
> *"With only 196 words, and no way to program new ones, this product suffers from a severe limitation."*

This type of digital recording works by sampling the signal from the microphone several thousand times per second. Generally, the sampling rate must be at least twice as fast as the highest frequency of the sound to be recorded, in order to get a clean, undistorted signal. The samples are converted from analog signals to digital data and stored in memory. The playback of this data entails converting the digital data back to an analog signal that can be amplified and reproduced by the speaker.

The highest quality possible with the *PCjr Speech Attachment* samples at 4800 bytes per second, which implies that frequencies above 2.4 kHz (thousand cycles-per-second) will not be recorded accurately—i.e., without noise or distortion. For speech, this is more than adequate. The lowest quality recording that the *PCjr Speech Attchment* makes available is 1800 bytes per second. I found this to be *very* low quality, even for speech; it is quite noisy with static.

|  |  |
|---|---|
| Name: | IBM PCjr Speech Attachment |
| Product Type: | Speech synthesis |
| Machine: | IBM PCjr |
| Distributor: | IBM Corp.<br>Boca Raton, FL 44432 |
| Price: | $300 |

System Requirements:IBM PCjr and color TV with RF modulator, or video monitor and external sound amplifier and speaker.



As mentioned above, the highest quality recording is limited to a length of 5 seconds. By reducing the number of bytes per second to the minimum, thus reducing the quality, you increase the length of the recording time to approximately 15 seconds. No easy way exists to increase the amount of memory accessible from Microsoft BASIC (64K for program and variables combined), so any *PCjr Speech Attachment* BASIC program is severely limited to very short recording times.

I was able to save my recording to disk by using the VARPTR command to locate the array in memory, and then BSAVEing it to disk. Then I could BLOAD the array back into another sound program where the recording could be played back. Each of these disk accesses took about 25 seconds. This means that in order to play back 5 seconds of high quality sound from a disk file, your program must wait 25 seconds to get the data. My conclusion is that, although the Record/Playback mode is a fascinating feature, it has no real practical application for the BASIC programmer.

### Too Little for the Money

While talking with IBM representatives about this product, they told me that there is no additional speech-development software available from them. In addition, such speech development systems normally run in the thousands of dollars. The sound chip used in the *PCjr Speech Attachment* is the TI-5220 (similar to the technology used in the TI-99/4A's *Speech Synthesizer*). IBM said that any vocabulary expansion for the PCjr's unit would have to come via Texas Instruments. In terms of state-of-the-art sound synthesis, the *PCjr's Speech Attachment* seems years behind the times.

The only software product IBM produces that supports the *PCjr Speech Attachment*, and thus might increase its vocabulary, is the educational package called *Writing to Read* . This product is meant to be used by educational institutions—it's not practical for the normal home budget. [See *New Product News* in *HCM* Vol. 4, No. 4 for details on this school-system-oriented product—Ed.]

The recording feature is the product's one new enhancement—but memory limitations make it only a toy digital recorder. In short, unless you have a spare $300 to spend on a woefully underpowered speech and digital recording system, look elsewhere for computer add-ons. HCM

# Commodore-Taming for the Shrewd

## A Review of Sysres

by Randy Thompson
*and the HCM Staff*

***Although powerful, the C-64's lack of editing and debugging features makes programming it a "hacker's nightmare." Originally written for PET and CBM systems, this programmer's tool takes the "hack" out of programming.***

If you've ever done any programming on the Commodore 64, then you have probably wished for a more cooperative computer. Due to the Commodore's lack of built-in programming aids, writing a program can seem like waging war. In fact, one could honestly describe the C-64's debugging commands in just one word: LIST. Don't get me wrong—the Commodore 64 is a very powerful machine, but without such things as a trace or an automatic line-numbering function, programming it is truly "hacking." Well, here to help end this battle between computer and hacker is the program *Sysres*.

*Sysres* is what is often referred to as a *program development system*. It is used to aid the writing and debugging of software. The relationship between a programmer and a development system like *Sysres* is very similar to that of a writer and a word processor. By adding features that enable the programmer to manipulate computer programs in much the same way a word processor manipulates text, the development system becomes an invaluable tool. And like a book written on a word processor, programs created with the aid of *Sysres* are totally independent. In other words, you don't need *Sysres* to RUN a program, and you don't need a word processor to read a book. As the *Sysres User's Guide* so aptly puts it, *Sysres* is "addictive to programmers but *not* addictive to programs."

### No Preservatives, Just Additives

*Sysres* adds a "disk wedge" program and 33 commands to your computer. [For a complete explanation of a disk wedge, see the "Home Computer Tech Note" for the Commodore 64 in *HCM* Vol. 4, No. 3—Ed.] Most of these commands fall into two basic categories: disk manipulation, and editing and debugging. (See Figure 1 for a list of these commands.) Explaining every command that *Sysres* contains could fill a magazine in itself, so here are just a few examples.

### Editing

The most powerful of the editing functions are probably the FIND and CHANGE commands. FIND will search a program for any combination of characters and list its results. CHANGE is a programmer's search and replace. It not only finds things, but it also changes them to your liking.

Both of these commands conduct intelligent searches. If you prefer, you can search only the beginning or ending of program lines. You can limit your search to certain line numbers. Maybe you only want to find an item if it is contained in quotes. Whatever your choice, the options are there.

*Sysres* provides many time-saving commands. You can erase several lines at once by using the DELETE statement, a common feature of more powerful BASICs. The AUTO command will save you a lot of typing effort by automatically providing evenly incremented line numbers when keying in programs. An added feature of the AUTO command is its ability to supply all or part of a program line in addition to the line number. If you're entering several lines that are very similar, such as a series of IF-THENs, this feature can be very useful.

My favorite of the time-saving commands is KEY. This command allows any shifted key to be defined as a BASIC command, a string, or whatever you wish. Just think: with the push of a button, you could list a disk's directory, change the screen's color, or save an updated version of the program in memory. You could define keys to list specific subroutines that you are working on. Certain keys are automatically predefined by *Sysres* to output many of the common BASIC commands, saving you the time of typing them in.

The most practical of the editing commands is RENUMBER. *Sysres* spared no expense here. Unlike so many renumbering routines I've used, this one works. Not only does it renumber GOTOs, GOSUBs, and the ON statements related to these two, but it also renumbers RUNs, LISTs and GO TOs. A GO TO is identical in operation, but different in the way it is stored in memory, from its more compact sibling GOTO. Unfortunately, many authors of resequencing routines forget this little quirk and do not account for it within their programs.

RENUMBER also performs various juggling tricks with program lines. With the aid of the MERGE command, whole blocks of code can be moved by renumbering them. How about having all your line numbers the same? It's not very practical, but possible if you use a renumber increment of zero. Believe it or not, such a program will RUN, providing you don't care where a GOTO sends you.

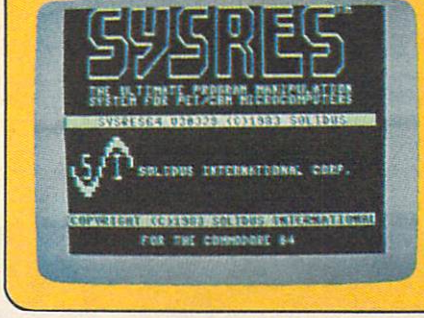
### Losing Your Memory?

Uh-oh, you've accidentally typed NEW. Not to worry—just type OLD, and your program will be saved from an early retirement. For those of you who have added a reset button as described in the "Home Computer Tech Note" for Commodore (Vol. 4, No. 4 of *HCM*), you may have noticed that using this button resets the BASIC pointers, thus NEWing any program in memory. By loading in *Sysres* and entering OLD, you can now resurrect that program. How's that for a time saver?

### Debugging

Does SYNTAX ERROR appear to be your computer's favorite phrase? Well, now you can reply with WHY. When you enter WHY, *Sysres* will list the offending line with the error accented in reverse video. With the TRACE command, you can monitor a program's execution. TRACE will list the line numbers being executed and/or inform you of any changes in the values of your

## HCM Review



SYSRES™

THE ULTIMATE PROGRAM MANIPULATION SYSTEM FOR PET/CBM MICROCOMPUTERS

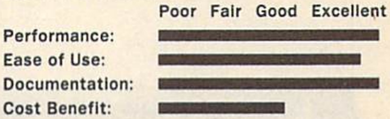
SYSRES64 030329 (C)1983 SOLIDUS

SOLIDUS INTERNATIONAL CORP.

COPYRIGHT (C)1983 SOLIDUS INTERNATIONAL

FOR THE COMMODORE 64

Name: Sysres
Program Type: Program development utility
Machine: C-64
Distributor: Solidus International Corporation
2151W. Holly, Suite 241
Bellingham, WA 98225
(206) 734-3744
Price: $95
System Requirements: Commodore-64, a color TV or monitor and a 1541 disk drive.



variables. To check your variables when a program is not RUNing, use DUMP. All nondimensional variables and their values will be listed.

### Disk Commands

When you buy a VIC-1541 disk drive, you are supplied with a demo disk containing a program called the *C-64 Wedge*. *Sysres* supports about 95% of the *Wedge*'s commands, and adds the ability to MERGE and APPEND programs (see Table 1). Both of these functions are surprisingly fast considering the operating speed of the 1541. If you are one of those who likes to build a program from a library of subroutines, then these two commands are for you. [For those of you who are *ON DISK* users, the MERGE command of *Sysres* is tailor-made for merging the updated files to your old programs—Ed.]

### All The Extras

*Sysres* doesn't just add commands; it modifies many of the pre-existing ones. For instance, program listings can be scrolled with the cursor keys. You can also freeze listings with the space bar. The SAVE command has been given a save-with-replace feature and both the SAVE and LOAD commands now default to the disk drive.

> *"Once you've used Sysres, you'll never start a programming session without it."*

For non-Commodore printers, *Sysres* provides an ASCII conversion mode. *Sysres* will even print formatted listings to your printer. You can dump an entire screen of text or graphics to the printer simply by pressing [CTRL] and [CMD] simultaneously. Also, the output of any command can now be sent to the printer by preceding that command with an asterisk. This feature is very useful when used with the TRACE command.

Ever been bit by an infamous C-64 bug? It usually happens while you are just adding the finishing touches to that all-important program. You try scrolling up to edit the last line on the screen and the computer locks up. Well, with *Sysres* installed this little bug is gone.

### Documentation

Unlike so many manuals for Commodore products, the documentation for *Sysres* is complete, and after months of use I've found it to be error-free. Each command is well-explained and illustrated with easy-to-follow examples, and the commands are listed alphabetically for quick referencing. You can tell that a great deal of time was spent writing this manual, making it easy to use and understand.

### Faults?

I have only one real complaint about *Sysres*. Occasionally it will print a line or two of garbage onto the screen after ending a program, or saving one to disk. This flaw creates no serious problems, but it can be a visual annoyance.

Some people may complain that *Sysres* takes too much time to learn. It is true that like any complex piece of software, you may have to read the manual—but believe me, the time spent doing it is paid back in full.

### Conclusion

Once you've used *Sysres*, you'll never start a programming session without it. It is a complete and faithful program. How I ever programmed without it, I'll never know. For taking the hack out of hacking, *Sysres* is definitely a best buy.

HCM

**FIGURE 1**

**Disk Commands Available with Sysres**

| Command | Function |
|---|---|
| @ | Display disk status |
| @N | Format or "New" a diskette |
| @I | Initialize the 1541 drive |
| @V | Validate a diskette |
| @D | Duplicate a diskette |
| @C | Copy or concatenate a disk file |
| @R | Rename a file |
| @S | Scratch or delete file(s) |
| @$ | List directory |
| @U: | Reset disk drive speed |
| @L | List disk file |
| / | Load from disk |
| ↑ | Load from disk and RUN |
| APPEND | Append from disk to current program |
| BLOAD | Load machine language (binary) file |
| BRUN | Load and RUN machine language program |
| CLOSE | Close one or all files |
| CMD | Send output to a file (does not send "READY") |
| EXEC | Execute a file as keyboard commands |
| GET | Read a sequential file into editor |
| LOAD | Default to disk drive |
| MERGE | Merge program from disk into memory |
| PUT | Send a program to disk as a text file |
| SAVE | Default to disk drive |
| SETD | Set disk device #, allow multiple drives |
| VERIFY | Compare current program against disk/tape |

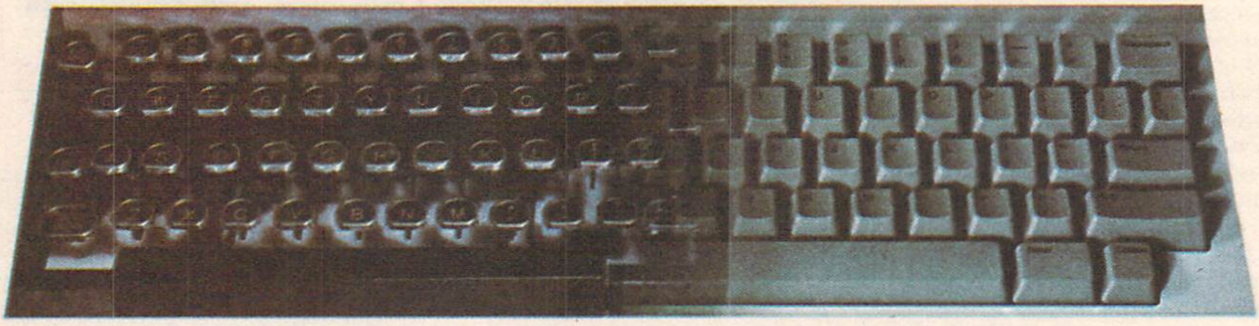
**Editing and Debuggging Commands Available with Sysres**

| Command | Function |
|---|---|
| AUTO | Auto line number |
| CHANGE | Change pattern to another pattern |
| DELETE | Delete range of lines from program |
| DUMP | Dump all scalar variables to screen or file |
| FIND | Find occurrances of a pattern |
| KEY | Define a key as a special function |
| KEYS | Turn on key functions |
| KILL | Disable Sysres |
| KILL* | Disable Sysres and unreserve memory |
| LIST | Improved BASIC LIST command |
| MON | Break to current machine language monitor |
| OLD | Restore program after new |
| RENUMBER | Renumber all or part of program |
| RUN | Run current program, ignore screen garbage |
| SETP | Set printer channel, format mode, paging |
| TRACE | Select 1 of 3 trace/step modes and speeds |
| WHY | Print position of last error |
| WHY? | Last line of break or error |
| * | Send output to printer |
| # | Display current version of Sysres |

# COMPUTER-AGE TYPING:

## A LOOK AT SOME KEY TYPING-INSTRUCTION PROGRAMS

**by Dana M. Campbell**
*HCM Staff*



***With all of the programs available out there waiting to teach us how to type or help us type better and faster, how do we know what is a bargain, and what to avoid?***

The more we learn about the human body, the more we realize how important exercise is for it. But although exercise may be generally regarded as healthy for the body and essential for its well-being, that doesn't mean that people always enjoy it, or even do it. Part of the problem may be attributed to the dreariness of traditional calisthenics, for along with the spread of novel, enjoyable ways to exercise (aerobics classes, jogging, health clubs, etc.) came droves of newly converted exercise nuts.

The same can be said of the importance of learning to type or improving existing typing skills. Now that computers are marching into the home, the school, and every office from the main floor up to the executive suite, typing ability has become a crucial survival skill in the technological age. Unfortunately, traditional typing classes and manuals generate about as much enthusiasm as the thought of drinking a glass of prune juice. Well fear not, modern one—with the exciting, ever-changing technology of the computer come brand new ways of learning typing skills. Today's available typing software spans the range from edge-of-your-seat, typing game excitement to more subdued fingering techniques. Here we take a look at several typing instruction programs in relation to a list of desirable typing program traits, as shown in Figure 1.

> *"Any computer typing-tutor program worth its salt as a learning medium should be able to take the information, process it, and use it to help you in future lessons . . ."*

### Games People Play

By offering a game option, it's possible for typing-tutor programs to provide a truly engrossing way to exercise typing skills. The more innovative game/learning programs make good use of the computer medium, stepping beyond the capabilities of traditional typing manuals. For instance, *MasterType* and *WizType* are almost entirely based on a game format, and they are certainly not of the drudge school of learning.

*MasterType* in particular offers a visually exciting way to learn and improve, as well as a great deal of flexibility. In this program, the user is a wizard who must defend his spaceship against an onslaught of words scattered around him in space. The words are defended by missiles, which advance at a speed equal to the user's words-per-minute (wpm) rate set in practice sessions. If a word is typed before its missile reaches the wizard, it and its missile are destroyed with a mighty zap by the old necromancer, only to be replaced by a new word. Similarly, *WizType*—which features the comic strip characters Wizard and Spirit from *The Wizard of Id*—levitates words or letters in the castle basement. As they are typed, the Wizard zaps them. If the words are typed too slow, Spirit turns into a dragon and torches the Wizard. Again, the words must be typed at a rate set by the user.

*Space Invaders* look-a-likes abound in the game options of *Keyboard Command*, *Touch Typing Tutor*, and *Typing Tutor III*. Though these 3 programs are more serious in format, they all offer variations on the *Space Invaders* theme—here letter combinations and increasingly difficult words and symbols drop out of the sky at faster and faster rates, and must by typed before hitting the baseline, cityscape, (fill-in the appropriate screen scene), etc. However, *Typing Instructor* takes a dive with its Lobster Sea Adventure. You simply type a paragraph accurately and quickly enough to stay ahead of the lobster (which looks nothing like a lobster) before it "eats" you.

Generally, the game element seems to help decrease the boredom of drill work by making these programs *interesting*, and that's not just my opinion. *MasterType*, for example, has been on top of the educational-software charts for more than 2 years.

### Can You Put Your Finger On It?

Whether lifting weights or trying to type, knowing the proper way to execute the exercise when you first begin will save you from picking up harmful bad habits later on. *Touch Typing Tutor* and *Typing Instructor* begin by detailing the proper typing posture and hand position and carefully continue on. *Type 'N' Write* merely highlights the proper keys to press on a screen reproduction of the keyboard, and it does this only when in the first of 8 modes.

Worse yet, *Typing Coach* assigns each home-row finger a color, and uses colors on the screen keyboard to clue you in to which keys to press with each finger. This essentially provides a crutch that you come to rely on, hindering the process of learning which letters to associate with each finger. It's adding an unnecessary extra step: First you must remember that the forefingers are red—later you learn that red stands for the F and J keys.

Brief written and visual guides on proper keyboard fingering accompany the other typing programs at the start of each new lesson. But because the depth and presentation of beginning finger-placement instruction varies so much between programs, you should view a demonstration of several of these programs before buying to determine what is appropriate for you (or your child, or whoever the intended user is).

### As Long As It's Accurate . . .

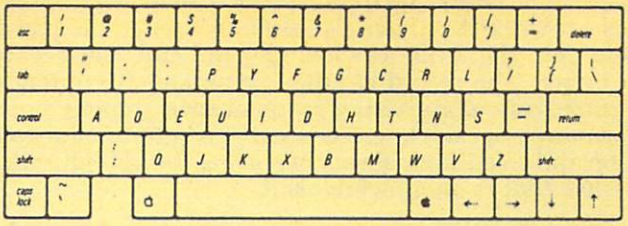
The phrase "Do it until you get it right," usually spoken by angry teachers and parents, seems to be the over-riding philosophy threaded through these programs. Apparently, most typing-program developers believe that given enough repetitious exercise, anyone can improve their typing accuracy. It's a sad theory, and I hate to advocate anything that promotes drudgery, but it seems to be true. For instance, *Typing Coach* has 28 lessons, each introducing 2 new keys. You type various letter sequences that include those two keys—and any others that you learned in previous lessons—for 10 lines. You may continue to repeat the exercise as long as you want for each lesson. Believe me, when you finish a lesson, you *know* those keys.

*Type 'N' Write* is an attractive alternative for kids, with colorful icons of children at play guiding you toward keyboard success. Here again you type letter sequences, continuing to repeat the same line until infinity or until you complete it with fewer than 5 errors, whichever comes first. It may be tedious, but it does work.

### Zipping Along

Speed building on the track or on a keyboard is synonymous with racing—either against a competitor, or against your own time. In a typing context, wpm tests for comparison against your own previously set rate are the most common way used in typing software to build speed. Some programs simply time you as you type during the regular lessons, while others have separate, specific speed tests and drills. The latter are the ones indicated with a YES answer in Figure 1. *Typing Tutor III* provides several timed test options, covering just words, just numbers, the full keyboard, and a "standard" speed test of up to 7 lines of words to type. These are good options to look for when shopping for speed-building programs.

*Touch Typing Tutor*—the only program that we could find still being distributed for the TI-99/4A—is ponderous to use, especially for speed building. Part of the problem stems from the 99/4A's keyboard layout. The fact that you must reach for the (FCTN) key to produce such punctuation as a question mark, and apostrophe necessarily slows you down. If you are already nimble on a typewriter keyboard, with its standard layout of punctuation keys, you will certainly want to adapt to your TI machine before you start timing yourself. But even without the keyboard differences, this program provides little impetus to increase typing speed.



### Dvorak: A Keyboard Alternative

If you are just learning to type and you have an Apple *IIc*, you might consider trying the Dvorak keyboard layout, as opposed to the QWERTY layout common to most typewriters and computers. The original keyboard design that we still use today (with the letters Q, W, E, R, T, and Y appearing on the second row, left side) was arranged to deliberately slow down typists so that the mechanical keys on the old-time typewriters would not tangle and jam up. On a Dvorak keyboard, the 10 home-row keys (A, O, E, U, I, D, H, T, N, and S) comprise 70 percent of all letters typed, according to extensive studies on the subject. Thus, even those people who are already accomplished typists can learn this new keyboard and greatly increase their typing speed.

On the U.S. version of the Apple *IIc*, the (Keyboard) switch above the keyboard toggles between the Dvorak and the QWERTY layouts. By carefully popping off the keys on the *IIc* and arranging them in the Dvorak configuration shown in the IIc manual, you can load up *MasterType* or *WizType* and choose their Dvorak options to learn a whole new modern way of typing. The lessons are constructed in the same way as they are for the QWERTY keyboard, except that the keys are learned in a different order, with both versions starting with their respective home rows.

### Don't Forget The Numbers . . .

A quality exercise program would not exclusively focus on the torso and forget the extremities, and neither would a good typing program. All of the programs listed in Figure 1 provide an opportunity to practice skills on the top row of numbers and the punctuation keys. Number drills are, in most cases, separated from the letter drills. Although people who simply want to brush up on those top-row keys may like the separation, others who appreciate the practice that numbers and letters thrown together provides will probably miss the option to do so.

### The More Records, The Better

Not to be overlooked by homo-sapien typing tutors is whether these programs are capable of supporting scores, lessons, and progress records for more than one user. *MasterType* and a few others keep track of high scores for their games, but *Keyboard Command* seems specially suited for classroom use. Its Student Management File (a separate data-base program on the disk) is accessible only by a password, and it will hold the records of up to 100 students. Once in the file, the teacher, parent, or user can preselect the lesson number (from 1 to 19 covering all keys), the skill concentration (accuracy, speed, or both), and whether the keyboard should be shown on screen during the lesson. A user may also edit, add, delete, or scroll through and view student records, which consist of a student's name, last lesson worked on, average drill accuracy and speed, whether the drill was completed, and the record number. The record list may be alphabetized and/or printed out, along with a graph of each student's performance on each key, if desired. It's quite a handy option and an additional way of putting the old computer to use that is often overlooked.

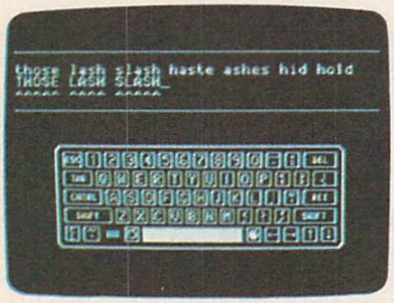
## Create Your Own Lessons

The latest in home video and computer health programs have options that let you adjust the program to suit your own needs, and *MasterType*, *WizType*, and *Typing Tutor III* are no different in this respect. *MasterType* allows you to create a lesson file of 40 words, each no longer than 10 characters. They can be words often used and specific to a certain profession, spelling

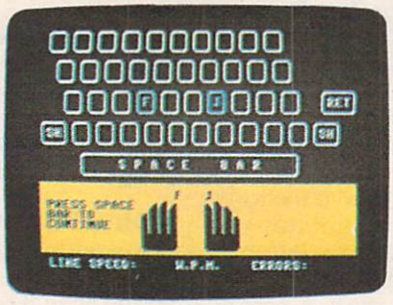
**FIGURE 1** **Some Popular Typing Programs and their Features**

| | Key Command<br>🍎 | Type 'N' Write<br>C64 | Typing Instructor<br>PC PCjr | Touch Typing Tutor<br>TI | MasterType<br>🍎 C64<br>PC PCjr | Typing Coach<br>🍎 C64<br>PC PCjr | Typing Tutor III<br>🍎 C64<br>PC PCjr | WizType<br>🍎 C64 |
|---|---|---|---|---|---|---|---|---|
| Game Option | YES | | YES | YES | YES | | YES | YES |
| Teaches Finger Placement | YES | YES | YES | YES | YES | YES | YES | YES |
| Builds Accuracy | YES | YES | YES | YES | YES | YES | YES | YES |
| Builds Speed | YES | | YES | YES | YES | | YES | YES |
| Numbers & Symbols | YES | YES | YES | YES | YES | YES | YES | YES |
| Multiple Records | YES | | | | | | YES | YES |
| Lesson Creation | | | | | YES | | YES | YES |
| Analysis Incorporated | | | YES | YES | YES | | YES | |
| Progress Displayed | YES | | | | | YES | YES | |
| Sentences & Paragraphs | | | YES | YES | YES | | YES | YES |
| Dvorak Keyboard | | | | | YES | | | YES |

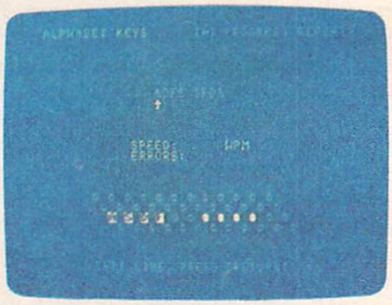
1


those lash slash haste ashes hid hold
THOSE LASH SLASH_

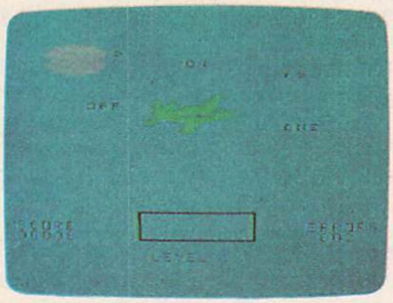
2



3

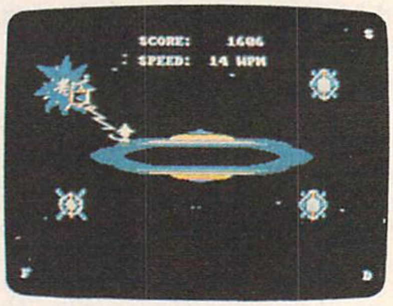


4

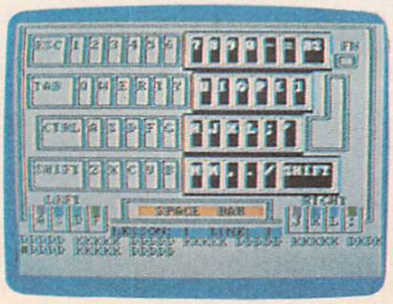


1. *Key Command.*
2. *Type 'N' Write.*
3. *Typing Tutor III.*
4. *Touch Typing Tutor.*
5. *MasterType.*
6. *Typing Coach.*
7. *Typing Instructor.*
8. *WizType.*

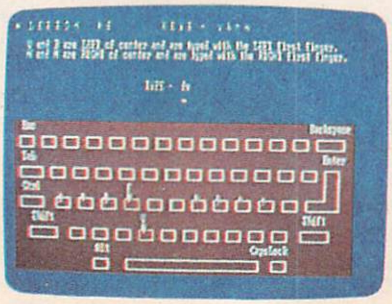
5



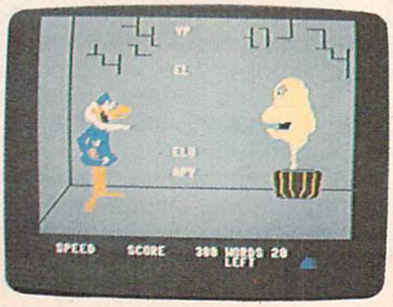
6



7



8



words for grade schoolers, or any other word type. *WizType* allows you to do the same, only it's capable of supporting a list of 240 words no longer than 8 characters each. You can also enter your own text for practice in Paragraph mode.

*Typing Tutor III* lets you write your own Standard Typing Test text of no more than 7 lines, but only on its IBM PC version. It's a shame this option isn't also offered on the Commodore and Apple versions. With these, the same copy appears again and again—so when your speed continues to dramatically improve, it is probably due to your familiarity with the material. In any case, a lesson-creation option certainly increases any typing program's flexibility.

### Functional Feedback

Any computer typing-tutor program worth its salt as a learning medium should be able to take the information input—your typing strengths and weaknesses—process it, and use it to help you in future lessons and tests within the same program. Unfortunately, it appears that only 3 programs here do this. Following each lesson in *Touch Typing Tutor*, the keys on which more practice is needed are displayed, along with the numbers of the lessons where they appear. Another practice option lets you work on these problem keys a little more.

*MasterType* repeatedly displays letters and words that you miss or are slow to type, and *Typing Tutor III* has something called TRM—Time Response Monitoring. This program monitors the time it takes you to type each letter and then uses that information to create new lessons tailored to your needs.

> ***"By offering a game option, typing-tutor programs provide an engrossing way to practice typing skills that makes good use of the computer medium, stepping beyond the capabilities of traditional typing manuals."***

### At-A-Glance Progress Reports

I believe that the ability to directly compare past and present performance in any endeavor does provide some motivation for steady improvement. Even though graphs and other visual representations of a person's performance usually are not integral for a program's use, they provide an instant picture of your progress and problems. *Typing Coach* keeps a chart of your old and current wpm speed and number of errors for each one of its 28 lessons. *Keyboard Command* allows you to print out or simply display a bar graph of your percentage of accuracy and speed for every single key on the keyboard. Finally, *Typing Tutor III* keeps a list of your "fast" and "slow" keys, and also graphs your accuracy.

### Sentences: Putting It All Together

You can learn to do an unlimited number of exercises, but unless you perform an adequate number of each one in your routine, you will be doing only a superficial workout and won't build much stamina. So it is with the discipline of typing. Unfortunately, only one of the programs reviewed here displays more than a word or a few sentences to type at a time. This is fine when initially learning the keys. But in the everyday world, most information comes in bulk form, usually in paragraphs. *WizType* is the only program that provides extensive text samples to practice your speed and skill, and what samples they are! The text choices include pieces from *Tale of Two Cities*, *Hamlet*, *Mother Goose*, Typing Trivia, the Declaration of Independence, the Gettysburg Address, *Cremation of Sam McGee*, and Facts About the Wizard of Id. You can choose to have Bung—the *Wizard of Id*'s drunk court jester—pace you as he jumps on his pogo stick; or, you can try typing at your leisure. However, nowhere does the program keep track of the number of errors you make when typing a drill or paragraph.

Perhaps the next generation of typing programs will take this extra step and provide lengthier samples to provide more experience in dealing with consistency, rhythm, and fatigue.

### In Summary

Perhaps the greatest advantage that typing-tutor programs offer is the ability to immediately calculate your words per minute and number of errors. And those programs that offer a game element obviously provide a stimulus for increasing speed that no typewriter and manual can offer. Some programs instantly appeal to both youngsters and adults, while others take a drier, more regimented approach to typing instruction and work just as well. All of them are incredibly easy to use, with the barest of documentation required because everything is menu-driven, and, in most cases, it is easy to get into and out of various modes. The "key" to finding the *best* program *most* suited to your needs is to first determine what those needs are, and visit a software dealer to actually try out and compare back-to-back those that are available for your machine. Then sit down and give your fingers a real workout! **HCM**

### *Typing Instruction Software*

***Keyboard Command***, for the Apple *II* Family, retails for $34.95 and is available from Compu-tations, P.O. Box 502, Troy, MI, 48099.

***Type 'N' Write***, for the Commodore 64, retails for $24.95 and is distributed by Human Engineered Software, 150 North Hill, Brisbane, CA 94005.

***Typing Instructor***, for the IBM PC and PCjr, retails for $49.95 and produced by Individual Software, 1163-1 Chess Dr., Foster City, CA 94404.

***Touch Typing Tutor***, for the TI-99/4A, retails for $19.95 and is distributed by Triton Software Co., P.O. Box 8123, San Francisco, CA 94128.

***MasterType***, for the Apple *II* Family, Commodore 64, and IBM PC and PCjr retails for $39.95 and is available from Scarborough Systems, Inc., 25 North Broadway, Tarrytown, New York, 10591.

***Typing Coach***, for the Apple *II*, II+, and *IIe*, Commodore 64, and IBM PC and PCjr retails for $24.99 and is distributed by K-tel Software, Inc., 11311 K-tel Dr., Minnetonka, MN., 55343.

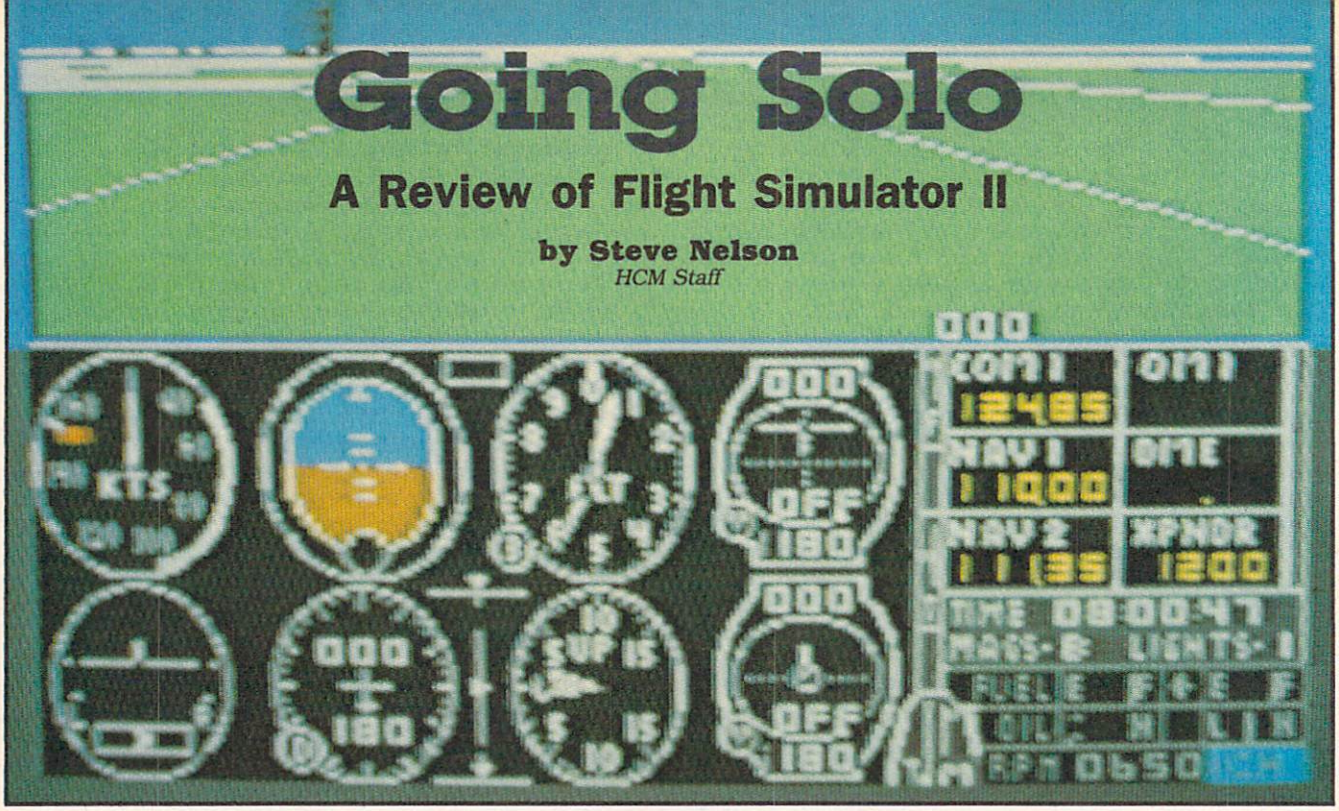
***Typing Tutor III***, for the Apple *II* Family, Commodore 64, and IBM PC retails for $49.95 and is distributed by Simon & Schuster Electronic Publishing, 1230 Avenue of the Americas, New York, NY, 10020.

***WizType***, for the Apple *II* Family, Commodore 64, and IBM PCjr retails for $34.95 and is available from Sierra On-Line, Inc., Coarsegold, CA 93614.

# Going Solo

## A Review of Flight Simulator II

**by Steve Nelson**
*HCM Staff*



***Wondering if Flight Simulator II is as good as people say? We tell you why it has soared to the top of the charts.***

*A sense of nervous anticipation settles over me as I perform my preliminary flight check. The rudder, ailerons, and elevators all respond perfectly to the faint pressure of my fingers on the controls. A quick check of my on-board navigation equipment shows me that the NAV radio receivers are working, and I enter my flight-plan coordinates. Sitting at the controls, I hear the faint but reassuring thump thump of the motor and I wipe a bead of sweat from my brow. This is it. My first solo flight. One more look around just to double check and I'm ready. All systems are go. Taking a deep breath, I slowly open the throttle to obtain full power, and begin taxiing down the runway while watching the airspeed indicator climb: 25 knots, 30 knots. At 55 knots, I raise the nose of the plane and instantly feel the stomach-flipping sensation of being airborne. I'm on my own . . .*

*Flight Simulator II* (FS2) is a real-time flight simulation program for the IBM PC and PCjr, the Apple *II* Family including the *II*c, and the Commodore 64 computers. It puts you at the controls of a single-engine plane (a Piper 181 on the Apple and C-64 versions, and a Cessna 182 in the IBM versions). With three-dimensional graphics and full flight instrumentation, this is a simulation that doesn't leave much to the imagination.

> ***"In the simulator you can fly over 100 million square miles—including the continental United States, Canada, Mexico, and the Caribbean."***

### Instant Airport Hopping

In the simulator you can fly over 100 million square miles—including the continental United States, Canada, Mexico, and the Caribbean. You can take off from, or land at, 80 different airports. But because *Flight Simulator II* is a real-time simulator, it would take you hours to travel from New York to Los Angeles. You would probably get very bored along the way (no movies on this flight), so the program also features two speedier methods of getting from point A to point B: setting destination coordinates, and "slewing." Setting destinations is the fastest method. Simply enter the Editor (more about this later), set the North and East parameters of your destination along with the altitude that you want to be flying at when you arrive, press the E key (to leave the Editor), and you are there.

With the second method, called "slewing," you can move very rapidly from one location to another while watching the scenery fly by as if "compressed" to another scale. Again, you must enter the Editor and set the slew parameters (there are 2). Then, using the keyboard controls, fly the plane to your destination. It sounds real easy, but be prepared for a wild ride. On my first attempt at slewing I thought I had joined Lee Majors in an episode of *The Fall Guy*—I barrel-rolled for the first 200 miles. The slew keys are essentially the same as the regular flight keys, but they operate a little differently—you can only control your altitude and direction.

### Turbulence Anyone?

If flying the same old skies bores you, don't worry—the people at Sublogic have thought of everything. Using the Editor, you can add all sorts of interesting effects to make your trip more thrilling. For instance, you can take a night flight, relying only on your instruments to guide you to your destination. Or, you can change a beautiful day into a cloudy one. How about flying through gale force winds for a little excitement? If you like a bumpy ride, you can reset the turbulence from very low to very high.

The Editor gives you access to 30 flight modes. Ten of them are preset, ranging from easy flight to (emergency) airborne startup situation—don't try this one first.

You can change the flight parameters by entering the Editor (a menu of parameters will appear on the screen). In the normal flying mode, the plane automatically starts for you, and you fly under what Sublogic calls easy-flight conditions. In this mode, the plane is very forgiving, although you can still have some spectacular crashes. Once you become accustomed to flying, you will probably want to change to the "reality" mode to make things a little more true to life. In this mode, you must start your motor using the magneto switch and starter before taking off, you may run out of gas if you are not careful, or your carburetors may ice up; you can even kill the motor at a crucial moment (like when you are trying to pull out of a dive at 5000 feet). All this and more helps make your trip as real as possible.

### Where Am I?

It's easy to get confused about where you are once you are up in the sky and have no recognizable point of reference. *Flight Simulator II* features two of the most frequently used navigational aids to help you determine where you are, and where you are going. In real life, radio stations called VORs (Very high frequency Omnidirectional Range) transmit an omnidirectional signal, followed by a circular sweeping directional signal, which the NAV receiver in your plane detects and uses to determine the "radial" or the position of your plane in reference to the transmitting station. This is called getting a "fix." Suffice to say that once you are airborne, you can find your way around using your plane's NAV receiver and the simulated signal from the VOR station.

> ***"The 3-D view from the cockpit is uncanny in its realistic simulation of the horizon spinning away when doing a sharp bank, or going topsy turvy as you perform a loop-the-loop."***

### What a View

While flying around, you may want to check out the scenery. You have a full field of vision from inside the cockpit: front view, back view, and right and left side views. You can switch back and forth between views with just a keypress. The 3-D graphics are quite good, especially the aerial views as you are flying over an airport or city. Everything that you see on screen really exists—I once tried to take out the John Hancock Building while attempting to land at Meigs Field in Chicago.

Name: Flight Simulator II
Program Type: Flight simulation game
Machines: Apple II Family, C-64, IBM PC and PCjr
Distributor: (Apple, C-64 versions)
Sublogic Corporation
713 Edgebrook Drive
Champaign, IL 61820
(IBM PC & IBM PCjr versions)
Microsoft Corporation
10700 Northup Way
Box 97200
Bellevue, WA 98009
Price: $49.95
System Requirements: Disk drive, color monitor recommended

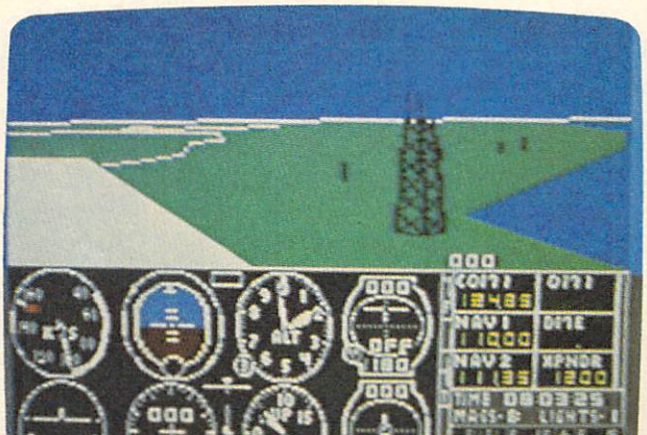
| | Poor | Fair | Good | Excellent |
|---|---|---|---|---|
| Performance: | ▬ | ▬ | ▬ | ▬ |
| Engrossment: | ▬ | ▬ | ▬ | |
| Documentation: | ▬ | ▬ | ▬ | ▬ |

The graphics are virtually identical on all three machine versions, with the edge in sharpness going to the IBM PC and PCjr versions. The instrument panel is easy to read and reacts instantly to any change you make. All three versions simulate the sound of the engine as you are flying, as well as a stall horn and the sound of the propeller windmilling when you are in a powerless glide. The sound effects had a major influence on my preference of machine versions. The C-64 version was far and away the most realistic and enjoyable to fly—mainly because of its excellent reproduction of the Piper's motor noise. The IBM's and Apple's sound effects were almost annoying in comparison—fortunately, you have the option to turn them off.

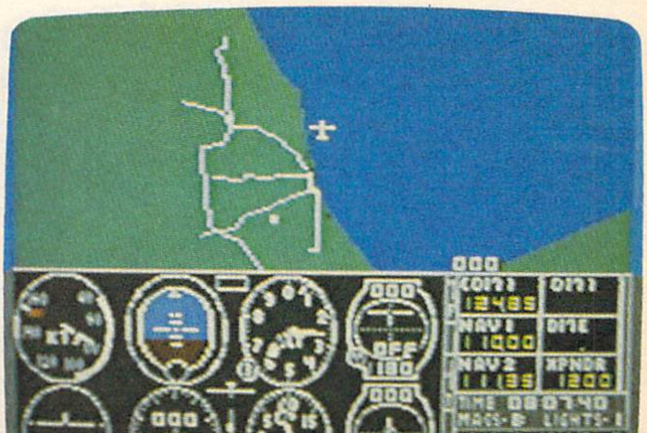
Using the keyboard to fly a plane may sound kind of weird, but it works pretty well. You have the option to use a joystick if you wish, and some people may prefer it because it may seem more like the controls inside a plane. The plane's response to the controls is excellent—maybe even too good, because you can get out of control very easily. The main thing to remember when flying is to be *very gentle* with the controls until you become comfortable with them. The keyboard controls are arranged so that you can access all controls with one hand.

*Continued on next page.*

*A view of the control tower after take-off.*



*A "radar" screen shows your position in this wide-angle view.*



### Super Documentation

*Flight Simulator II* comes with two manuals: the airplane flight manual and operating handbook, which gives you the basics on how to fly your particular aircraft; and another manual on flight physics and aircraft control, with an introduction to aerobatics. This second manual is more like a short flying course. In fact, there are 8 lessons, and working through them will have you flying high in no time. Both manuals cover the subject of flight in depth, and it takes a concerted effort just to get through it all, especially if you are not already a pilot. The operations manual will get you aloft, but if you want to fly on the wild side, the introduction to aerobatics will have you doing the Immelman turn before you know it.

> ***"Included in the program is World War I Ace, a game where you put your flying skills to the ultimate test—dogfight!"***

### Loop the Loop

Once you "get your wings" and feel confident about flying, you may want to put your new-found skills to a test. In the normal mode, easy-flight conditions can be changed to simulate a more realistic flight, as I mentioned earlier. In the easy-flight mode, your plane's controls are auto-coordinated—meaning that the rudder and the ailerons are linked, giving you the safest flight possible. You can change to uncoordinated flight using the Editor, which lets you control the ailerons and the rudder separately. Now you can begin performing aerial maneuvers, like flying sideways (slipping and skidding), which help when landing in windy conditions. You can also start to do the fun stuff, like figure eights, loops, spins, and other aerobatic maneuvers. Attempting these maneuvers is a blast, as long as you don't let yourself get frustrated from crashing too often. The 3-D view from the cockpit is uncanny in its realistic simulation of the horizon spinning away when doing a sharp bank, or going topsy turvy as you perform a loop-the-loop.

Crashing your plane is painless for both you and the *Flight Simulator II*. You simply start over again from the airport where you last took off. But biting the dust on the Apple or the IBM versions leaves something to be desired—you hear only a pipsqueak sound as the plane impacts the ground or water. I have to admit, it kind of left me flat (no pun intended). Not so with the Commodore version: when you go down, there is a satisfying thud as your plane smacks good ol' terra firma.

### Der Red Baron

If you are a dyed-in-the-wool video gamer, and just have to have something to shoot down, the people at Sublogic haven't forgotten you. Included in the program is World War I Ace, a game where you put your flying skills to the ultimate test—dogfight! In this game, you are the pilot of an ancient biplane armed with a machine gun and bombs. Your mission: destroy the enemy bases, fuel depots, and factories, and shoot down as many enemy fighters as you can. Making the Kaiser's crack pilots eat hot lead isn't as easy as you might think, because they are trying to do the same thing to you. If you can down 5 enemy planes, you become a WWI flying ace. Your aerobatic skills and pinpoint accuracy are the only things that will see you through the coming battles. Again, the C-64 version is clearly the superior. Its sound effects are outstanding. When you go on a bombing run and let one rip, or machine-gun down an enemy plane, it sounds just like a John Wayne movie.

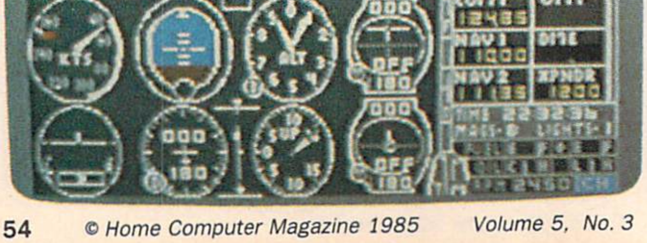
*Flight Simulator II* has been at the top of the best-seller lists for more than 110 weeks the last time I checked, and I have to admit that I was hoping that I could find something to really complain about when I took my first flight. But the opposite happened. After logging a substantial number of FS2 flying hours, I must tell you that I am impressed. This is one program that will meet your expectations. The only thing I can complain about is the sub-par sound effects on the Apple and IBM versions, which may be more of a hardware problem than a programming flaw. The actual flying itself can be rather boring if you are just going from point A to point B—but each time I was aloft, I found myself pretending to be a barn stormer or Charles Lindburgh. And if I ever really got bored, I would press [ESC] and access the Editor, and try to win WWI all over again. *Flight Simulator II* is difficult, challenging, and lives up to its "lofty" reputation.

### Landing

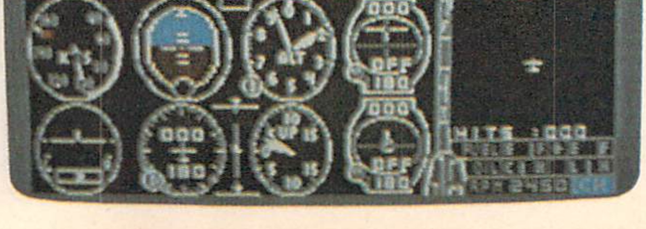
*There it is, the airport landing strip. I check my air speed as I make my approach: 80 knots and slowing. Altitude 120 feet, elevators up, I seem a little shaky. Oops—my nose is too low. I pull back on the stick and increase the throttle; she settles down. Altitude, 90 feet, air speed, 65 knots. Holding steady, easing back on the throttle now. Altitude is dropping, 75 feet, 50 feet, going into my flare, gently now, straightening out. Altitude 30 feet, air speed, 60 knots, I ease off on the throttle and let her stall out. For a moment I feel like I'm the only person in the whole world. I touch down, take a little hop, and stabilize. Apply the brakes, not too hard. I've made it . . .*

**HCM**

*Night-flying is easy if you watch your instruments.*



*Looking through the sights of the World War I Ace Biplane.*



# Build a LOGO Adventure

## Part 3 of 4

**by Andrew Keith**

*and the HCM Staff*

**In this installment, the brave hero interacts with his surroundings and can acquire rich treasures as we develop the procedures for expanding this "gripping" adventure.**

If you have been following this series, you know that we have come a considerable distance in the development of a LOGO Adventure. Not only do we have a way to make the computer understand our typed-in English commands, but we have taught it to also read a map of our surroundings and move us around from place to place. In this part we are going to introduce a new and very necessary element to the development of our LOGO Adventure scenario: We will place objects and treasures in various locations on the map, and then discuss how to use *list processing* to enable us to pick them up, put them down, and keep an inventory of what we have.

List processing is one of the cornerstones of LOGO's power. The basic concept is that any list of objects, thoughts, etc.,can be manipulated or processed using a few simple commands. In our case, we will create lists containing numbered "keys" to tell us where an object is located, or how visible the object is. Then, whenever we use one of our "verb" procedures to act upon an object, we can search our list to take appropriate action.

## One List Creates the Objects

First, we must create a new list, called "ITEMS. Each element will contain the name of an item followed by the room number in which it can be found (see the map from the previous issue or look at "ROOMS for the room numbers in the listing ). "ITEMS is established by putting it into the SET.UP procedure of last time (see Listing 1 for the fully updated SET.UP procedure).

Notice that in :ITEMS, the flute and the sword are considered to be in Room #50, yet there are only twelve rooms! These objects are in a "hidden" location. They won't be in plain sight, so the Adventurer will have to do special things to obtain them (more about this next time).

In addition, a new variable, named "MINE, has been introduced to SET.UP. Initially set to zero, it will be used to keep count of the number of objects that have been picked up or dropped. If the number stored in :MINE exceeds three, it means that the player is carrying the maximum of four items and will not be allowed to pick up anything more without dropping something else first (see Listing 2 for this issue's modified TAKE procedure.)

Once SET.UP is executed, the items are placed in their respective locations. However, when walking from room to room, we still cannot see any change. We must add two new procedures to allow us to see what is in each room: TO LOOK, and TO LOOKAT (Listing 3).

LOOK is called by the ID.LOC procedure (also in Listing 3) to inform the Adventurer of the objects visible in a particular room. When executed, LOOK sets up and clears to zero the local variable "SEE?, which functions as a *flag*, and then executes LOOKAT. If :SEE? still contains a zero upon return, it means that there was nothing seen in that location, and LOOK prints this message.

The LOOKAT procedure tests the last element of the first list in :ITEMS to see whether the room number of that item matches our current location. If it does match, it prints out the name of the item it sees while setting the flag :SEE? to a value of one. Then, it calls itself recursively, looking at the BUTFIRST of :ITEMS. It continues this recursive check, printing out all that it sees until the list is empty—at which point it encounters STOP and returns to LOOK.

LOOK is not the only procedure that calls LOOKAT. The I procedure (also in Listing 3), which prints out an inven-

### Listing 1

APPLE LOGO II
LOGO SYSTEMS LOGO — IBM PC & PCjr

```
TO SET.UP
MAKE "VERBLIST1 [N S E W U D I]
MAKE "VERBLIST2 [TAKE DROP TOSS]
MAKE "VERBLIST (SE :VERBLIST1 :VERBLIST2)
MAKE "ROOMS [[1 IN A FOREST] [2 IN A GLEN] [3 AT THE ENTRANCE OF A CAVE] [4 IN THE LAIR OF A DRAGON] [5 IN A TREEHOUSE] [6 IN A MARSH] [7 IN A STONY PASSAGE] [8 BEFORE A CASTLE] [9 IN A RAVINE] [10 UNDER A TREE] [11 ON A MOUNTAIN PATH] [12 IN THE WITCH'S
CHAMBER]]
MAKE "DIRECTIONS [[. . 2 . . .] [. 6 3 1 . .] [. . 7 2 . .] [. . . . 7 .] [. . . . . 10] [2 10 . . . .] [. 11 . 3 . 4] [. . . 11 . .] [. . 10 . . .] [6 . . 9 5 .] [7 . 8 . . .] [. . . . . .]]
MAKE "ITEMS [[[A JAR] 1] [[FLIES] 6] [[A FLUTE] 50] [[A SWORD] 50] [[A BUCKET] 11] [[A CROWN] 9] [[A HOOK ATTACHED TO A ROPE] 4] [[WATER] 6] [[A SCEPTRE] 12]]
MAKE "HERE 2 MAKE "MINE 0
END
```

TERRAPIN LOGO — C-64

```
TO SET.UP
 MAKE "VERBLIST1 [N S EW U D I]
 MAKE "VERBLIST2 [TAKE DROP TOSS]
 MAKE "VERBLIST ( SE :VERBLIST1 :VERBLIST2 )
 MAKE "ROOMS [[1 IN A FOREST] [2 IN A GLEN] [3 AT THE ENTRANCE OF A CAVE] [4 IN THE LAIR OF A DRAGON] [5 IN A TREE HOUSE] [6 IN A MARSH] [7 IN A STONY PASSAGE] [8 BEFORE A CASTLE] [9 IN A RAVINE] [10 UNDER A TREE] [11 ON A MOUNTAIN PATH] [12 IN THE WITCH'S CHAMBER]]
 MAKE "DIRECTIONS [[. . 2 . . .] [. 6 3 1 . .] [. . 7 2 . .] [. . . . 7 .] [. . . . . 10] [2 10 . . . .] [. 11 . 3 . 4] [. . . 11 . .] [. . 10 . . .] [6 . . 9 5 .] [7 . 8 . . .] [. . . . . .]]
 MAKE "ITEMS [[[A JAR] 1] [[FLIES] 6] [[A FLUTE] 50] [[A SWORD] 50] [[A BUCKET] 11] [[A CROWN] 9] [[A HOOK ATTACHED TO A ROPE] 4] [[WATER] 6] [[A SCEPTRE] 12]]
 MAKE "HERE 2 MAKE "MINE 0
END
```

### Listing 2

APPLE LOGO II
LOGO SYSTEMS LOGO — IBM PC & PCjr

```
TO DROP :OBJ
IF NOT HAVE.IT? :OBJ [PR (SE [YOU DON'T HAVE THE] :OBJ ".) STOP]
IF AND :OBJ = "BUCKET HAVE.IT? "WATER [PUT.IN :HERE "WATER MAKE "MINE :MINE - 1]
IF AND :OBJ = "JAR HAVE.IT? "FLIES [PUT.IN :HERE "FLIES MAKE "MINE :MINE - 1]
PUT.IN :HERE :OBJ MAKE "MINE :MINE - 1
PR [DROPPED.]
END

TO TAKE :OBJ
IF NOT HERE? :OBJ [PR [THAT CAN'T BE DONE HERE.] STOP]
IF :OBJ = "FLIES [IF NOT HAVE.IT? "JAR [PR [YOU HAVE NOTHING TO PUT THEM IN.] STOP]]
IF :OBJ = "WATER [TEST HAVE.IT? "BUCKET IFF [IF HAVE.IT? "JAR [PR [THE JAR LEAKS.] STOP] [PR [YOU'LL NEED TO CARRY IT.] STOP]]]
IF :MINE > 3 [PR [CAN'T - CARRYING TOO MUCH!] STOP]
PUT.IN -1 :OBJ MAKE "MINE :MINE + 1
PRINT SE [YOU HAVE THE] :OBJ
END

TO TOSS :OBJ
IF NOT HAVE.IT? :OBJ [PR SE [YOU DON'T HAVE THE] WORD :OBJ ". STOP]
TEST AND :HERE = 8 :OBJ = "HOOK
IFT [PR [THE HOOK CATCHES, WITH THE ROPE HANGING DOWN.] PUT.IN 12 "HOOK MAKE "MINE :MINE - 1 STOP]
(PR SE [TOSSING THE] :OBJ [DOES NO GOOD.])
PUT.IN :HERE :OBJ MAKE "MINE :MINE - 1
IF AND :OBJ = "BUCKET HAVE.IT? "WATER [PUT.IN :HERE "WATER MAKE "MINE :MINE - 1]
IF AND :OBJ = "JAR HAVE.IT? "FLIES [PUT.IN :HERE "FLIES MAKE "MINE :MINE - 1]
END
```

TERRAPIN LOGO — C-64

```
TO DROP :OBJ
 IF NOT HAVE.IT? :OBJ PR SE [YOU DON'T HAVE THE] WORD :OBJ ". STOP
 PUT.IN :HERE :OBJ MAKE "MINE :MINE - 1
 IF ALLOF :OBJ = "BUCKET HAVE.IT? "WATER THEN PUT.IN :HERE "WATER MAKE "MINE :MINE - 1
 IF ALLOF :OBJ = "JAR HAVE.IT? "FLIES THEN PUT.IN :HERE "FLIES MAKE "MINE :MINE - 1
 PR [DROPPED.]
END

TO TAKE :OBJ
 IF NOT HERE? :OBJ PR [THAT CAN'T BE DONE HERE.] STOP
 IF :OBJ = "FLIES THEN IF NOT HAVE.IT? "JAR PR [YOU HAVE NOTHING TO PUT THEM IN.] STOP
 IF :OBJ = "WATER TEST HAVE.IT? "BUCKET IFF THEN IF HAVE.IT? "JAR PR [THE JAR LEAKS.] STOP ELSE PR [YOU'LL NEED A WAY TO CARRY IT.] STOP
 IF :MINE > 3 PR [CAN'T--CARRYING TOO MUCH!] STOP
 PUT.IN - 1 :OBJ MAKE "MINE :MINE + 1
 PRINT SE [YOU HAVE THE] :OBJ
END

TO TOSS :OBJ
 IF NOT HAVE.IT? :OBJ PR SE [YOU DON'T HAVE THE] WORD :OBJ ". STOP
 TEST ALLOF :HERE = 8 :OBJ = "HOOK
 IFT PR [THE HOOK CATCHES, WITH THE ROPE HANGING DOWN.] PUT.IN 12 "HOOK MAKE "MINE :MINE - 1 STOP
 ( PR SE [TOSSING THE] :OBJ [DOES NO GOOD.] )
 PUT.IN :HERE :OBJ MAKE "MINE :MINE - 1
 IF ALLOF :OBJ = "BUCKET HAVE.IT? "WATER THEN PUT.IN :HERE "WATER MAKE "MINE :MINE - 1
 IF ALLOF :OBJ = "JAR HAVE.IT? "FLIES THEN PUT.IN :HERE "FLIES MAKE "MINE :MINE - 1
END
```

tory of all items being carried, uses LOOKAT. Instead of using :HERE as an input, however, it uses a value of negative one (-1). The negative one indicates that the object is being carried by the Adventurer, so LOOKAT produces a list of those items being carried rather than those seen in the room.

## Handling Objects

Once the Adventurer identifies an item, three verb procedures will allow interaction with the object: the TAKE procedure (first introduced in *Home Computer Magazine*, Vol. 4, No. 5), the DROP procedure, and the TOSS procedure (Listing 2). Note that we've used the word TOSS instead of THROW because THROW is a primative in both IBM and Apple II LOGO.

The key to these procedures lies in figuring out how to replace the room number of a particular object with a value of negative one, and vice-versa? The workhorses that will accomplish this feat are PUT.IN and PUT.IT.HERE (Listing 4).

The input :N of PUT.IT.HERE will be equal to negative one if PUT.IN is called from TAKE, because the item is being put into the inventory of the Adventurer. When PUT.IN is called from DROP, :N will equal the current room number (:HERE). This value is passed along to PUT.IT.HERE, along with :OBJ and :ITEMS in the LIST variable. The procedure then uses *list processing* to sort through :ITEMS and insert :N at the appropriate place. Finally, it outputs the reconstructed list to PUT.IN.

PUT.IT.HERE works its magic by checking to see whether the object in question (:OBJ) is a member of the first sublist in the first list in :ITEMS. If it is, LASTPUT and BUTLAST are used in combination to mask out the original value and substitute :N. Conversely, FIRSTPUT and BUTFIRST are used to mask out the original first list in :ITEMS and substitute the newly reconstituted one, thus updating the present state of the :ITEMS list.

## Interacting

The remainder of this month's additional procedures (Listing 5) will allow you to get a taste for the Adventure we've been building over the past few issues. With this month's procedures added to the ones in *Home Computer Magazine*, Vol. 4, No. 5 and Vol. 5, No. 1, you will be able to collect and move items around. With all the procedures presented thus far in memory, you simply type ADVENTURE (as explained last issue) and you're on your way.

Next issue we will add living beings to create challenge and give you assistance in your LOGO Adventure.

HCM

### Listing 3

APPLE LOGO II
LOGO SYSTEMS LOGO — IBM PC & PCjr

```
TO ID.LOC↵
PR [ ]↵
PR SE [YOU ARE] BF ITEM
 :HERE :ROOMS LOOK↵
IF :HERE = 8 [PR [THERE
 IS AN OPEN WINDOW HIGH
 ABOVE] TEST LOOKTHRU :
ITEMS "HOOK 12 IFT [PR
[WITH A ROPE HANGING FR
OM IT]]]↵
IF :HERE = 7 [PR [YOU S
EE SOME DIMLY LIT STONE
 STEPS.] STOP]↵
END↵

TO I↵
LOCAL "SEE? MAKE "SEE?
0↵
PR [YOU ARE CARRYING:]↵
LOOKAT :ITEMS -1↵
IF :SEE? = 0 [PR [NOTHI
NG]]↵
END↵

TO LOOKAT :LIST :N↵
IF EMPTYP :LIST [STOP]↵
IF (LAST FIRST :LIST) =
 :N [PR FIRST FIRST :LI
ST MAKE "SEE? 1]↵
LOOKAT BF :LIST :N↵
END↵

TO LOOK↵
LOCAL "SEE? MAKE "SEE?
0↵
PR [YOU CAN SEE THE FOL
LOWING ITEMS:]↵
LOOKAT :ITEMS :HERE↵
IF :SEE? = 0 [PR [NOTHI
NG]]↵
PR [ ]↵
END↵
```

TERRAPIN LOGO — C-64

```
TO ID.LOC↵
 PR [ ]↵
 PR SE [YOU ARE] BF ITE
M :HERE :ROOMS LOOK↵
 IF :HERE = 8 PR [THERE
 IS AN OPEN WINDOW HIGH
 ABOVE] TEST LOOKTHRU :
ITEMS "HOOK 12 IFT PR [
WITH A ROPE HANGING FRO
M IT]↵
 IF :HERE = 7 PR [YOU S
EE SOME DIMLY LIT STONE
 STEPS.] STOP↵
END↵

TO LOOK↵
 LOCAL "SEE? MAKE "SEE?
 0↵
 PR [YOU CAN SEE THE FO
LLOWING ITEMS:]↵
 LOOKAT :ITEMS :HERE↵
 IF :SEE? = 0 PR [NOTHI
NG]↵
 PR [ ]↵
END↵

TO I↵
 LOCAL "SEE? MAKE "SEE?
 0↵
 PR [YOU ARE CARRYING:]
↵
 LOOKAT :ITEMS ( - 1 )↵
 IF :SEE? = 0 PR [NOTHI
NG]↵
END↵

TO LOOKAT :LIST :N↵
 IF EMPTY? :LIST STOP↵
 IF LAST FIRST :LIST =
:N PR FIRST FIRST :LIST
 MAKE "SEE? 1↵
 LOOKAT BF :LIST :N↵
END↵
```

### Listing 4

APPLE LOGO II
LOGO SYSTEMS LOGO — IBM PC & PCjr

```
TO PUT.IT.HERE :N :OBJ
:LIST↵
IF MEMBERP :OBJ FIRST F
IRST :LIST [OP FPUT LPU
T :N BL FIRST :LIST BF
:LIST]↵
OP FPUT FIRST :LIST PUT
.IT.HERE :N :OBJ BF :LI
ST↵
END↵

TO PUT.IN :N :OBJ↵
MAKE "ITEMS PUT.IT.HERE
 :N :OBJ :ITEMS↵
END↵
```

TERRAPIN LOGO — C-64

```
TO PUT.IT.HERE :N :OBJ
:LIST↵
 IF MEMBER? :OBJ FIRST
FIRST :LIST OP FPUT LPU
T :N BL FIRST :LIST BF
:LIST↵
 OP FPUT FIRST :LIST PU
T.IT.HERE :N :OBJ BF :L
IST↵
END↵

TO PUT.IN :N :OBJ↵
 MAKE "ITEMS PUT.IT.HER
E :N :OBJ :ITEMS↵
END↵
```

### Listing 5

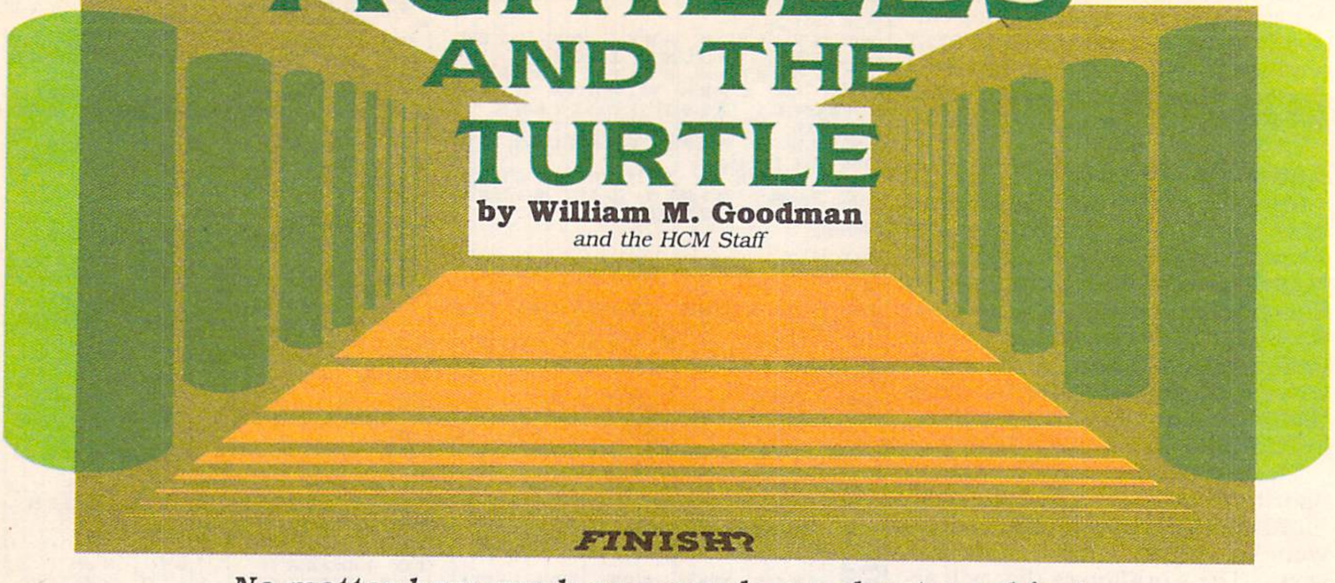
APPLE LOGO II
LOGO SYSTEMS LOGO — IBM PC & PCjr

```
TO U↵
TEST AND :HERE = 8 LOOK
THRU :ITEMS "HOOK 12↵
IFT [MAKE "HERE 12 STOP
]↵
MOVE 5↵
END↵

TO HAVE.IT? :OBJ↵
OP LOOKTHRU :ITEMS :OBJ
 -1↵
END↵

TO HERE? :OBJ↵
OP LOOKTHRU :ITEMS :OBJ
 :HERE↵
END↵

TO D↵
TEST LOOKTHRU :ITEMS "H
OOK 12↵
IFT [MAKE "HERE 8 STOP]
↵
MOVE 6↵
END↵

TO LOOKTHRU :LIST :OBJ
:N↵
IF EMPTYP :LIST [OP "FA
LSE]↵
IF :OBJ = "A [OP "FALSE
]↵
TEST AND MEMBERP :OBJ (
FIRST FIRST :LIST) (LAS
T FIRST :LIST) = :N↵
IFT [OP "TRUE]↵
IFF [OP LOOKTHRU BF :LI
ST :OBJ :N]↵
END↵
```

TERRAPIN LOGO — C-64

```
TO HAVE.IT? :OBJ↵
 OP LOOKTHRU :ITEMS :OB
J ( - 1 )↵
END↵

TO HERE? :OBJ↵
 OP LOOKTHRU :ITEMS :OB
J :HERE↵
END↵

TO D↵
 TEST LOOKTHRU :ITEMS "
HOOK 12↵
 IFT MAKE "HERE 8 STOP↵
 MOVE 6↵
END↵

TO U↵
 TEST ALLOF :HERE = 8 L
OOKTHRU :ITEMS "HOOK 12
↵
 IFT MAKE "HERE 12 STOP
↵
 MOVE 5↵
END↵

TO LOOKTHRU :LIST :OBJ
:N↵
 IF EMPTY? :LIST OP "FA
LSE↵
 IF :OBJ = "A OP "FALSE
↵
 TEST ALLOF ( MEMBER? :
OBJ FIRST FIRST :LIST )
 ( LAST FIRST :LIST = :
N )↵
 IFT OP "TRUE↵
 IFF OP LOOKTHRU BF :LI
ST :OBJ :N↵
END↵
```

# ACHILLES AND THE TURTLE



**by William M. Goodman**
*and the HCM Staff*

***No matter how much one may learn about a subject, one never knows everything about it. The LOGO turtle shows a Greek philosopher why models don't always lead to accurate conclusions.***

When one makes a model of something, even the best modeler has to fill in the gaps with assumptions, guesses, and just plain random choices. This is perfectly normal. But problems arise later if one assumes (incorrectly) that because a feature appears on the model, it is a part of the original object as well.

This article is based on an ancient model devised several centuries before Christ by the Greek philosopher Zeno. Zeno "proved" that *motion is impossible*—thus if we think that things can and do actually move, we are merely being deluded. Doubtless you are asking: "With what model could he possibly hope to support *that* view?!" As it turns out, he relied on several images; but the one to be used here takes the form of a race between the fleet-footed warrior Achilles and a tortoise (an ancestor of our beloved LOGO Turtle). This race, suggested Zeno, is a *good model* for motion; so if it turns out that the race envisioned cannot be completed, then neither can motion occur—despite all appearances. Unfortunately for Zeno, this is perhaps the first documented case where someone mistook the features of his model—imperfections and all—for real-live properties of the object it was designed to represent.

## Zeno's Model

Zeno's model was basically constructed like this: Give the turtle (or tortoise, if you prefer) a head start at place T1, and place Achilles behind him at A (see Figure 1). Then begin the race.

When Achilles reaches T1, the turtle will have advanced to T2. Then, once Achilles gets to T2, the turtle will have already made his way to T3. But when Achilles finally advances in turn to T3, the turtle will have by now arrived at T4 . . . and so on. In short (Zeno concluded), we are deluded if we think that Achilles can ever move to overtake the turtle, because this model *shows* that catching up is simply out of the question—the turtle will always be "a step ahead."

Now I don't know about you, but I for one found Zeno's conclusion to be a bit extreme! So, I decided to use LOGO to model the race between Achilles and the turtle, to see what *really* happens. This spawned the procedures called *RACE1*, controlled by FIRSTRACE. (For convenience, I will begin refering to Achilles as A., and to the turtle as T.)

The race begins when FIRSTRACE calls the procedure SHORTRACE, where the illustration of A. and T. running is produced by simple animation. Although there is a single sprite for each of A. and T. (Sprite 1 for A., Sprite 2 for T.), each sprite alternates in carrying one of the two distinct shapes for each runner. (See Figure 2 for the shapes and characters.) After each runner takes a "step," the procedure WINYET? checks to see whether A. has won yet. When Achilles finally *does* catch up and win, WINNER causes him to jump for joy while the turtle slinks away.

## Was Zeno Wrong?

"But if Achilles *wins*," you ask, "does this not *prove* that Zeno was wrong after all?" Not quite, I'm afraid. Look at some of the features of *Race1* that do not exactly apply to the "real" race:

1. The relative heights of Achilles and the turtle are not proportional to their real heights.
2. The LOGO colors of A. and T. are not the same as the actual colors (at least, this is unlikely).
3. The two-stage animation of their running styles is a very crude simulation of their actual running styles.
4. The relative speeds of the LOGO A. and T. are not proportional to their real speeds.
5. The running routine in SHORTRACE causes Achilles to move in "quantum leaps" of two or three "turtle-paces" at a time. Moreover, A. does this at moments when T. is motionless. But the "real" Achilles has to somehow "pass through" all of the distance that comprises each step, and must do this when the turtle is also in motion.

Of these distinctions between *Race1* and the real race, the first three are of the "harmless" variety—they make no significant difference in the accuracy of the model to the real race. The fourth distinction is potentially more significant: If the purpose of the model is to illustrate something about *motion*, is it not important how we choose to represent *speed*? Maybe yes, maybe no. This is just the sort of place where the objectivity of the model can get clouded.

It is the fifth distinction between Zeno's race and *Race1*, however, that is most critical in this example. Zeno is really asking: "how can a moving Achilles ever catch up to a moving turtle?" Yet our program always moves A. and T. *one at a time*—and always allows A. a leap of several turtle-paces *regardless of where the turtle is positioned*. So of course A. is bound to win! But this is hardly a model for the same race that Zeno himself had in mind. For him, even if the turtle is only one pace ahead of Achilles, it is still doubtful that A. can catch up to T., provided T. keeps moving. In short, when *Race1* mechanically adds a few extra steps to Achilles' position (hence letting him "get ahead"), it is not acting in a manner which *models* the race that Zeno described. Thus (and this is the key point), the winner of *Race1* does not necessarily correspond to who should have won in Zeno's race!

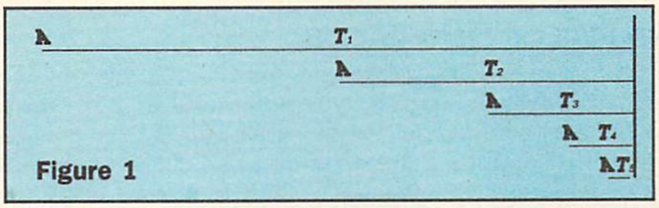


Figure 1

*"Now I don't know about you, but I for one found Zeno's conclusion to be a bit extreme!"*

## A LOGO Model For Zeno's Theory

A better model for Zeno's race is found in the *Race2* listing. To appreciate this program, imagine that the race between A. and T. is being filmed by a new type of "3-lens video camera." One lens focuses only on Achilles, while the second focuses only on the turtle (so that these runners appear on the screen always at the same size). The third lens focuses only on the *space between* A. and T. The upshot of this supposed "new technology" is that the space between A. and T. can now be blown up in scale without enlarging the images of these runners at the same time.

The sequence of *Race2* is as follows: First (using tiles) the procedure MARKERS causes a row of suitably-scaled flags to appear alongside the race path. As in *Race1*, A. and T. are given their starting positions, with T. having an initial head start. This time the A. sprite is given an initial speed of 8 in the :ASPEED variable.

GETSETGO first calls TURT to advance the turtle two steps with animation. The procedure ACHILLES then updates Achilles' speed and calls upon RACE to animate a step for A. WITHIN26? next checks to see whether A. has advanced to within 26 "screen-units" of catching T. ("Screen-units" are just a measure of the apparent distance between two points on the video screen image.) If Achilles is not yet that close to the turtle, GETSETGO starts over.

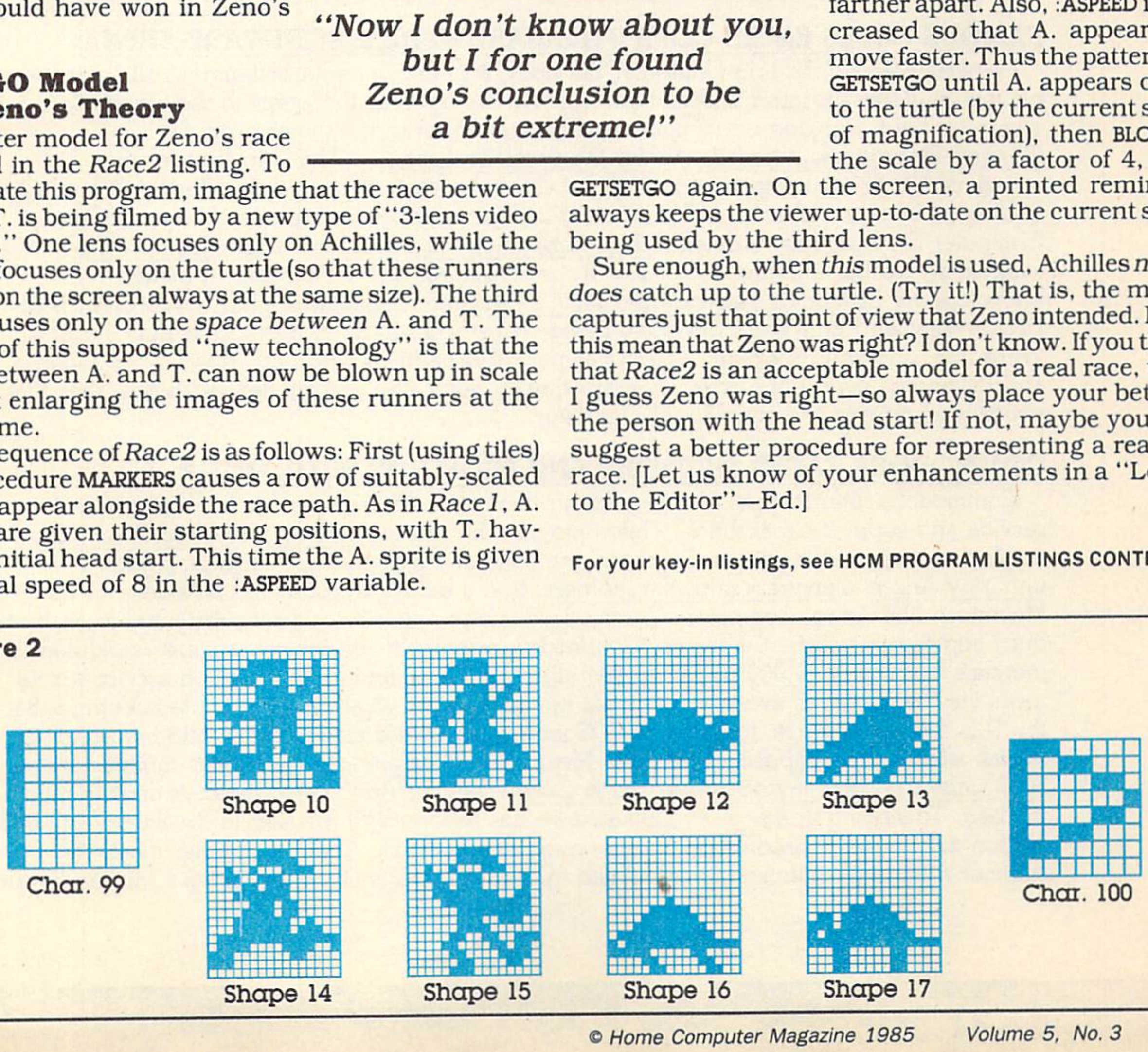
Once A. *does* come within 26 screen-units of the turtle, however, BLOWUP changes the *scale* of the "video image" of the imagined third lens by a factor of 4. Hence, suppose that the real Achilles and turtle are separated by, say, five feet, and that this distance is initially represented on the screen by a distance of 26 screen units. Then, after the procedure BLOWUP has been run, this same separation would now be represented by 26 x 4 = 104 screen units—which is only to say that the distance between A. and T. would appear to have been magnified four times.

To enhance this appearance of magnification, the procedure MARKERS is used again to redraw the flags along the race path so that they look farther apart. Also, :ASPEED is increased so that A. appears to move faster. Thus the pattern is: GETSETGO until A. appears close to the turtle (by the current scale of magnification), then BLOWUP the scale by a factor of 4, and GETSETGO again. On the screen, a printed reminder always keeps the viewer up-to-date on the current scale being used by the third lens.

Sure enough, when *this* model is used, Achilles *never does* catch up to the turtle. (Try it!) That is, the model captures just that point of view that Zeno intended. Does this mean that Zeno was right? I don't know. If you think that *Race2* is an acceptable model for a real race, then I guess Zeno was right—so always place your bets on the person with the head start! If not, maybe you can suggest a better procedure for representing a real-life race. [Let us know of your enhancements in a "Letter to the Editor"—Ed.] **HCM**

**For your key-in listings, see HCM PROGRAM LISTINGS CONTENTS.**

**Figure 2**



# INDUSTRY WATCH

### SURPLUS TI-99/4A PARTS APPEARING IN RADIO SHACKS

Tandy seems to have snapped up a good supply of surplus parts for the TI-99/4A—parts that were sold when TI left the home computer market. Available now through the Radio Shack catalog are video modulators with a cable assembly (just $4.95), power supply boards ($4.95), and genuine tan 99/4A keyboards (only $2.95). Imagine replacing half a computer for only $13 . . .

### THE FALL AND RISE OF IBM'S JUNIOR

Although the dust around IBM's March announcement to discontinue production of its PCjr has settled, speculation over the impact of its withdrawal from the market continues to stir things up. Many industry analysts believe that IBM pulled PCjr to make way for the introduction of a "PC II" and the corresponding price cuts on the existing line that would effectively bring the price of the current PC closer to Junior's original price. The PCjr departure also leaves Apple relatively free to clinch its hold on the school market, but it may benefit only temporarily in the home market. Once PCjr production stops, IBM may offer it at fire-sale prices to clear out its existing inventories (reportedly to total another 150,000-300,000 units), thus undercutting the Apple II Family computers. These price cuts (and IBM's promise to fully support the machine with software and parts) are good news for PCjr's installed and potential user-base. In addition, several third-party peripheral vendors have stated their intentions to feed off the machine for at least another year. One firm, Racore Corp., has even introduced its "ATcessory," adding a 1.2 megabyte floppy disk drive to the PCjr, and a local area network that is said to link up to 16 PCjrs, PCs, XTs, and ATs. Smart money is betting on Big Blue re-entering the home market with a lower priced machine in fourth quarter.

### FAMOUS FACES ENTER COMPUTER HARDWARE & SOFTWARE ARENA

General Electric, Boeing Computer Services, and Encylopedia Britannica, all heavyweight players in electronics and information, are now taking their first steps in the personal computer field. Banking on its name and reputation for quality in appliances, General Electric has decided to enter the peripheral market. The company recently introduced a thermal printer, a modem, and two monitors. Rather than offering the industry's traditional 90-day warranty, GE is offering 2-year warranties. In another arena, Boeing Computer Services has released a three-dimensional spreadsheet called *Boeing Calc* that is supposedly similar to Lotus' *1-2-3*. It will run on micros and mainframes. Finally, Encyclopedia Britannica has acquired MSA's Educational Business Unit, which consists of DesignWare and Edu-Ware Services. DesignWare will sell its own products plus the Edu-Ware line, and will reportedly operate as part of Britannica Learning. Now, if GE, Boeing, and Britannica ever join forces, we may actually get to see a computerized dishwasher that writes term papers and runs *Flight Simulator*.

### COMMODORE FACES TRIUMPHS AND PROBLEMS WITH AMIGA

Commodore International Ltd. is working hard at customer relations by expanding its service and support capabilities, while enjoying the unexpected success of its Commodore 16 in Mexico. Behind the scenes, however, the company is reportedly struggling with unhappy Amiga managers and shareholders, and a delayed production schedule for its Macintosh-like Amiga computer. Commodore is also expected to post substantial losses in its third and fourth quarters. Even so, Commodore's regional service centers are scheduled to increase from 650 to 1,000 by May 1. A toll-free hotline, and an information service for its users via CompuServe are also being set up. In Mexico, where the market lags behind that of the U.S. by 2 to 3 years, the little $170 Commodore 16 has made its way into thousands of homes after having flopped in the U.S. Meanwhile, Commodore's stock has tumbled by about 50% since the company acquired Amiga Corp., leaving Amiga shareholders angrily holding the bag. To smooth things over, Commodore has reportedly promised to award more than $7 million in bonuses if production on the crucial Commodore Amiga computer meets its summer schedule—a timetable calculated to offset the parent's cash problems with early sales.

## TO ST, OR NOT TO ST—THAT IS THE QUESTION

While Atari scrambles to raise enough cash to remain solvent and develop new products, decisions are being made regarding what may be its saving grace—its ST series of computers. Two underlying operating systems are currently being tested in ST prototypes. The more sophisticated of the systems would allow GEM, part of the overall operating system, to take advantage of the system's capability to set up hierarchical files on a hard disk, and provide color. The ST line will imitate Apple's Macintosh (which has no color), but will be marketed at prices low enough to cause both IBM and Apple some concern in the consumer market. If the ST series can, in fact, be shipped in sufficient quantity this year—without any major product flaws—it stands a good chance of keeping the phoenix-like Atari Corp. flying high. In a recent development, however, Atari's announced withdrawal from exhibiting at June CES casts serious doubt upon their ability to deliver.

## PRICE WARS HEAT UP AGAIN ON NEW FRONTS

The introduction of Atari's and Commodore's new machines this summer is expected to ignite another price war on the computer battleground—but others have already fired the first shots. Desk-organizer programs, which entice users with resident utilities such as appointment calendars, notepads, calculators, etc., have recently experienced cuts ranging from 30 to 50 percent. Prices are hovering in the $50 to $100 range, and have not yet stabilized. Dot-matrix and daisy-wheel printers also have been rapidly tumbling in price, while increasing in performance and features. Keyboard macros are also following suit: Software Research Technologies has declared war on Borland International by dropping the price of its keyboard-redefinition program, *Smartkey*, to $49. *Superkey*, Borland's similar program, retails for $69. The only sure victor of these price wars will be the consumer.

## A TURNING POINT FOR BIG RED?

Apple Computer has been experiencing a number of growing pains lately with the departure of key engineers and executives, poor morale in its Apple *II* division, and new-product worries. As Apple management began operating more and more like a Fortune 500 company, it opted to wedge its "favorite fruit," the Macintosh, into similar corporate environments—while paying less and less attention to its "cash cow" Apple *II* Family development. As a result, many disgruntled engineers and middle managers from the less-glamorous Apple *II* division, feeling slighted and ignored, left for greener pastures elsewhere. Since co-founder Steven Wozniak sold more than 3 million shares of his Apple stock and joined the exodus, morale reportedly eroded even further. In addition, Apple's long-awaited flat-panel LCD screen for the *II*c was universally panned for its small, hard-to-read display. In spite of the problems, Apple has released an enhancement kit for the *II*e that provides (among other things) full compatibility with the *II*c, improved mouse response, and the ability to enter BASIC commands in upper and lower case. Big Red is also said to be working on a "Turbo Mac," expected to support 2 megabytes of RAM; and, by 1986, a next-generation, 16-bit Apple *II* machine.

## RUMORS SUPPORT NOTION ABOUT TI-99/8 CLONE

Well-substantiated rumors persist that a TI-99/8 "clone" is being beta-tested now, and that it will make an appearance at the Summer Consumer Electronics Show in Chicago. The East-Coast manufacturer is said to be negotiating with Texas Instruments to use some of the 99/4A's proprietary operating system. The new machine reportedly uses the 9995 chip, and will come with 128K RAM expandable to 512K. It will have 16K VDP RAM expandable to 64K. All expansion will be through the TI Peripheral Expansion Box. The video display is expected to support composite and RGB monitors, and may offer a variety of high-resolution options, including 80 columns. Look for a $300 price range for a minimum configuration that can run existing 99/4A software and make use of existing peripherals.

# Apple Seedlings

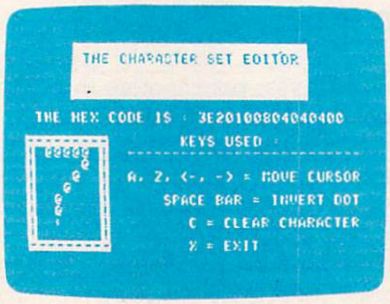
## Character Graphics on the Hi-Res Screen

**by Steve Cordon**
*and the HCM Staff*

***Attention Apple programmers looking for character graphics on the hi-res screen. Look no further—this is it!***

Although Applesoft BASIC contains several graphics commands, it lacks provisions for handling character-level graphics. Because using shape tables is a cumbersome method of defining characters for the high-resolution (hi-res) screen, I decided to write my own machine-language character-graphics routine. Without any hardware modifications, this routine will draw any user-defined characters on the hi-res screen via easy-to-use CALLs from BASIC. Plus, you may define characters via an easy-to- use *Character Graphics Editor* program.

The Character Graphics System fits in 537 bytes. The program itself requires about 281 bytes, leaving 256 bytes for character data. The routine will handle up to 32 different user-defined characters at one time, numbered from 0-31. Two listings accompany this article: The *Character Graphics Demonstration* program is a short example of how characters are defined and displayed on the hi-res screen. The *Character Graphics Editor* program aids you in determining the codes needed to define your own characters. You can add the Character Graphics System to your own programs by copying the machine-language loader subroutine of the *Character Graphics Demonstration* program (lines 370-460) into your routines.



```
     THE CHARACTER SET EDITOR

THE HEX CODE IS : 3E20100804040400
              KEYS USED :
        ---------------------------
        A, Z, <-, -> = MOVE CURSOR
          SPACE BAR = INVERT DOT
            C = CLEAR CHARACTER
            % = EXIT
```

***This picture of the Editor screen demonstrates how the code for the number 7 was obtained for the Demonstration program.***

### HI-Res Protection and Program Placement

The Character Graphics System takes advantage of the fact that programs accessing the hi-res screen often must be relocated above the screen memory area. [See the "Home Computer Tech Note" for Apple in *HCM* Vol. 4, No. 1 for details on this—Ed.] Consequently, *all* programs using this system must be relocated. This is easily accomplished with the following lines at the beginning of the program:

```
200 IF PEEK (104) = 64 THEN 230
210 POKE 104,64:POKE 103,1:POKE 16384,0
220 PRINT CHR$ (4);"RUN file.name'
230 . . . main program begins here
```

The disk file name of the program is *file.name*. The *Demonstration* listing uses these exact lines. Because this program relocates itself, always SAVE modifications before you RUN.

Before you begin to learn the details of the Character Graphics system, take a few minutes to RUN the *Character Graphics Demonstration* program to see how quickly a large number of character shapes may be displayed and edited. First, a cautionary note to anyone keying-in this program: If you make an error in the DATA statements and then RUN the program—you may need to reboot the system and reload your program to do your debugging, because this program manipulates certain pointers used by DOS (both 3.3 and ProDOS). Because this program will attempt to reload itself, you should always save any modifications you have made before running.

The first half of the program displays 270 small stick figures—the rapidly changing characters create an animated-movement effect. If this were done using shape tables and the DRAW and XDRAW commands, the graphic changes wouldn't be nearly as fast; plus, there would be a "flickering" effect that so often accompanies Applesoft BASIC animation. Because the Character Graphics System *replaces* an entire 64-pixel (picture element) block of the hi-res screen at one time, characters can be changed with no noticeable flicker.

The second half of the *Demonstration* creates a labeled bar graph and quickly displays random numbers. These results are caused by a unique redirection of the PRINT statement to actually PRINT to the hi-res screen.

### Defining Characters for Hi-Res

The Apple hi-res screen is controlled by an area of the Apple's memory called the hi-res page. [For details about the hi-res page, see the "Home Computer Tech Note" for Apple in *HCM* Vol 5, No. 1—Ed.] Pixels are turned "on" by setting bits in this memory area to 1, and turned "off" by clearing bits to 0. The screen is 280 pixels wide by 192 high (or 160 high in "mixed-mode").

In hi-res memory, 7 bits of each 8-bit byte represent 7 adjacent pixels on the screen. The eighth bit of each byte is used to control the color of the bits turned on in the byte. Including all of the details of this system would be beyond the scope of this article, but you must be aware that I've made the characters 7 pixels wide by 8 characters high to keep each each of the 8 rows of the hi-res character within a single byte of hi-res memory.

Before you can place graphic characters on the hi-res screen, they must be defined. The Character Graphics System creates a CALL CHAR command for just this purpose. Here's the format for that command:

CALL CHAR,*Charnum*,*String*

CHAR is actually a variable defined in the BASIC routine that LOADs the machine-language section (lines 370-460) and represents the address being CALLed. *Charnum* is the number (0-31) of the character being defined, and *String* is a hexadecimal code used to define the character. (See lines 250-290 of the *Demonstration* program for examples.) The way to determine the hexadecimal codes for a character is to use the *Character Graphics Editor* program.

To create characters, first RUN the *Editor*. You will be presented with a larger-than-actual-size space in which to design a character; the cursor will be flashing inside the space. You can move the cursor right and left using the arrow keys, up with the A key, and down with the Z key. The @ symbol represents a turned-on pixel in the character, and a blank represents a turned-off pixel. To change the state of a pixel, just press the (SPACE BAR). As you turn on pixels, notice that the hexadecimal code for your character changes.

When you are satisfied that the character is complete, copy down the hexadecimal code so that you can use it in a CALL CHAR command in a program that uses this character graphics routine. To exit the program, type X. To clear the character-defining area, type C. Remember: **You must copy and execute the code from lines 370-460 of the *Character Graphics Demonstration* program in any program where you want to use the Character Graphics System.** These lines could be located anywhere in the new program, and may be given different (but sequential) line numbers—but they *must* be executed before you use any of the "commands" defined by this article.

## PRINTing and HCHARing

Once you define your characters, the Character Graphics System uses two different commands to put them on the screen. The CALL HCHAR command places a specified number of one of the characters on the screen, beginning at a certain screen location. Here's the format:

CALL HCHAR,*vpos*,*hpos*,*charnum*,*num*,*code*

The *vpos* and *hpos* values are very similar to VTAB and HTAB values on the text screen, just one less (i.e., the upper-left corner is 0,0 instead of 1,1). The *charnum* value is a number from 0-31 identifying the character (previously defined in a CHAR command) that is to be displayed. The *num* value can range from 1 to 255. It determines how many of the specific character will be displayed repeatedly across the screen. If the routine reaches the edge of the screen, it will wrap to the next line.

The *code* may be between 0 and 3. It determines how the character will be displayed. A zero stores the character in the screen memory, erasing what used to be there—this is by far the most often-used *code*. The *codes* 1, 2, and 3 perform a logical AND, exclusive OR, and OR respectively, between the existing memory value and the character specified in the CALL HCHAR statement. These three techniques allow you to "merge" different characters on the screen with the existing background. If you use *code* 1 (AND), then the only pixels that will remain "on" after execution are those that were already "on" before you did the CALL HCHAR command. With *code* 2 (exclusive OR), only pixels that were in a different state than those in the character will be "on" after the CALL HCHAR has been executed. Using *code* 3 will effectively *add* the character specified to whatever is already on the screen. It will turn on pixels where bits are set to 1 in the character, but it will not turn off any that are already "on." (*Wild Kingdom* in *HCM* Vol. 4, No. 3 used this technique.)

The CALL PRNT command redirects all regular Applesoft PRINT statements so that characters are displayed on the hi-res screen. This is accomplished by changing certain zero-page pointers used by the DOS routines so that they control output of the PRINT statement. The routine even determines whether DOS 3.3 or ProDOS is being used and changes the appropriate pointers to accommodate the current operating system. The CALL PRNT command is used in the second half of the *Demonstration* program, when the numbers are PRINTed to the hi-res screen. The CALL PRNT statement in line 300 has changed the pointers so that the PRINT commands on lines 340 and 350 display our redefined characters on the hi-res screen. The CALL OFF statement in line 360 changes the pointers back, so that future PRINT statements will once again place characters only on the text screen.

In order to PRINT characters using this system, they must all be defined using a CALL CHAR command—as they are in lines 270-290 in the *Demonstration* program. You cannot simply PRINT a character on the hi-res screen unless you've defined it in a CALL CHAR statement. Then, use Figure 1 to determine which shape number you will use in your PRINT statement. In our program, we have chosen to define the numbers as the same characters used in the PRINT statement—i.e., the number 1 is shape number 17. Had we not defined shape 17 in line 280, or if we'd used a character other than one of those shown below, then the shape of the character displayed on the hi-res screen would be unpredictable—it would be just a random sample of memory.

**FIGURE 1** **CALL PR Shape Numbers**

| Shape # | Character | Shape # | Character |
|---|---|---|---|
| 0 | [Space] | 16 | 0 |
| 1 | ! | 17 | 1 |
| 2 | ' | 18 | 2 |
| 3 | # | 19 | 3 |
| 4 | $ | 20 | 4 |
| 5 | % | 21 | 5 |
| 6 | & | 22 | 6 |
| 7 | ' | 23 | 7 |
| 8 | ( | 24 | 8 |
| 9 | ) | 25 | 9 |
| 10 | * | 26 | : |
| 11 | + | 27 | ; |
| 12 | , | 28 | < |
| 13 | - | 29 | = |
| 14 | . | 30 | > |
| 15 | / | 31 | ? |

If you wish to see more examples of ways that this routine can make your programs more exciting graphically, here's a list of the *HCM* programs for the Apple II Family that have used this Character Graphics System:

| Program | Issue |
|---|---|
| Wild Kingdom | Vol. 4, No. 3 |
| Stadium Jumping | Vol. 4, No. 4 |
| Orbital Defender | Vol. 5, No. 1 |
| Evacu-Pod | Vol. 5, No. 2 |
| Over-Reaction | Vol. 5, No. 3 |

For your key-in listings, see HCM PROGRAM LISTINGS Contents.

HCM

# Commodore Hornblower

## Waveforms and Envelopes

*Want to know what a particular waveform and envelope sound like on your C-64? There's no PEEKing and POKEing necessary with this easy-to-use program.*

by Roger Wood
*HCM Staff*

In the previous issue, we introduced the Sound Interface Device (SID) chip of the Commodore 64. We also included a short routine which mathematically determined the proper POKE values for 8 octaves of frequencies, placed the values in easy-to-access arrays, and provided a simple keyboard routine for playing some tunes. This time we'll use a similar program to allow you to quickly and easily determine the sound of any of the SID's waveforms—including a selectable pulse width. You will also be able to adjust the envelope generator of the SID chip and hear the effect that it has on your selected waveform. (For a detailed discussion of waveforms and envelopes, see "Commodore Hornblower" in *HCM* Vol. 5, No. 2.)

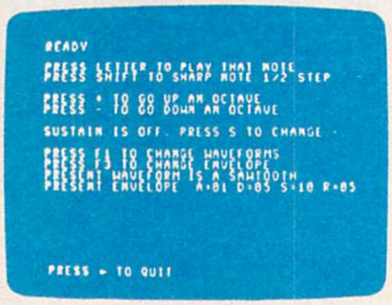
### Using the Program

When you RUN the *Wave/Envelope Select* program, you are presented with a descriptive list of the program's options and functions (see screen picture). The keyboard input routine is quite similar to last issue's program: just press a letter to play a particular note. Change octaves by pressing either the + to go up an octave, or – to go down. However, there are a few enhancements: To play a note one-half step higher, press shifted letters—e.g., a shifted A plays an A#. Also, at any time in Play mode, you may press the S key to activate Sustain. As long as Sustain is "on," as indicated on the screen, the note continues to sound until you either press another note or turn off Sustain.

The major way this program differs from last time is that, in addition to Play mode, you will also have selection options available. You can now select the waveform and the envelope you wish to use. To change waveforms, first press (F1). A message will appear on the lower part of the screen informing you that you may either:

PRESS (SPACE BAR) TO CHANGE WAVEFORM
PRESS (RETURN) TO GO TO PLAY MODE

By leaving Sustain on and simply pressing the (SPACE BAR), you will see and hear the various waveforms, one after the other. The only exception is the pulse wave: Each time it is selected you will be prompted for a value from 01 to 99 to determine the pulse width or "duty cycle" of the pulse. The duty cycle is the percentage of the length of time that the pulse is "on" compared to when it's "off." As examples, the square pulse in Figure 1 is a 50 percent duty cycle and the narrow pulse is about a 16 percent duty cycle. Try experimenting with this option to see what effect pulse width has on sound. Note that you are prompted only once for the pulse width each time that you select the pulse wave.



```
READY

PRESS LETTER TO PLAY THAT NOTE
PRESS SHIFT TO SHARP NOTE 1/2 STEP

PRESS + TO GO UP AN OCTAVE
PRESS - TO GO DOWN AN OCTAVE

SUSTAIN IS OFF. PRESS S TO CHANGE

PRESS F1 TO CHANGE WAVEFORMS
PRESS F3 TO CHANGE ENVELOPE
PRESENT WAVEFORM IS A SAWTOOTH
PRESENT ENVELOPE  A=01 D=05 S=10 R=05




PRESS ← TO QUIT
```

### Envelope Control

If you press (F3) from Play mode, you can alter the envelope. By pressing either A (Attack), D (Decay), S (Sustain), or R (Release), you select the part of the envelope that you want to affect. As soon as you make your selection, the cursor appears over the appropriate value to be changed—press (RETURN) to accept the value when you are satisfied. The values here range from 01 to 15, and are the actual values that the SID chip will use to determine the time (for A, D and R) or volume level (for S). A lower value will shorten the time interval or lower the level of the parameter; higher values lengthen the time or raise the sustain level.

Stay tuned in future issues as we to continue our look at the SID chip and sound out its capabilities. We'll play with filters, and take a look at the interaction of several waveforms in the next "Commodore Hornblower."

**Wave/Envelope Select (C-64)**
**Explanation of the Program**

| Line Nos. | |
|---|---|
| 100-180 | Program header. |
| 190-250 | Program initializtion. |
| 260-480 | Main play routine. |
| 490-940 | Change waveform and envelope. |
| 950-1160 | I/O subroutines. |
| 1170-1310 | Initialization subroutine. |

Figure 1

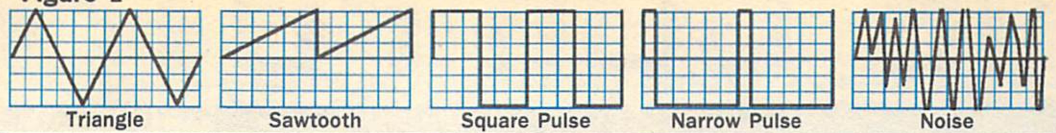
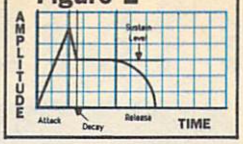


Figure 2



```
100 REM ***********************
110 REM * WAVE/ENVELOPE SELECT *
120 REM ***********************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO
150 REM BY ROGER WOOD
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 5.3.1
180 REM COMMODORE 64 BASIC
190 POKE 53280,15:POKE 53281,0
200 PRINT "{SHIFT CLR}{CTRL WHT}{2CRSRDOWN}  PREPARING SID POKE VALUES":PRINT "{CRSRDOWN}  PLEASE WAIT":GOSUB1170
210 FOR I=S TO VL(0):POKE I,0:NEXT
220 VL(1)=15:POKE VL(0),VL(1):OC=4
230 AD(1)=21:SR(1)=165:AK$="01":DC$="05":SN$="10":RL$="05"
240 CR(1)=32:WV$="SAWTOOTH":GOSUB780:GOSUB750:POKE CR(0),CR(1)
250 PL(1)=FN L(VAL(PW$)):PH(1)=FN H(VAL(PW$)):POKE PL(0),PL(1):POKE PH(0),PH(1)
260 IF PEEK(653)=0 THEN SH=0:GOTO280
270 SH=1
280 K=PEEK(197):IF K=64 THEN260
290 NT=(-1)*((K=20)+(3*(K=18))+(5*(K=14))+(6*(K=21))):IF NT THEN420
300 NT=(-1)*((8*(K=26))+(10*(K=10))+(12*(K=28))):IF NT THEN420
310 IF K=40 THEN OC=OC+1:IF OC>7 THEN OC=7
320 IF K=43 THEN OC=OC-1:IF OC<0 THEN OC=0
330 IF K=40 OR K=43 THEN FOR DE=1 TO 50:NEXT:IF SU THEN420:GOTO260
340 IFK<>13 THEN380
350 FOR DE=1 TO 50:NEXT
360 SU=NOT(SU):IF SU THEN R=8:C=13:GOSUB1020:PRINT"{CTRL RVSON}ON {CTRL RVSOFF}":GOTO260
370 R=8:C=13:GOSUB1020:PRINT"{CTRL RVSOFF}OFF":GOTO470
380 IF K=4 THEN GOSUB490:GOTO260
390 IF K=5 THEN GOSUB580:GOTO260
400 IF K=57 THEN POKEVL(0),0:POKE198,0:PRINT"{SHIFT CLR}BYE BYE. SEE YOU NEXT TIME":END
410 GOTO260
420 NT=NT+SH:IF NT=13 THEN NT=1
430 POKE FL,NT%(NT,OC,0):POKE FH,NT%(NT,OC,1)
440 CR(1)=CR(1)OR1:GOSUB1010
450 K1=PEEK(197):IF K1=K THEN450
460 IF SU THEN260
470 CR(1)=CR(1)AND 254:GOSUB1010
480 GOTO260
490 GOSUB1050:GOSUB1020:PRINT " PRESS [SPACE BAR] TO CHANGE WAVEFORM"
500 PRINT " PRESS [RETURN] TO GO TO PLAY MODE"
510 IF PEEK(197)=1 THEN GOSUB1050:RETURN
520 IF PEEK (197)<>60 THEN510
530 IF CR(1) AND 16 THEN CR(1)=(CR(1)AND1)OR32:W=2:GOTO570
540 IF CR(1) AND 32 THEN CR(1)=(CR(1)AND1)OR 64:W=3:GOTO570
550 IF CR(1) AND 64 THEN CR(1)=(CR(1)AND1)OR 128:W=4:GOTO570
560 IF CR(1) AND 128 THEN CR(1)=(CR(1)AND1)OR 16:W=1
570 GOSUB1010:GOSUB880:GOSUB1010:GOTO490
580 GOSUB1050:GOSUB1020:PRINT " PRESENT  ATTACK=";AK$;" DECAY=";DC$
590 PRINT " PRESENT SUSTAIN=";SN$;" RELEASE=";RL$
600 PRINT " PRESS A, D, S, OR R TO CHANGE VALUE":PRINT " OR [RETURN] TO ACCEPT"
610 CH=PEEK(197):IF CH=64 THEN610
620 IF CH=1 THEN GOSUB1050:RETURN
630 CG=-1*((CH=10)+(CH=18)*2+(CH=13)*3+(CH=17)*4)
640 ON CG+1 GOTO610,650,650,670,670
650 RT=16:IF CG=1 THEN CT=16:TP$=AK$:GOTO690
660 CT=25:TP$=DC$:GOTO690
670 RT=17:IF CG=3 THEN CT=17:TP$=SN$:GOTO690
680 CT=28:TP$=RL$
690 R=RT:C=CT:GOSUB1020:P=1:POKE 198,0:LO$="00":HI$="15":GOSUB1060
700 IF CG=1 THEN AK$=TP$:AD(1)=(AD(1)AND15)+(16*VAL(AK$))
710 IF CG=2 THEN DC$=TP$:AD(1)=(AD(1)AND240)+VAL(DC$)
720 IF CG=3 THEN SN$=TP$:SR(1)=(SR(1)AND15)+(16*VAL(SN$))
730 IF CG=4 THEN RL$=TP$:SR(1)=(SR(1)AND240)+VAL(RL$)
740 GOSUB750:GOSUB760:GOTO580
750 POKE AD(0),AD(1):POKE SR(0),SR(1):RETURN
760 R=13:C=0:GOSUB1020
770 PRINT "  PRESENT ENVELOPE: A=";AK$;"  D=";DC$;"  S=";SN$;"  R=";RL$:RETURN
780 PRINT "{SHIFT CLR}  READY"
790 PRINT "{CRSRDOWN}  PRESS LETTER TO PLAY THAT NOTE"
800 PRINT "  PRESS SHIFT TO SHARP NOTE 1/2 STEP"
810 PRINT "{CRSRDOWN}  PRESS + TO GO UP AN OCTAVE"
820 PRINT "  PRESS - TO GO DOWN AN OCTAVE"
830 PRINT "{CRSRDOWN}  SUSTAIN IS OFF. PRESS S TO CHANGE."
840 PRINT "{CRSRDOWN}  PRESS F1 TO CHANGE WAVEFORMS"
850 PRINT "  PRESS F3 TO CHANGE ENVELOPE"
860 GOSUB880:GOSUB760
870 PRINT "{7CRSRDOWN}  PRESS - TO QUIT":RETURN
880 IF W=1 THEN WV$="TRIANGLE  ":GOTO920
890 IF W=2 THEN WV$="SAWTOOTH  ":GOTO920
900 IF W=3 THEN WV$=PW$+"% PULSE  ":GOTO920
910 IF W=4 THEN WV$="NOISE     "
920 R=11:C=0:GOSUB1020:PRINT "{CRSRDOWN}  PRESENT WAVEFORM IS A ";WV$
930 IF W=3 THEN GOSUB950:POKE PL(0),PL(1):POKE PH(0),PH(1)
940 RETURN
950 P=1:TP$=PW$:POKE 198,0
960 GOSUB1050:GOSUB1020:PRINT " SELECT PULSE WIDTH OR PRESS [RETURN]"
970 PRINT " TO ACCEPT SHOWN VALUE.":PRINT" INPUT NEW PULSE WIDTH: ";TP$;"%";
980 R=18:C=24:GOSUB1020:RT=R:CT=C:LO$="01":HI$="99":GOSUB1060
990 R=12:C=24:GOSUB1020:PRINT TP$;
1000 PL(1)=FN L(VAL(TP$)):PH(1)=FN H(VAL(TP$)):PW$=TP$:RETURN
1010 POKE CR(0),CR(1):RETURN
1020 POKE 781,R:POKE 782,C:POKE 783,.:SYS 65520:RETURN
1030 POKE 204,0:POKE 207,0:GET K$:IF K$="" THEN1030
1040 RETURN
1050 R=16:C=0:GOSUB1020:PRINT CL$:PRINT CL$:PRINT CL$:PRINT CL$:RETURN
1060 GOSUB1030:POKE 204,1:POKE 207,1
1070 IF K$<>CR$ THEN1090
1080 GOSUB1050:RETURN
1090 IF K$<"0" OR K$>"9" THEN1060
1100 IF P=1 THEN TP$=K$+RIGHT$(TP$,1):P=0:GOTO1120
1110 IF P=0 THEN TP$=LEFT$(TP$,1)+K$:P=1
1120 IF VAL(TP$)>VAL(HI$)THEN TP$=HI$
1130 IF VAL(TP$)<VAL(LO$)THEN TP$=LO$
1140 C=CT:GOSUB1020:PRINT TP$;
1150 IF P=1 THEN C=CT:GOSUB1020:GOTO1060
1160 C=CT+1:GOSUB1020:GOTO1060
1170 DIM PL(1):DIM PH(1):DIM CR(1)
1180 DIM AD(1):DIM SR(1):DIM VL(1):PW$="50":CR$=CHR$(13)
1190 S=54272:FL=S:FH=S+1:PL(0)=S+2:PH(0)=S+3
1200 CR(0)=S+4:AD(0)=S+5:SR(0)=S+6:VL(0)=S+24:SU=0
1210 CL$="                                 "
1220 DEF FN PF(FR)=FR/.06096:DEF FN FR(BA)=BA*(1.05946309)
1230 DEF FN HF(FR)=INT(FR/256):DEF FN LF(FR)=INT(FR-(256*(INT(FR/256))))
1240 DIM NT%(12,7,2)
1250 FQ=16.3515977:FOR OC=0 TO 7:FOR NT=1 TO 12
1260 NT%(NT,OC,0)=FN LF(FN PF(FQ))
1270 NT%(NT,OC,1)=FN HF(FN PF(FQ))
1280 FQ=FN FR(FQ):NEXT NT:NEXT OC
1290 DEF FN H(V)=INT(V*4096/25600)
1300 DEF FN L(V)=INT(V*4096/100-(256*INT(V*4096/25600)))
1310 RETURN
```

HOME COMPUTER

## Key-In Checking

Errors are inevitable when typing-in programs, so here is an Applesoft routine that should help pinpoint one common error. Home Computer Magazine's programs always begin at line 100 and continue in increments of 10. But if a line number is mistyped, not only is the wrong line placed in the wrong place, but unknown previous lines can be erased. This utility can help eliminate this problem.

Lines 3 through 9 create a text file called LINE.CHECK, and lines 10 through 90 compose its contents. You will be able to merge this text file (via the **EXEC** command) with your keyed-in programs, and it will tell you which lines are missing, or which ones have line numbers that are not a proper increment of 10. To create this file, just key-in this Applesoft program. Then place a disk in the drive and type **RUN**. The disk-drive light will come on and the LINE.CHECK file will be placed on your disk.

```
 1 REM  COPYRIGHT 1985
 2 REM  EMERALD VALLEY PUBLISHING CO
 3 PRINT  CHR$ (4)"OPEN LINE.CHECK"
 4 PRINT  CHR$ (4)"WRITE LINE.CHECK"
 5 POKE 33,33
 6 LIST 10 -
 7 PRINT  CHR$ (4)"CLOSE"
 8 TEXT
 9 END
10 DEF  FN PQ(X) =  PEEK (X) + 256 *
    PEEK (X + 1):PTR =  FN PQ(103)
20 IF  FN PQ(PTR + 2) < LN + 10 THEN
    PRINT "EXTRA LINE NUMBER" FN PQ(PTR
    + 2): GOTO 90
30 IF  FN PQ(PTR + 2) > LN + 10 THEN
    PRINT "MISSING LINE NUMBER "LN + 10
   :LN = LN + 10: ON  FN PQ(PTR) > 0 G
   OTO 30: END
40 LN = LN + 10
50 :
60 :
70 :
80 :
90 PTR =  FN PQ(PTR): ON  FN PQ(PTR) >
    0 GOTO 20: END
```

Now, after you've keyed in a program and **SAVE**d it to disk, place the disk that contains your LINE.CHECK text file into the drive. Next, type **EXEC LINE.CHECK**, to merge lines 10 through 90 of the above program onto the front of your newly keyed-in program. Now type **RUN** and press **[RETURN]**. This will **RUN** the LINE.CHECK part of the program (located before line 100); as it discovers extra and missing lines in your typed-in program, they will list to the screen. This procedure won't **RUN** your keyed-in program, because it **END**s before the computer reaches line 100. It is important that you only use this program with a listing that starts at line 100 and continues by increments of 10—as do all listings in Home Computer Magazine. If you use a program that starts at line 200, for example, LINE.CHECK will inform you that lines 100-190 are missing. Also, before you **EXEC LINE.CHECK** you should **LIST** the program to be sure no lines have been mistakenly entered before 100, as they could interfere with the LINE.CHECK routine and cause it to act erratically.

This program works by first defining a **FuN**ction in line 10 that **PEEK**s the memory location of the next line in memory. It uses this **FuN**ction to continually get the next line number in the program, checking to see whether it is in the proper sequence. If it finds an increment of less than 10 between lines, then line 20 **PRINT**s **EXTRA LINE NUMBER**, followed by the extra line's number. If the increment is greater than 10, line 30 **PRINT**s **MISSING LINE NUMBER**, followed by the missing line's number. If neither of these error conditions exist, then the proper increment of 10 must exist. Line 40 increments the line counter (**LN**) by 10. Line 90 then determines whether another line exists in the program. If it does, then control returns to line 20 to check the next line. If line 90 discovers that the last line number has been reached, program execution halts.

After you've **RUN** this program and before **SAVE**ing your key-in, be sure to immediately use a **DEL 10,90** command to **DEL**ete the LINE.CHECK routine from your program in memory, so your key-in program will not have this extra program tacked onto the front of it.

*—Ellis McDaniels*

# TECH NOTES

## C64 Merging Programs from Disk

A fast and accurate way of developing programs is to simply merge a number of "tried-and-true" subroutines together. When integrating a DeBug from an ON DISK program with an older version of a program, the ability to merge them is essential. Unfortunately, the Commodore 64 doesn't provide a merge option. [One can buy a utility program such as Sysres, reviewed elsewhere in this magazine, to facilitate the effort—Ed.] A merge routine was printed in the "Home Computer Tech Note" for Commodore in HCM Vol. 5, No. 1—but it was rather difficult to use for all but experienced programmers. Here is a program that makes merging programs easy for anyone with a 1541 disk drive.

To use the program, first **LOAD** it into memory. If keying it in, **SAVE** it before **RUN**ing it, because it erases itself when **RUN**. This program is actually a BASIC loader: The BASIC program loads into memory a machine-language routine contained in **DATA** statements. In this case, the machine language is stored in the standard machine-language area of the C-64's memory—starting at address 49152 (hex $C000) and ending at 49330 ($C0B3). Because this area of memory is not used by BASIC, storing it here will not conflict with BASIC programs being merged.

Once the listing is keyed in and **SAVE**d, **RUN** the program. After a brief pause, the computer will display a short set of instructions on how to use the routine. BASIC programs may now be **LOAD**ed, **SAVE**d, and merged without affecting the machine-language routine. To merge a program from disk with a program in memory, merely insert the disk containing the program to be merged and type **SYS 49152,"**file name**"** (where file name is the name of the program to be merged). The computer will then merge the two programs together.

For example, let's merge the program CALC (from ON DISK 4.3) with a program file CALC.FIX (from ON DISK 4.5). First **LOAD** and **RUN** Merge-64 as it is listed here. Next, type **LOAD"CALC",8** to **LOAD** the old program into memory. Then, put the disk containing CALC.FIX in the drive and type **SYS 49152,"CALC.FIX"**

The computer will now access the disk drive and merge the 2 programs. When the disk light goes off, and the cursor returns to the screen, the two programs will have become one in memory. If the file being merged contains a line number identical to one in memory, the line in memory is replaced.

When a program line is entered from the keyboard, the characters are stored in an input buffer. When the **[RETURN]** key is pressed, the BASIC interpreter abbreviates the line into BASIC "tokens," searches through program memory for the new line's proper location, and inserts it into the program. This machine-language routine does essentially the same thing. Each program line is read from the disk into the same input buffer, but because the line is already tokenized (abbreviated), we simply find the right place in memory and insert the line.

The key to making this routine work lies in changing two "jump vectors" used by the BASIC interpreter. One of these is the IMAIN vector. Every time a program line is entered, the computer jumps "through" this vector to enter "READY." mode. Because merging program files requires inserting more than one line at a time, we redirect this jump vector back to our merge routine. The second vector that has to be changed is the ICALL vector. Whenever a line is inserted into a program, a **CLR** command is executed—**CLOSE**ing all files. We need to keep our disk file containing the program being merged **OPEN**, so we change this vector to keep the file **OPEN**. Both of these vectors will be restored to their original value when the merge program is finished.

*—Randy Thompson*

**For your key-in listings, see HCM PROGRAM LISTINGS Contents.**

### *Merge-64*
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-180 | Program header. |
| 190-200 | READ and POKE merge routine. |
| 210-280 | Print instructions on screen. |
| 290-510 | DATA containing machine language. |

IBMpressions

# Frequency Blender

**by William K. Balthrop**
*HCM Staff*

*Even the most complex sound is made up of simple elements— many layers of sine waves blending into one complicated pattern. Now you can bring this process to light on your computer screen . . .*

Many of us are familiar with the common sine wave. When it is viewed on an oscilloscope, it resembles evenly spaced rolling hills. This simple sine wave, with a constant frequency and amplitude, is quite rare in nature. Most sound waves that we hear are invariably a simultaneous mixture of several frequencies at different amplitudes.

Given several sine waves blended together, what would the resultant wave look like? You might be surprised at the results. The program listed below allows you to enter up to four frequencies, each with its own amplitude. The program then plots the sine waves of the four frequencies on the screen, followed by a fifth wave pattern. This fifth pattern is the algebraic sum, or "composite" of the first four.

### One Good Wave Deserves Another

When you RUN the program, you will be asked first for a frequency, and then for an amplitude, for four sine waves. For best results, the frequencies you enter should be close enough in value to be within a *viewable* range of each other, though this is not a requirement. Finally, you will be asked how many complete cycles of the lowest frequency wave you wish to see displayed. This will determine the horizontal scale of the screen—the time scale. For example, if you choose waves with frequencies 100, 200, 300, and 400, and ask for 5 cycles of the lowest frequency wave, the program will display 5 of the 100 Hz. waves, 10 of the 200 Hz. waves, etc.

Any amplitude you enter is relative to the other specified amplitudes. The program totals all of the amplitudes, and then calculates how much "weight" each one has. If frequency 1 has an amplitude of 150 and frequency 2 has an amplitude of 50, then frequency 2 will only have one third as much effect on the final wave as frequency 1.

One problem arises when you enter high frequencies with low frequencies, and ask to see many cycles of the low frequency. The display may show a lower-frequency sine wave for the high-frequency wave, rather than a thick white bar across the screen as you might expect. This occurs because the computer is a digital device, not analog. The program takes 640 samples across the screen. If the number of cycles is greater than what can be displayed, the program may jump whole cycles in the sampling. This makes for erroneous sampling, and can produce a lower-frequency display.

Examples 3 and 4 are two examples of the sort of screen display you will obtain using this program. For other interesting results, try these two examples:

| Example 1 | Example 2 |
|---|---|
| FREQUENCY: 100<br>AMPLITUDE: 100 | FREQUENCY: 100<br>AMPLITUDE: 100 |
| FREQUENCY: 110<br>AMPLITUDE: 100 | FREQUENCY: 200<br>AMPLITUDE: 100 |
| FREQUENCY: 120<br>AMPLITUDE: 100 | FREQUENCY: 300<br>AMPLITUDE: 100 |
| FREQUENCY: 130<br>AMPLITUDE: 100 | FREQUENCY: 400<br>AMPLITUDE: 100 |
| Number of cycles<br>of Lowest Frequency: 50 | Number of cycles<br>of Lowest Frequency:10 |

### The Program

You might think that the program to do all of this would have to be large and cumbersome. Actually, as you can see by the listing below, it's quite simple.

Two arrays are used to hold the frequency and amplitude values for the four sine waves: FREQ( ), and AMP( ). These values are input with the help of a FOR-NEXT loop in line 230. The value you enter for a frequency should always be a positive number. The amplitude, however, can be either positive or negative. A negative amplitude will cause the wave form to be 180 degrees out of phase. Line 230 also turns off the function key display (line 25), and sets the screen to high resolution (640 by 200).

The function FNF( ) is defined in line 250 to help simplify the program. The parameter passed with this function determines which of the four frequencies are to be used. The value returned by this function is an offset used to determine the vertical position of the next plot position.

Lines 260 through 290 will plot the first four sine waves. Each wave will be plotted only if you have specified a frequency greater than 0. The PSET command is then used to locate the pixel cursor at the start of the line.

A FOR-NEXT loop then controls the line plotting. You will notice that there is only one set of coordinates with the LINE command, with a hyphen before it. This option of the LINE command will plot a line from the last position plotted to the coordinates specified. This ensures that a solid line will always be plotted. The variable Z determines the horizontal position of the line's destination. The function FNF( ) is used to find its vertical position, based on its current horizontal position and the frequency. The amplitude does not cause this wave form to change in size. The amplitude will have an effect on one of the four sine waves only if it is a negative value. If the amplitude is negative, then the wave form displayed will be 180 degrees out of phase with wave forms using a positive amplitude (see Example 4). The value returned by the function is multiplied to enlarge the sine wave for viewing. An offset is then added to position the wave at its proper place on the display.

Line 300 is perhaps the most complex. This is where the final wave form is calculated and displayed. Before the line can be displayed, the variable TA is set to the sum of all the amplitudes. The value added to TA is the ABSsolute value of the amplitudes for each frequency which converts a negative number to its positive counterpart (e.g. – 100 becomes 100). By dividing each frequency's amplitude by TA we can find each frequency's relative "weight" in the total amplitude of the composite wave. Again, PSET is used to locate the pixel cursor, and a FOR-NEXT loop is used to control the wave plotting. Each of the frequencies involved are summed together to arrive at a combined wave. This is accomplished by adding the results of the function FNF( ) for each of the frequencies. But, before the results of the FNF( ) function can be added together, they must be adjusted to their proper amplitude. By multiplying each result by (AMP(x)/TA), the frequency is set to its proper amplitude.

HCM

> ***"Any amplitude you enter is relative to the other specified amplitudes. The program totals all of the amplitudes, and then calculates how much 'weight' each one has."***

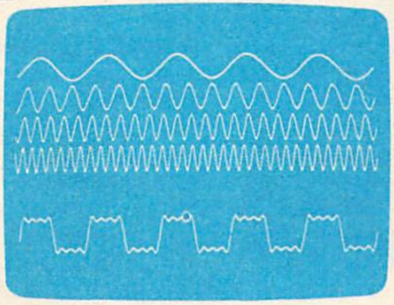
**Example 3:** *This approximates a square wave. Here are the values entered into the program:*

FREQUENCY 1: 100
AMPLITUDE 1: 100

FREQUENCY 2: 300
AMPLITUDE 2: 33

FREQUENCY 3: 500
AMPLITUDE 3: 20

FREQUENCY 4: 700
AMPLITUDE 4: 14.3



Number of cycles of Lowest Frequency: 5

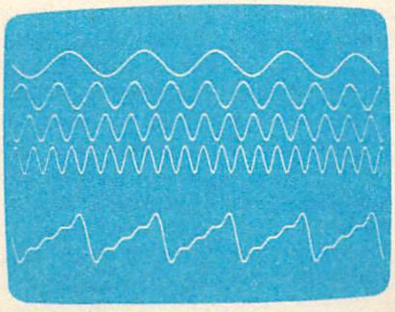
**Example 4:** *This gives a wave similar to an alternating sawtooth wave. Here are the values:*

FREQUENCY 1: 100
AMPLITUDE 1: 100

FREQUENCY 2: 200
AMPLITUDE 2: – 50

FREQUENCY 3: 300
AMPLITUDE 3: 33

FREQUENCY 4: 400
AMPLITUDE 4: – 25



Number of cycles of Lowest Frequency: 5

### *Frequency Blender* (IBM PC & PCjr) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-220 | Program header. |
| 230 | Get frequency and amplitude for the four signals. |
| 240-250 | Determine lowest frequency. |
| 260 | Get scale of the screen. |
| 270 | Define function FNF(F) to convert the frequency at any given point along the axis to a vertical offset for plotting. |
| 280 | Plot frequency 1. |
| 290 | Plot frequency 2. |
| 300 | Plot frequency 3. |
| 310 | Plot frequency 4. |
| 320 | Calculate the sum of the frequencies and plot the fifth signal on the screen. |
| 330 | Wait for a key to be pressed, then restart program. |

```
100 '  ********************
110 '  * FREQUENCY BLENDER *
120 '  ********************
130 '  COPYRIGHT 1985
140 '  EMERALD VALLEY PUBLISHING CO.
150 '  BY WILLIAM K. BALTHROP
160 '  HOME COMPUTER MAGAZINE
170 '  VERSION 5.3.1
180 '  IBM PC W/ BASICA AND
190 '  COLOR/GRAPHICS ADAPTER AND
200 '  COLOR MONITOR OR
210 '  IBM PCjr WITH CARTRIDGE BASIC
220 '
230 KEY OFF:CLS:SCREEN 2:FOR Z=1 TO 4:P
    RINT:PRINT "ENTER FREQUENCY #";Z;:I
    NPUT FREQ(Z):PRINT "ENTER AMPLITUDE
    :";:INPUT AMP(Z)
240 IF FREQ(Z)>0 AND FM=0 THEN FM=FREQ(
    Z) ELSE IF FREQ(Z)<FM AND FREQ(Z)>0
     THEN FM=FREQ(Z)
250 NEXT
260 PRINT:PRINT:PRINT "HOW MANY CYCLES
    OF THE LOWEST FREQUENCY DO YOU WISH
     TO SEE DISPLAYED:":INPUT " CYCLES:
    ",CY
270 DEF FNF(F)=SIN(Z*(FREQ(F)*(.01/(FM/
    CY))*SGN(AMP(F)))))
280 CLS:IF FREQ(1)>0 THEN PSET (0,25):F
    OR Z=0 TO 639:LINE -(Z,FNF(1)*10+25
    ),1:NEXT
290 IF FREQ(2)>0 THEN PSET (0,50):FOR Z
    =0 TO 639:LINE -(Z,FNF(2)*10+50),1:
    NEXT
300 IF FREQ(3)>0 THEN PSET (0,75):FOR Z
    =0 TO 639:LINE -(Z,FNF(3)*10+75),1:
    NEXT
310 IF FREQ(4)>0 THEN PSET (0,100):FOR
    Z=0 TO 639:LINE -(Z,FNF(4)*10+100),
    1:NEXT
320 TA=0:FOR Z=1 TO 4:TA=TA+ABS(AMP(Z))
    :NEXT:PSET (0,160):FOR Z=0 TO 639:L
    INE -(Z,(FNF(1)*(AMP(1)/TA)+FNF(2)*
    (AMP(2)/TA)+FNF(3)*(AMP(3)/TA)+FNF(
    4)*(AMP(4)/TA))*25+160),1:NEXT
330 A$=INKEY$:IF A$="" THEN 330 ELSE 23
    0
```

HCM

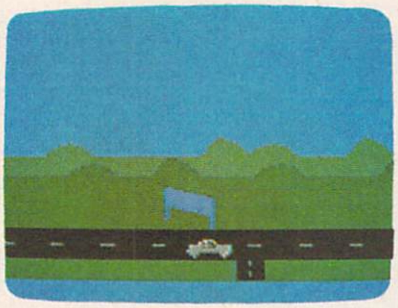
# RAZZLE DAZZLE

**by William K. Balthrop**
*HCM Staff*

***Want to create beautiful 3-D scenery without mountains of code? This SPRITEly tutorial shows you how . . .***

Ever wonder how programmers get the computer to generate three-dimensional graphics in a game? Does it take mountains of computer code to, say, make mountains on the screen? Does it have to be written in machine language? The answer to both of the last two questions is *no*. TI Extended BASIC can do all of this, and more.

With the use of sprites, you can create an almost unlimited number of special effects. For the most part, sprites have been used in computer games as the focus of attention: the alien spaceship, your own tank, flying asteroids . . . But sprites can play a big role in helping to create a more realistic background, as well.



By now you have seen many games in which the background moves in order to create the illusion of motion, while the focus of attention—the object under your control—is almost stationary. In order to do this with normal characters on a large scale, you would probably still need to resort to machine language. However, by using sprites, you can create this effect on a limited scale.

Simulating horizons which move past as you travel down the highway is really quite simple. In fact, you can add several "layers" to the horizon that all move at different speeds. Try looking out the side window of your moving car while sighting past your finger and watching the view outside go by. Objects farther away from you will seem to pass slower than closer objects. This effect is easily transferable to sprites by giving those sprites "farther back" on the horizon slower speeds.

These are only a few of the tricks that sprites are capable of performing. To reinforce the three-dimensional effect, you can also use the fact that "lower-numbered" sprites pass in front of "higher-numbered" sprites. (When creating a sprite in TI Extended BASIC, you assign it a number between 1 and 28.)

Only one drawback hampers the use of sprites—the maximum number of visible sprites on any horizontal row (row meaning individual pixel rows) of the screen is four. If there are more than four sprites on any screen row, those four sprites with the lowest sprite numbers will remain visible, while sprites with higher sprite numbers will become invisible. Under certain circumstances, this can benefit special effects creation, but it does not help us here where we need to be able to create an entire horizon across the screen.

The best solution is to enlarge the sprites to their maximum magnification of 4 by invoking CALL MAGNIFY(4). This creates a sprite that is four characters high by four characters wide. With four of these magnified sprites you can cover 16 characters across the screen. It is thus possible to cover one-half of the screen's width with these four sprites to create our horizon. Then we fill-in the gaps between each sprite where the background shows through by placing normal characters of a certain color behind like-colored sprites. When the sprites move, it's as if an entire section of the screen is moving.

### The Program

In *Horizon*, we actually use two animation tricks. We use sprites as mentioned above, but we also use normal character animation to a limited extent. This is because a problem resides in getting the dashed lines on the highway to scroll by along with the horizon. There are too many dashed lines to do this with our limited number of sprites. Thus, we will settle for a less elegant kind of animation in trade for the increased feeling of motion that the program will evoke.

The sprites will be set to a magnification of four. Because sprites get their shape from the redefinable characters of the TI computer, we must define the graphics shapes of four individual characters for the overall shape of each sprite. Taking this into consideration, we will define the shapes of eight characters to create two sprite shapes (four character shapes per sprite shape). These two sprite shapes will be used for all of the mountains in the background. Fortunately, we can save some overhead in the program because 2 or more sprites can share the same four-character shape definition. Sprites 1, 3, 5, and 7 all use the same shape. The shape is defined in the four ASCII characters 96, 97, 98, and 99 to form one sprite shape. Sprites 2, 4, 6, and 8 use character shapes 100, 101, 102, and 103 to define their sprite shape. The four characters used in a sprite shape must be *contiguous*. In other words, they must all be in a four-character group (e.g., 64, 65, 66, and 67), and must start with a character whose ASCII value is evenly divisible by 4.

We will also need to define sprite shapes for the car, which will remain stationary on the screen. The last 2 sprite shapes we need are for the road intersection and the billboard, both of which "move" alongside the road.

Line 200 contains a FOR-NEXT loop which reads the graphics information for each character from DATA statements and then assigns that information to the proper character. The variable A$ is used later to help animate the lines in the road with standard character animation. Line 210 assigns color to the four character groups we will be using. Notice that only one CALL COLOR statement is required. You can assign as many color groups as you like, up to the legal line-length limit.

Line 220 will paint part of the background with solid colors. This sets the stage for the sprites. Earlier we stated that because the sprites can only cover one-half of the screen, we need to place background characters of the same color behind them.

> *"By now you have seen many games in which the background moves in order to create the illusion of motion . . . by using sprites, you can create this effect on a limited scale."*

Line 230 and 240 will paint a dashed line down the road. The line is actually a solid line of four different characters repeated again and again. Only one character at a time, however, will contain a shape that can be seen on the screen. The other characters are assigned a shape with no pixels turned on—so they are invisible.

Line 250 sets the magnification factor for the sprites to 4. As discussed earlier, this will cause the sprites to reach their maximum size.

Lines 260, 270 and 280 each consist of only one statement: CALL SPRITE. However, these three lines are responsible for the initialization, placement, and speed of all 11 sprites. Line 260 will assign sprite numbers 5, 6, 7, and 8. These sprites will each use one of the two mountain shapes, and will all be placed on the screen to appear as the mountains closest to the viewer. Line 270 places the next four mountains on the screen, farther away from the viewer and moving more slowly. Finally, line 280 places the car, intersection, and billboard on the road.

The final character animation takes place in lines 290 through 350. The variable X will continue counting from 1 to 4 as long as you let the program run. This controls the location to which the ON GOSUB statement in line 310 will branch. Each of the statements in lines 320 to 350 cause the shapes used for the dashed line on the highway to shift. The shape is cleared from one character and added to another. HCM

### *Horizon* (TI-99/4A) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-190 | Program header. |
| 200-210 | Initialize program graphics and color. |
| 220-240 | Display static character graphics. |
| 250-280 | Place sprites on the screen. |
| 290-350 | Animate the dashed line on the highway. |
| 360-380 | Character-graphics shape data. |

XB *Horizon* requires TI Extended BASIC.

```
100 ! ***********
110 ! * HORIZON *
120 ! ***********
130 ! COPYRIGHT 1985
140 ! EMERALD VALLEY PUBLISHING CO.
150 ! BY WILLIAM K. BALTHROP
160 ! HOME COMPUTER MAGAZINE
170 ! VERSION 5.3.1
180 ! TI EXTENDED BASIC
190 !
200 CALL CLEAR :: FOR Z=96 TO 116 :: RE
    AD A$ :: CALL CHAR(Z,A$):: NEXT Z :
    : A$="00000000FF"
210 CALL COLOR(11,16,2,12,2,2,13,13,13,
    14,4,4)
220 CALL HCHAR(13,1,136,64):: CALL HCHA
    R(15,1,128,160):: CALL HCHAR(20,1,1
    20,96):: CALL HCHAR(23,1,128,64)
230 X=111 :: FOR Z=1 TO 32 :: X=X+1 ::
    IF X>115 THEN X=112
240 CALL HCHAR(21,Z,X):: NEXT Z
250 CALL MAGNIFY(4)
260 CALL SPRITE(#5,96,13,83,1,0,-6,#6,1
    00,13,83,55,0,-6,#7,96,13,83,120,0,
    -6,#8,100,13,83,200,0,-6)
270 CALL SPRITE(#1,96,4,67,228,0,-2,#2,
    100,4,67,35,0,-2,#3,96,4,67,100,0,-
    2,#4,100,4,67,200,0,-2)
280 CALL SPRITE(#9,108,2,161,240,0,-10,
    #10,104,15,143,120,#11,116,5,121,20
    0,0,-10)
290 X=0
300 X=X+1 :: IF X=5 THEN X=1
310 ON X GOSUB 320,330,340,350 :: GOTO
    300
320 CALL CHAR(112,"0",115,A$):: RETURN
330 CALL CHAR(115,"0",114,A$):: RETURN
340 CALL CHAR(114,"0",113,A$):: RETURN
350 CALL CHAR(113,"0",112,A$):: RETURN
360 DATA 0,0000000071F3F7FFF,0,0000C0F8F
    CFEFFFF,0,070F1F3F3F7FFFFF,0,00C0F0
    F8FCFEFFFF
370 DATA 0,00070A7A7FFF7020,0,008040 3EF
    EFE1C08,0,1E1E1F1F1E1E1F1F,0,F0F0F0
    F0F0F0F0F0,00000000FF
380 DATA 0,0,0,E0FCFFFFFFFFFF9F83808080
    00000000000000C0FCFFFFFFFFFF1F030101 0
    10101
```

# Selective Input

When writing a BASIC program (especially a productivity program), it is often wise to limit the keyboard input that the program will accept. For example, only certain characters should be used to input a PC-DOS file name. If you use the **INPUT** statement, a user can key-in any characters or numbers; but only after **[ENTER]** has been pressed, is the input scanned. The **INPUT** statement is only selective if a numeric variable is specified. Further, if an incorrect character is input, the BASIC interpreter responds with a somewhat cryptic "Redo from start" message, which is not exactly a "user-friendly" way of telling the user that the wrong thing has been keyed in.

```
  100 '  ******************
  110 '  * SELECTIVE INPUT *
  120 '  ******************
  130 '  COPYRIGHT 1985
  140 '  EMERALD VALLEY PUBLISHING CO
  150 '  BY ROGER WOOD
  160 '  HOME COMPUTER MAGAZINE
  170 '  VERSION 5.3.1
  180 '  IBM PC OR PCjr BASIC
  190 '
  200 CLS:PRINT "INPUT FILE NAME: ";:ROW=
      1:COL=18:MAXLEN=8:SELECT$="ABCDEFGH
      IJKLMNOPQRSTUVWXYZ0123456789O!@#$%&
      (){}'`-_":GOSUB 10010:END
10000 '  INPUT SUBROUTINE *****
10010 IN$="":PT=1
10020 LOCATE ROW,COL,0:PRINT IN$;SPACE$(M
      AXLEN-LEN(IN$));:LOCATE ROW,COL+(PT
      -1),1:K$="":WHILE K$="":K$=INKEY$:W
      END
10030 IF K$=CHR$(13) THEN RETURN
10040 IF INSTR(SELECT$,K$) THEN IN$=LEFT$
      (IN$,PT-1)+K$+MID$(IN$,PT+1):PT=PT+
      1:IF PT>MAXLEN THEN PT=MAXLEN:GOTO
      10020
10050 IF K$=CHR$(8) AND PT>1 THEN IN$=LEF
      T$(IN$,PT-2)+MID$(IN$,PT):PT=PT-1:G
      OTO 10020
10060 IF K$=CHR$(0)+CHR$(83) THEN IN$=LEF
      T$(IN$,PT-1)+MID$(IN$,PT+1):GOTO 10
      020
10070 IF K$=CHR$(0)+CHR$(82) AND LEN(IN$)
      <MAXLEN THEN IN$=LEFT$(IN$,PT-1)+"
      "+MID$(IN$,PT):GOTO 10020
10080 IF K$=CHR$(0)+CHR$(77) AND LEN(IN$)
      >=PT THEN PT=PT+1:IF PT>MAXLEN THEN
       PT=MAXLEN:BEEP:GOTO 10020
10090 IF K$=CHR$(0)+CHR$(75) AND LEN(IN$)
      >1 THEN PT=PT-1:IF PT<1 THEN PT=1 :
      GOTO 10020
10100 GOTO 10020
```

It would be much more helpful if the program could screen input as it is keyed in, and then check it against a list of characters that the program will accept. If only one key is to be pressed, the **INKEY$** statement is adequate to limit character input. If, however, longer input is desired (such as a file name), a lengthier routine is required. The demonstration program at the left contains just such a selective input subroutine in lines 10000-10100. Line 200 prompts the user for the file name, then specifies the parameters needed by the input subroutine and does a **GOSUB** to the subroutine.

The **ROW** variable contains the row where the input is to be accepted, and the **COL** variable the column where it is to begin. **MAXLEN** is the maximum length of the input. A word of caution here: If **MAXLEN+COL** exceeds the maximum width of the screen (either 40 or 80 columns depending on whether the **SCREEN** mode or **WIDTH** command is being used), then the **LOCATE** statement in line 10020 could have a value placed in it that will cause an illegal function call. It is up to you, the programmer to see that this doesn't happen.

The **SELECT$** variable is a list of all the characters that you have determined are acceptable. In the program shown here, we've selected legal characters for a PC-DOS 2.10 file name, although this could be modified to select any list of characters. (Notice that we arbitrarily chose not to include small letters.) The key to this routine is the **INSTR** function in line 10040. It scans all the characters in **SELECT$** and checks on whether **K$** (the last key pressed) is acceptable. If so, it is added to **IN$**; otherwise, **K$** is checked to see whether the key pressed is an editing key (such as **[BACKSPACE]**). If the keypress does not fit into either of these categories, it is ignored, and the routine continues to scan the keyboard. The routine is exited only when the **[ENTER]** key is pressed.

*—Roger Wood*

# Full-Screen Editor

**ACCEPT AT** is one command built into TI Extended BASIC that makes normally complex tasks very simple. A few of its many powerful options include the ability to (1) accept input from any location on the screen, (2) limit input to a specified number of characters, (3) check the input for valid characters, (4) erase the screen before input, and (5) sound a beep when the prompt appears. Of interest here is the **SIZE** option, which allows you to create a field in which characters can be entered.

```
100 !  ******************
110 !  *  SCREEN EDITOR *
120 !  ******************
130 !  COPYRIGHT 1985
140 !  EMERALD VALLEY PUBLISHING CO.
150 I  BY WILLIAM K. BALTHROP
160 !  HOME COMPUTER MAGAZINE
170 !  VERSION 5.3.1
180 !  TI EXTENDED BASIC
190 !
200 CALL CLEAR :: LN=1 :: DIM L$(24)
210 ACCEPT AT(LN,1)SIZE(-28):L$(LN):: C
    ALL KEY(0,K,S)
220 IF K=10 OR K=13 THEN LN=LN+1 :: IF
    LN>24 THEN LN=1 :: GOTO 210 ELSE 21
    0
230 IF K=11 THEN LN=LN-1 :: IF LN<1 THE
    N LN=24 :: GOTO 210 ELSE 210
240 GOTO 210
```

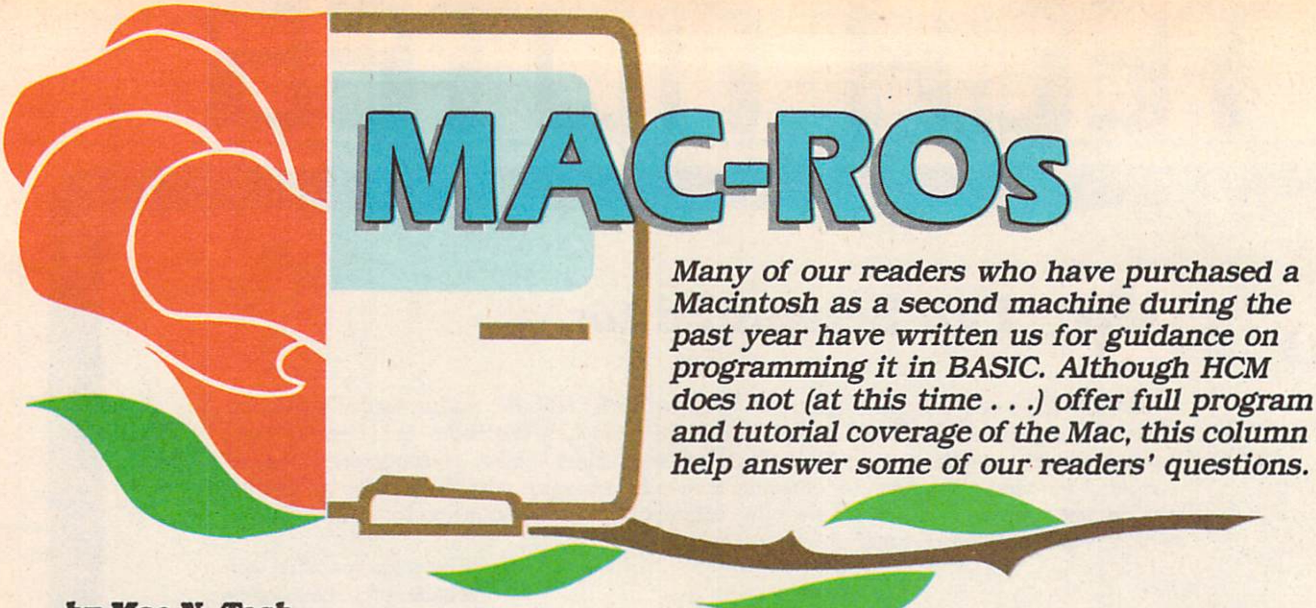
The value specified with the **SIZE** option determines the number of characters that may be entered (the maximum value is 28). The **ACCEPT AT** command will not "wrap" input to the next screen line. However, it is more than adequate for a screen editor, as exhibited in the program listed here. Either a negative or a positive value may be specified with the **SIZE** option. If a positive value is used, then only those characters typed after the prompt appears will be accepted. If a negative number is used, then any characters already on the screen within the field of the **ACCEPT AT** statement will also be accepted as input. This attribute permits input defaults simply by printing the default on the screen before the **ACCEPT AT** statement occurs.

The ability to read what is already on the screen makes for a perfect tool in the design of a full-screen editor. This program only allows text to be entered on the screen. (The text is placed into the **L$( )** array.) To do something with the text created here—save it to disk, print it out to the printer, or go to a second page of text—would require adding more code to the program. This routine is only meant to illustrate the abilities of the **ACCEPT AT** statement.

After this program runs, the screen will clear, and the cursor will start blinking in the upper left-hand corner. When typed characters reach the end of a line, the computer will beep. Characters may continue to be entered, but they will only replace the last character on the line. The input for that line may be accepted in one of three ways: by pressing **[ENTER]**, **[FCTN] X**, or **[FCTN] E**. Pressing **[ENTER]** and **[FCTN] X** will have the same effect—the cursor will move down to the first character position of the next line. Pressing **[FCTN] E** will move the cursor to the first character position of the next higher line. Attempting to wrap the cursor off the top or bottom of the screen will make it appear at the other end of the screen.

**CALL KEY** is used immediately after the **ACCEPT AT** statement so that the program can determine which key was used to accept the input. If one of the three keys to accept the input is pressed too rapidly, the program may not catch it in time. In this case, the cursor will simply be returned to the beginning of the line currently being worked on.

*—William K. Balthrop*



# MAC-ROs

*Many of our readers who have purchased a Macintosh as a second machine during the past year have written us for guidance on programming it in BASIC. Although HCM does not (at this time . . .) offer full program and tutorial coverage of the Mac, this column will help answer some of our readers' questions.*

***by Mac N. Tosh***

At first glance, the listing accompanying this article may look like some new, high-level programming language straight from the think tank at Microsoft. But a closer look will reveal true BASIC commands built around an entirely different concept—seen before only in highly structured programming languages.

### What is a Macro?

The word macro comes from the Greeks and means "large" or "expanded." In the world of computers, the term macro is generally used to describe a subroutine or utility which adds new commands, enhancing or otherwise expanding the existing vocabulary of the language.

With Microsoft BASIC version 2.0 for Macintosh, you can create macros (SUBroutines) which can be CALLed or used as any other command in BASIC is used (without the CALL keyword). The most powerful feature of these macros is their portability. Once a macro is defined, it can be used in *any* Microsoft BASIC program. By inserting a macro into a program, you will, in fact, be expanding the language.

Macros are most commonly used to replace repetitious code in a program. Before this version of Microsoft BASIC was available, BASIC programmers avoided repetition by using GOSUB's to branch to a subroutine. Unfortunately, the line number used with a GOSUB tells you nothing about the routine to which you are branching. Using the same subroutine in different programs will invariably result in it being relocated to different line numbers. This unstructured approach to programming makes programs difficult to read and debug.

No line numbers are used in Microsoft BASIC for Macintosh, only labels. You could use a number as a label, of course, though words which describe the entry point of a routine in a program make much more sense. You can then use the label with the GOTO or GOSUB statements just as you would do with line numbers. Compare the lines in these two examples and decide which one is easier to understand:

Conventional BASIC

```
 100 ON DIR GOSUB 200,300
.

200 PRINT "NORTH":RETURN
300 PRINT "SOUTH":RETURN
```

Microsoft BASIC for Macintosh

```
 ON DIR GOSUB NORTH,SOUTH
.

NORTH:PRINT "NORTH":RETURN
SOUTH:PRINT "SOUTH":RETURN
```

### A Drawing Macro

The fact that you can create your own macros for use in any program is even more powerful than this method of labeling. For example, there is one macro used in the program below (that creates the Mac faces shown in the illustration) called DRAW. The beginning of this special subroutine begins with the following line:

```
SUB DRAW (mode,p$) STATIC
```

The word SUB in the line denotes the beginning of a new subroutine. The name of the subroutine follows next: DRAW. This is the name you will use when you CALL this routine. You can CALL the routine using one of two methods:

```
CALL DRAW(mode,d$)
or
DRAW mode,d$
```

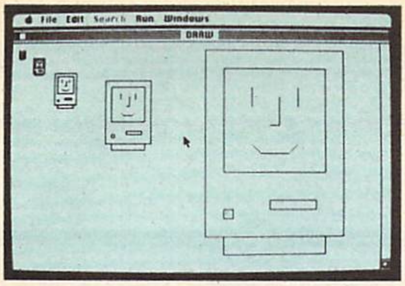
The method you use is simply up to your personal preference—there is no difference in the way they function. In the program shown here, we used the second method. When expressed in this fashion, it becomes a new command in this already powerful Microsoft BASIC.

As to the number of parameters that may be passed to a subroutine, you are limited only by the maximum length of a BASIC line (255 characters). In the SUBroutine DRAW, we pass two parameters.

The first parameter passed to the DRAW routine sets up the *mode* in which the shape is to be drawn. The Macintosh *toolbox* has provided for up to eight modes of line drawing. The mode parameter allows the DRAW command to access those modes. [For more information on the modes used by the system, consult the *Microsoft BASIC User's Manual* section on Macintosh ROM routines, under the topic CALL PENMODE—Ed.]

> ***"Once a macro is defined, it can be used in any Microsoft BASIC program. By inserting a macro into a program, you will . . . be expanding the language."***

The second parameter is the command string. This string contains a series of commands which will instruct the computer to draw the shape. A brief description of these commands follows.



*Above is the result of the program at the right.*

R *dis* —Draw right. This command will draw a line to the right a specified distance (*dis*). The parameter *dis* is optional. If the parameter is omitted, the distance defaults to 1. The following directional commands all work the same, except for the direction in which they draw.

D *dis*— Draw down.
L *dis*— Draw left.
U *dis*— Draw up.
E *dis*— Draw up and to the right.
F *dis*— Draw down and to the right.
G *dis*— Draw down and to the left.
H *dis*— Draw up and to the left.

B *com*—When this command is used, it will cause the next command (*com*) to move without drawing—it "lifts the pen" for the next command. Commands following the one affected by the B command will start drawing again—the B command affects only one command.

M *pos1,pos2* —This will change the absolute position of the pen that is used for drawing the shape. The X coordinate is *pos1*, and *pos2* is the Y coordinate.

N *com*—Like the B command, when the N command is encountered, only the following command is affected. The N command will cause the next command to draw its line segment, then return to the point from which it started. It's useful for drawing "rays."

S *scale*—This command is used to alter the scale of the drawing. If *scale* has a value of 4, the ratio of the drawing will be 1:1 with the commands in the command string; R5 will draw a line 5 pixels to the right. A value of 8 following the S command will draw the shape at twice that size—a 2:1 scale ratio. A value of 2 will draw the shape at half scale—a 1:2 ratio. The value assigned to *scale* must be an integer number.

*Don't forget to turn to this column in upcoming issues of HCM for more answers to your questions on Microsoft BASIC, and other topics related to Macintosh use.*

HCM

```
' *********************
'  MAC-ROs / DRAW
' *********************
' COPYRIGHT 1985
' EMERALD VALLEY PUBLISHING CO.
' BY WILLIAM K. BALTHROP
' HOME COMPUTER MAGAZINE
' VERSION 5.3.1
' MACINTOSH W/MICROSOFT BASIC 2.0
'

start:
   mode=8
   READ p1$,P2$:P$=P1$+P2$
   d$="bm10,10s2"+p$:DRAW mode,d$
   d$="bm30,20s4"+p$:DRAW mode,d$
   d$="bm60,40s8"+p$:DRAW mode,d$
   d$="bm130,50s16"+p$:DRAW mode,d$
   d$="bm260,10s48"+p$:DRAW mode,d$
   a$="":WHILE a$="":a$=INKEY$:WEND
END

DATA "D20R2D2R11U2NL11R2U20L15BF2D11R11U11L11BF2"
DATA "BRND2BR5ND2BLBGD3LBG2FR3EBD7BL3UR5DL5BL4LDRU"

' *******************
' This is the DRAW subroutine. Similar to the DRAW command on an
' IBM PC or PCjr.
' The X, P, C, A and TA functions of the IBM versions DRAW command
' have not been implemented.
' *******************

SUB DRAW (mode,p$) STATIC
   CALL PENNORMAL
   CALL PENMODE(mode)
   CALL SHOWPEN
   p$=UCASE$(p$):drawmode=0
   IF scale=0 THEN scale=1
   FOR a=1 TO LEN(p$):ox=x:oy=y
     char$=MID$(p$,a,1)
     d=INSTR("RDLUEFGH",char$):IF d>0 THEN dodraw
     IF char$="M" THEN GOSUB movepoint:GOTO doline
     IF char$="B" THEN drawmode=1
     IF char$="N" THEN lineonly=1
     IF char$="S" THEN GOSUB getnumber:scale=number*.25
     GOTO endofloop
   dodraw:
      ON d GOSUB right,down,left,up,upright,downright,downleft,upleft
   doline:
      IF drawmode=1 THEN CALL MOVETO (x,y) ELSE CALL LINETO (x,y)
      IF lineonly=1 THEN x=ox:y=oy:CALL MOVETO (x,y)
      lineonly=0:drawmode=0
   endofloop:
      NEXT
EXIT SUB

   movepoint:
      GOSUB getnumber:x=number:a=a+1
      GOSUB getnumber:y=number
   RETURN
   right: GOSUB getnumber:x=x+number*scale
   RETURN
   down: GOSUB getnumber:y=y+number*scale
   RETURN
   left: GOSUB getnumber:x=x-number*scale
   RETURN
   up: GOSUB getnumber:y=y-number*scale
   RETURN
   upright: GOSUB getnumber:x=x+number*scale:y=y-number*scale
   RETURN
   downright: GOSUB getnumber:x=x+number*scale:y=y+number*scale
   RETURN
   downleft: GOSUB getnumber:x=x-number*scale:y=y+number*scale
   RETURN
   upleft: GOSUB getnumber:x=x-number*scale:y=y-number*scale
   RETURN
   getnumber:
           num$=""
   num1:   a=a+1
           IF a<=LEN(p$) THEN num2
           IF num$="" THEN number=1:RETURN
           number=VAL(num$):RETURN
   num2:   c$=MID$(p$,a,1)
          IF c$>="0" AND c$<="9" THEN num3
          IF num$="" THEN number=1:a=a-1:RETURN
          number=VAL(num$):a=a-1:RETURN
   num3:  num$=num$+c$:GOTO num1
 END SUB
```

HCM

# Group Grapevine

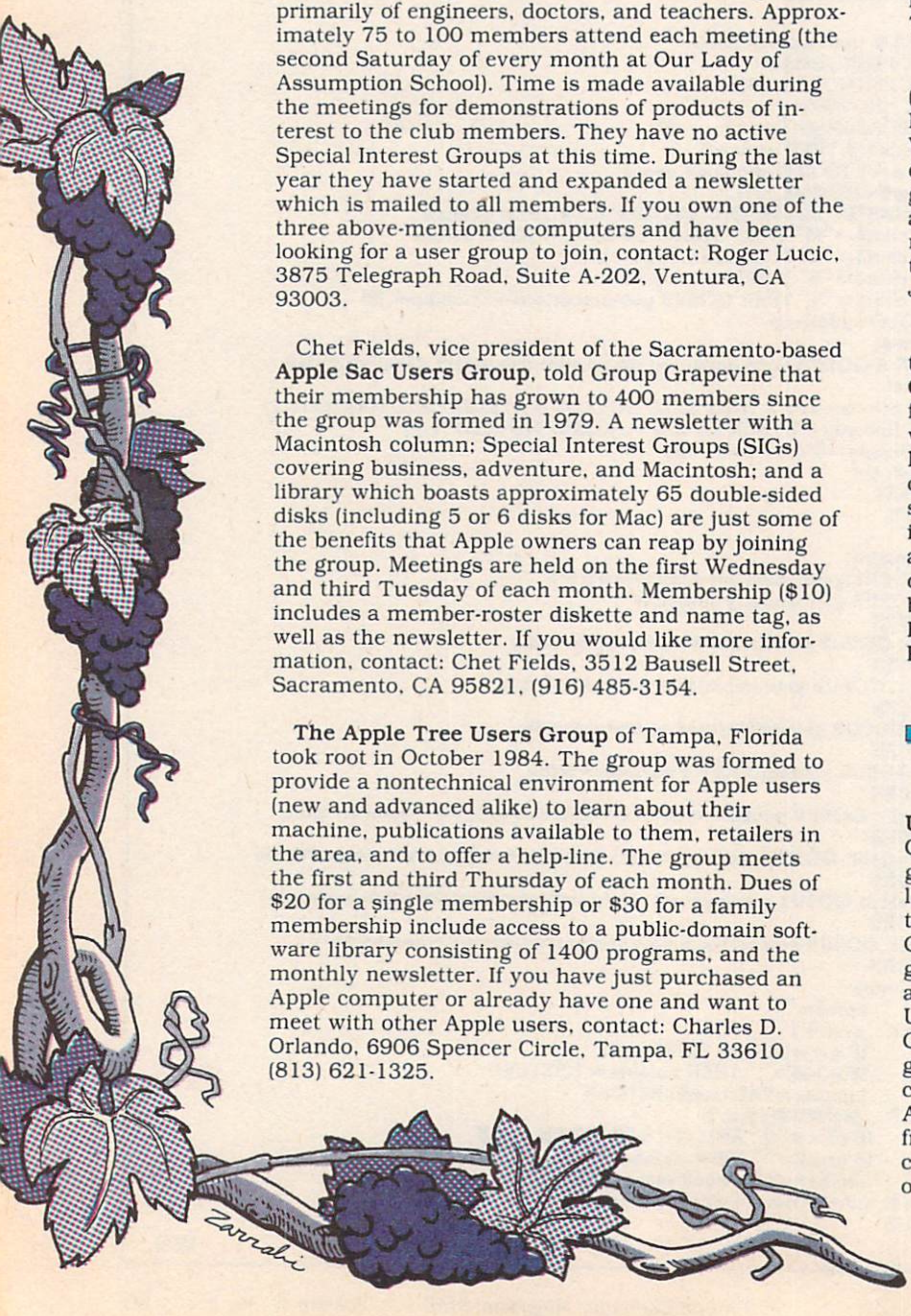
***News, information and upcoming events of home computer users groups around the world.***

**Looking to join a users group, exchange newsletters or software, increase your users group's membership or pep up your next meeting's agenda? For the latest users group news, put your ear to the Group Grapevine. And if you have a message to put out to other groups, if you are starting a new group, or have an interesting item to share, send a note or picture—or better yet, a group newsletter—to the Users Group Editor, Home Computer Magazine, 1500 Valley River Drive, Suite 250, Eugene, OR 97401, (503) 485-8796.**

**Appleholics Anonymous** in Ventura, California has been an active Apple user group for approximately 5 years. At present, the club has in excess of 150 active members, with 5 to 10 new members attending each month. The club is strictly an Apple II+, *II*e and Macintosh user group, with membership composed primarily of engineers, doctors, and teachers. Approximately 75 to 100 members attend each meeting (the second Saturday of every month at Our Lady of Assumption School). Time is made available during the meetings for demonstrations of products of interest to the club members. They have no active Special Interest Groups at this time. During the last year they have started and expanded a newsletter which is mailed to all members. If you own one of the three above-mentioned computers and have been looking for a user group to join, contact: Roger Lucic, 3875 Telegraph Road, Suite A-202, Ventura, CA 93003.

Chet Fields, vice president of the Sacramento-based **Apple Sac Users Group**, told Group Grapevine that their membership has grown to 400 members since the group was formed in 1979. A newsletter with a Macintosh column; Special Interest Groups (SIGs) covering business, adventure, and Macintosh; and a library which boasts approximately 65 double-sided disks (including 5 or 6 disks for Mac) are just some of the benefits that Apple owners can reap by joining the group. Meetings are held on the first Wednesday and third Tuesday of each month. Membership ($10) includes a member-roster diskette and name tag, as well as the newsletter. If you would like more information, contact: Chet Fields, 3512 Bausell Street, Sacramento, CA 95821, (916) 485-3154.

**The Apple Tree Users Group** of Tampa, Florida took root in October 1984. The group was formed to provide a nontechnical environment for Apple users (new and advanced alike) to learn about their machine, publications available to them, retailers in the area, and to offer a help-line. The group meets the first and third Thursday of each month. Dues of $20 for a single membership or $30 for a family membership include access to a public-domain software library consisting of 1400 programs, and the monthly newsletter. If you have just purchased an Apple computer or already have one and want to meet with other Apple users, contact: Charles D. Orlando, 6906 Spencer Circle, Tampa, FL 33610 (813) 621-1325.



C64

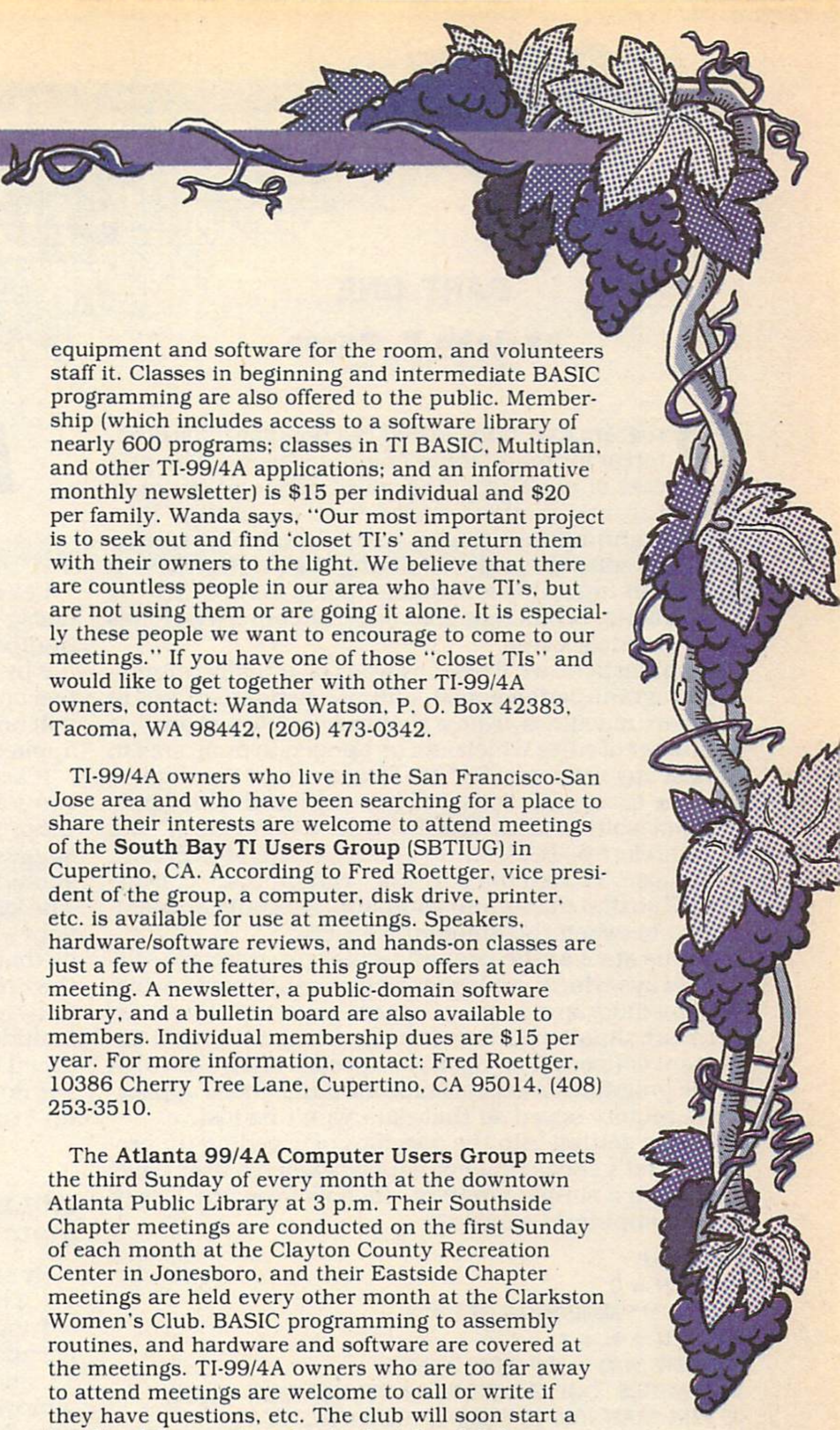
A Commodore users group convenient to both Minneapolis and St. Paul residents was recently established in Mendota, Minnesota. **The Metro-Area Commodore Computer Club** was formed just three months ago, and already membership has swollen to 100. Currently they are experimenting with hard-copy documentation (booklet form) to accompany their monthly disks. If the project proves successful, they may syndicate their efforts to other clubs by offering the camera-ready artwork or copies of the finished product for use. They would like to swap newsletters and ideas with other clubs and form an exchange program with foreign clubs on a regular basis. For more information, contact: James W. Meehan, Jr., Box M, Mendota, MN 55150, (612) 729-0232.

The **Rhode Island Commodore Users Group** (RICUG) in Johnston, Rhode Island meets on the first Monday of every month at 7:30 p.m. at the Video Vault. A monthly newsletter and a library of public-domain programs is available to all members. The group puts great emphasis on family membership and believes in supporting all members, from beginners to Special Interest Groups (SIGs). For information, write to: Joe Osborne, 4 Mowry Avenue, Johnston, RI 02919, (401) 231-3537.

Group Grapevine would like to take this opportunity to introduce the **North Coast Commodore Users Group** (NCCUG) of Erie, Pennsylvania. NCCUG was formed in the fall of 1983 to meet the needs of Northwestern Pennsylvania's Commodore users. With the help of local retailers, the group has sponsored successful demonstrations of the latest hardware and software available to Commodore owners. The group is divided into two chapters—the Erie/Main Chapter and the Edinboro Chapter. NCCUG boasts a public-domain library, a monthly newsletter, discounted blank diskettes, and Special Interest Groups (SIGs). For more information, contact: Dennis Witkowski, P.O. Box 6117, Erie, PA 16512, (814) 398-8146.

PC jr

Spokesperson Ben Blackstock of the **Hawkeye PC Users Group** in Cedar Rapids, Iowa gave Group Grapevine an earful of information on what this group is up to. Hawkeye PC meets at two different locations on the fourth Monday of every month—at the Cedar Rapids location one month and in Iowa City the next. Education is the main thrust of this group of approximately 200 members, most of whom are professional and nonprofessional people from the University of Iowa, Rockwell, etc. Special Interest Groups consist of communications, Lotus, and programmers. Many of their meetings deal with specific computer problems that members are encountering. A lot of the speakers for these meetings are gleaned from the group to talk about such topics as word processing and data-base programs. Mr. Blackstock operates a 24-hour bulletin board which is constantly



busy receiving calls from all over the country, as well as from overseas. Speaking of overseas, this club also has members from Germany and Italy. Dues are $10 per year, which includes an excellent newsletter and access to the club library of 42 double-sided disks of public-domain software. For more information, contact: Ben Blackstock, 385 Collins Road NE, #201, Cedar Rapids, IA 52402, (319) 393-5416.

Formed on November 28, 1984, the **Mohawk IBM PCjr Users Group** in Albany, New York is growing at a steady pace every month and now has 30 active members, according to spokesperson Gary Hermansen. Most of the membership is interested in business applications such as word processing and data bases for the PCjr. This group puts out a monthly newsletter and has a public-domain library. If you are a PCjr owner looking for others of like species, contact: Gary Hermansen, P. O. Box 12-305, Albany, NY 12212, (518) 869-9787.

Group Grapevine stepped into its own backyard and spoke to Jim Cox of the **Eugene IBM PC Club**, Eugene, Oregon. The Eugene club is not limited to PC owners, but welcomes PCjr and IBM-compatible owners as well. Meetings, on the third Thursday of each month, consist of a half-hour of questions and answers, a panel discussion, a speaker, and a demonstration of software or hardware. Dues are $20 per year, and include access to their disk library and a subscription to their monthly newsletter. For more information, contact: Jim Cox, P. O. Box 5070, Eugene, OR 97405.

**South, Mobile, and Alabama User's Group** (SMAUG) of Grand Bay, Alabama meets on the second and fourth Tuesdays of every month at 7 p.m. in the University of South Alabama Engineering Department building to provide technical assistance to owners of the TI-99/4 and 4A computers. By pooling their expertise they can assist in solving programming problems, and will share ideas, demonstrate equipment, and discuss new software for and uses of the TI home computer. Membership fees are $10 a year for individuals and $15 for families. Benefits include a newsletter and access to their public-domain library. If you are a TI-99/4A owner living in this area and would like the fellowship of people with a like interest, contact: Daniel L. Parrott, Route 2, Box 169-B, Grand Bay, AL 36541, (205) 865-6686.

The president of the **Tacoma Eastside 99ers Users Group**, Wanda Watson, recently wrote to Group Grapevine with information regarding their group. Beginning with just 5 members one year ago, they have grown to nearly 50 members today. Besides offering support to members, it is the goal of the group to promote computer literacy in the community. In October 1984 they opened a free, public-access computer room. Group members donated or loaned equipment and software for the room, and volunteers staff it. Classes in beginning and intermediate BASIC programming are also offered to the public. Membership (which includes access to a software library of nearly 600 programs; classes in TI BASIC, Multiplan, and other TI-99/4A applications; and an informative monthly newsletter) is $15 per individual and $20 per family. Wanda says, "Our most important project is to seek out and find 'closet TI's' and return them with their owners to the light. We believe that there are countless people in our area who have TI's, but are not using them or are going it alone. It is especially these people we want to encourage to come to our meetings." If you have one of those "closet TIs" and would like to get together with other TI-99/4A owners, contact: Wanda Watson, P. O. Box 42383, Tacoma, WA 98442, (206) 473-0342.

TI-99/4A owners who live in the San Francisco-San Jose area and who have been searching for a place to share their interests are welcome to attend meetings of the **South Bay TI Users Group** (SBTIUG) in Cupertino, CA. According to Fred Roettger, vice president of the group, a computer, disk drive, printer, etc. is available for use at meetings. Speakers, hardware/software reviews, and hands-on classes are just a few of the features this group offers at each meeting. A newsletter, a public-domain software library, and a bulletin board are also available to members. Individual membership dues are $15 per year. For more information, contact: Fred Roettger, 10386 Cherry Tree Lane, Cupertino, CA 95014, (408) 253-3510.

The **Atlanta 99/4A Computer Users Group** meets the third Sunday of every month at the downtown Atlanta Public Library at 3 p.m. Their Southside Chapter meetings are conducted on the first Sunday of each month at the Clayton County Recreation Center in Jonesboro, and their Eastside Chapter meetings are held every other month at the Clarkston Women's Club. BASIC programming to assembly routines, and hardware and software are covered at the meetings. TI-99/4A owners who are too far away to attend meetings are welcome to call or write if they have questions, etc. The club will soon start a Lending Library where books that the club has purchased can be checked out by members at the monthly meetings. For more information, contact: Marshall Gordon, P. O. Box 19841, Atlanta, GA 30325.

Information just blew in from **The Windy City 99 Club** of Chicago, Illinois. These TI-99/4A owners meet the third Thursday of every month at 7:30 p.m. and offer as well as a user's library featuring business, educational, and game software. If you would like to learn more about this club (established in 1982), write: Michael Mickelsen, 640 N. LaSalle Street, Suite 280, Chicago, IL 60610.

**HCM**

# Speeding Up A BASIC Program

## PART ONE

**by John P. Russo**
*and the HCM Staff*

Program execution speed is a popular topic of interest for many microcomputer users. The process of measuring the speed of one program or system against another is called "benchmarking." It involves running and timing two functionally similar pieces of software and calculating their differences in execution time. It is a valuable tool in comparing the performance of different machines, but otherwise it has little practical value.

In this article, we'll discuss ways of actually improving program performance, and use benchmarking to compare machines *against themselves*. We will present a number of rules which can be applied to programs in almost any version of BASIC. However, the times given and the examples discussed here will deal with Apple *II* (Applesoft) BASIC, IBM PC DOS 2.0 BASIC (Disk), Commodore 64 BASIC, and TI-99/4A Extended BASIC. We chose TI Extended BASIC rather than console BASIC so that maximum compatibility could be maintained between the different BASICs.

Let us state at the beginning that program speed is not always critical, and other factors—such as readability or memory space—may be more important. However, there are situations where fast execution is the most important concern. For example, speed is often critical in game programs. Also, certain communications applications require speed so that data won't be lost.

Before getting into the specifics of speeding up programs, let's consider what happens when a user types RUN when a simple program, such as *Program 1*, is in your computer's memory:

*Program1*

```
10 LET N = 0
15 REM *** BEGINNING OF LOOP
20 LET N = N + 1
25 IF INT (N/7) = N/7 THEN PRINT N;
30 REM THIS LOOP CHECKS EACH NUMBER THAT IS LESS
35 REM THAN 1000 TO SEE IF IT IS DIVISIBLE BY 7
40 REM IF SO, THE NUMBER IS PRINTED.
45 IF N < 1000 THEN GOTO 20
50 REM *** END OF LOOP
```

Line 10 is, of course, written in BASIC, a language which many humans understand, but which is not the native tongue (machine language) of a microcomputer. Before a computer can execute the instruction on line 10, it must be provided with an equivalent instruction written in its native tongue. Rather than investigate all the details of interpreted BASIC, we can view the execution of a BASIC statement as a two-step process, namely:

1. Translate the BASIC statement into machine language.
2. Execute the machine-language statement.

The major disadvantage of interpreted versions of BASIC is that step 1 consumes so much time. Some lines, such as those within loops, must be translated many times during the execution of a program.

Let's take a closer look at *Program 1*. The program doesn't do anything very exciting—it merely prints all numbers which are less than 1000 and (evenly) divisible by 7. It is a short program—excluding remarks it has only 4 lines. Despite this program's briefness, we will find that there are many techniques which can be applied to this program to speed it up.

It seems reasonable that our efforts to decrease execution time should center on loops, because they are the primary time-consumers in a running program. This suggests that if we wish to speed up *Program 1*, we should look more closely at the loop in lines 20 to 45. The REM statements within this loop may have caught your eye. Most of us acknowledge (perhaps grudgingly) that program remarks are important. However, when they're placed within loops, they can lengthen execution time by a surprising amount. If remarks must be included in the working version of your program and speed is essential, then make sure the remarks are not within loop bodies. Behold: we have our first "speed-up" rule:

**RULE 1: Remove REMark statements from program loops.**

By applying this rule to our program, we get *Program 1.1*. The new program runs significantly faster than the old one on three machines, and on the IBM PC it actually runs 25 percent faster. The REM statements on line 15 and 45 have not been removed, because their removal would not have saved a significant amount of time. Note also that although variables in BASIC are initialized to 0 at run time, deleting the apparently unnecessary line 10 would not yield any worthwhile time savings.

*Program 1.1*

```
10 LET N = 0
15 REM *** BEGINNING OF LOOP
20 LET N = N + 1
30 IF INT (N/7) = N/7 THEN PRINT N;
40 IF N < 1000 THEN 20
45 REM *** END OF LOOP
```

In general, how much time can be saved by removing REMarks? Naturally, any reduction in time depends on the number of remarks in the loop and the number of times the loop is executed. But savings also depend on the *length* of the REM statements. Figure 1 illustrates the time required to "execute" short (3 characters) and long (70 characters) REM statements.

In many versions of BASIC, loops can be constructed in just two ways: with GOTO statements, and with

*Want to add more snap to your BASIC programs? This two-part tutorial tells you not only HOW, but HOW MUCH execution time you can save by following our "rules of thumb."*

In this article, we will assume that you wish to write more efficient programs in an interpreted version of BASIC. An important alternative to interpreted BASIC is *compiled* BASIC. A BASIC compiler translates the entire BASIC code into executable machine language *before* the program is RUN, and creates what is called a *run-time module*. If a compiled version of BASIC is used, then a program line translation is carried out only once (during the compilation of the program) so at run-time, no further translation is required. It follows that compiled programs run much more quickly than interpreted programs.

Even faster execution speeds are possible by writing efficient assembly-language programs. For many persons, however, the methods of increasing speed that we present in this article will have to suffice—because they do not have access to a compiled language, and either do not know or do not wish to use an assembly language.

FOR-NEXT statements. (IBM disk BASIC for the PC and cartridge BASIC for the PCjr also provide WHILE loops.) In *Program 1* we used a GOTO statement, but clearly, a FOR-NEXT loop could have been used in the manner listed here:

```
PROGRAM 1.2
20 FOR N = 1 TO 1000
30 IF INT (N/7) = N/7 THEN PRINT N;
40 NEXT N
```

Notice that this program is a bit shorter than *Program 1.1*. It also runs substantially faster. In fact, on the Apple and Commodore machines, this program executes about 36 percent faster. On the TI and IBM computers, the improvement is not as great (25 percent faster on the former, and 19 percent on the latter), but there is still a noticeable difference. Of course, improving program speed sometimes results in a longer and/or less-readable program. However, replacing GOTO loops with FOR loops actually yields shorter, more readable programs. We then have our second rule:

### RULE 2: Whenever possible, use FOR-NEXT loops instead of loops constructed with GOTO statements.

To get a clearer picture of the time differences, it is helpful to compare the execution times of "empty" loops. This measures the time overhead of the loop structure itself. *Loop 1* and *Loop 2* below are the empty loops taken from *Program 1.1* and *Program 1.2*. Figure 2 compares their execution times.

```
Loop 1
10 N = 0
15 N = N + 1
20 IF N<1000 GOTO 15
```

```
Loop 2
10 FOR N = 1 TO 1000
20 NEXT N
```

Using a conditional GOTO when branching is usually faster than using an unconditional GOTO. For example, the next loop uses an unconditional branch in line 30 and requires slightly more execution time. Incidentally, the WHILE loop available in IBM BASIC also runs slightly slower than *Loop 1*, so RULE 2 applies to WHILE loops as well as to loops built with GOTO statements.

```
Loop 3
10 N = N + 1
20 IF N > 1000 GOTO 40
30 GOTO 10
40 REM PROGRAM CONTINUES ON THIS LINE
```

We can speed up execution even more if we omit the N of the NEXT N, to get the loop 4. (TI-99/4A users note: this option is not possible in TI's BASICs.) When the NEXT statement references the variable, the computers take longer to execute the loop.

```
Loop 4
10 FOR N = 1 TO 1000
20 NEXT
```

By not specifying the N variable in *Loop 4*, we eliminate the computer's need to search for a variable, and decrease execution time. The savings are sufficient to justify stating our next rule:

### RULE 3: FOR-NEXT loops execute more quickly if the control variable is not stipulated in the NEXT statement.

If we apply RULE 3 to *Program 1.2*, we get the following program and a 2- to 3-percent speed increase. In a while, we'll see that if we apply this rule to a loop within a more complex program, the time savings could be much greater.

```
Program 1.3
20 FOR N = 1 TO 1000
30 IF INT (N/7) = N/7 THEN PRINT N;
40 NEXT
```

We applied 3 rules, and now have a program which is only three lines long. However, two more improvements can be made! The first of these emerges

**FIGURE 1**

Time (milliseconds) Required to "Execute" One REM Statement.

| | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| Short REM | 0.3 | 0.3 | 0.3 | .5 |
| Long REM | 2.2 | 2.8 | 2.5 | .7 |

**FIGURE 2**

Comparing "Empty" Loops

| | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| Loop 1 | 9.2 | 5.0 | 10.0 | 12.0 |
| Loop 2 | 1.2 | 1.3 | 1.3 | 4.7 |

from the observation that the division in line 30 is performed twice. Perhaps the program will run more quickly if we can find a way to do it just once. But that is easy—just save the result of the first computation! When we do this, we obtain *Program 1.4*.

*Program 1.4*

```
20 FOR N = 1 TO 1000
25 LET CHECK = N/7
30 IF INT (CHECK) = CHECK THEN PRINT N;
40 NEXT
```

On the IBM and TI machines, the speed increase is negligible, but on the other two machines it is a full 10 percent. However, the program is now longer and has an additional variable—we had to trade memory space for time. This exchange of space for time is often required in order to speed up program execution. Deciding whether to make this trade is not always easy, but in any case, we have our next rule:

**RULE 4: If the same computation is performed more than once, execution time can sometimes be shortened by saving the result of the first computation and using this saved result, instead of repeating the computation.**

By applying RULE 4 to *Program 1.3*, we were able to eliminate one computation, N/7, at the expense of adding one assignment statement. Because division is a relatively time-consuming operation on Commodore and Apple computers, we obtained a significant increase in execution speed for those machines. On the IBM PC and TI-99/4A, division computing is more efficient, and our replacement doesn't save much time. RULE 4 works best when it eliminates more than two computations. The replacement of a faster operation, such as "+," should be done with care—some experimentation may be necessary. Some insights along these lines can be obtained by studying Figure 3, which gives the time costs of some common BASIC functions.

**FIGURE 3**

**Time (milliseconds) for Some Common Functions**

| | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| = | 1.2 | 1.6 | 1.3 | 3.9 |
| + | 1.0 | 1.0 | 1.1 | 2.7 |
| – | 1.0 | 1.1 | 1.1 | 2.9 |
| * | 3.1 | 1.3 | 3.2 | 3.0 |
| / | 3.2 | 2.2 | 3.3 | 6.2 |
| ^ | 46.0 | 14.7 | 46.6 | 14.3 |
| ABS | 0.5 | 0.4 | 0.6 | 0.6 |
| ASC | 0.9 | 0.7 | 0.9 | 3.4 |
| CHR$ | 1.3 | 1.2 | 1.4 | 10.2 |
| COS | 24.3 | 23.3 | 25.3 | 72.0 |
| INT | 1.3 | 0.9 | 1.4 | 1.2 |
| LEN | 1.0 | 0.8 | 1.1 | 2.9 |
| LEFT$ | 2.7 | 1.1 | 2.9 | 21.7* |
| MID$ | 4.4 | 1.7 | 4.6 | * |
| PEEK | 1.3 | 0.5 | 4.7 | 12.8 |
| SQR | 49.1 | 6.9 | 47.5 | 62.9 |
| STR$ | 7.2 | 1.9 | 7.3 | 14.2 |
| VAL | 7.2 | 17.8 | 7.6 | 14.7 |

Note: "=" is assignment, "^" is exponentiation.
* TI Extended BASIC does not have a MID$ or LEFT$ function—but the SEG$ command is identical to the MID$ in function and was used here.

The times one obtains in making up such a table are dependent on the operands used. For example, it takes longer to compute "1.23*3.48" than it does to compute "2*3." Another factor which affects the times is whether the operands are constants or variables. The times given above are based on "average" computations, with both operands being variables. For example, the addition problem timed was A + B, where A held the value 3, and B held the value 2.45. Recall the program yielded by RULE 4 above (*Program 1.4*).

In many versions of BASIC, time can be saved by restricting the use of constants such as 7 or 3.14159. To accomplish this, the constants are assigned to variables such as S or PI. Then, each time the constant is needed, the appropriate variable is referenced. This is the idea behind RULE 5, which is the first rule that does not work on all 4 machines (it does not work in IBM BASIC).

**RULE 5: If a constant is used many times in a program, then store the constant as a variable and use the variable instead.**

If we apply this rule to the program above, we get:

*Program 1.5*

```
10 LET S = 7
20 FOR N = 1 TO 1000
25 LET CHECK = N/S
30 IF INT (CHECK) = CHECK THEN PRINT N;
40 NEXT
```

The speed improvement in this case, even for the Apple and Commodore machines, is only 2 percent. However, the savings from this rule can be substantial. For example, if we compare the execution times of the two segments below (see Figure 4), we find that RULE 5 is quite effective on the Apple or Commodore machines, and to a lesser extent on the 99/4A. Both segments compute the area of a circle 1000 times.

Before RULE 5

```
30 FOR N = 1 TO 1000
40 LET AREA = 3.14159*14*14
50 NEXT
```

After RULE 5

```
10 LET PI=3.14159
20 LET R=14
30 FOR N=1 TO 1000
40 LET AREA=PI*R*R
50 NEXT
```

A slightly different application of RULE 5 involves the replacement of a frequently used constant (such as CHR$(13)) by a variable (like RETURN$). After one makes the assignment, RETURN$ = CHR$(13), it is then much faster to access the variable RETURN$ than it is to compute CHR$(13). This application of RULE 5 works even on the IBM PC and PCjr.

At least one additional way to improve our program exists. It involves a rule which is of a slightly different nature than those previously given. It can yield spectacular time savings, but may be difficult or impossible to apply.

**FIGURE 4**

**Execution Time Before and After RULE 5**

| | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| Before | 18.4 | 5.1 | 28.4 | 16.7 |
| After | 6.5 | 5.6 | 6.6 | 13.5 |

**RULE 6: If the usual "speed-up" methods still do not give the level of performance needed, then try to devise a better algorithm.**

In the case of our present programming task, it is possible to devise a much better algorithm. The key observation here is that the numbers which are divisible by 7 are precisely those numbers which are multiples of 7—i.e., the numbers 7, 14, 21 . . . .We can find these numbers by addition and avoid testing the numbers that we know are not multiples. Nearly any version of an algorithm based on this new idea will be significantly faster than our best previous effort. The next program, for example, runs at least 8 seconds faster than *Program 1.1*!

*Program 1.6*

```
10 LET S = 7
15 REM *** START OF LOOP
20 PRINT S;
30 LET S = S + 7
40 IF S < 1000 THEN 20
45 REM *** END OF LOOP
```

In view of the rules given above, it is clear that *Program 1.6* is not the fastest implementation of the new algorithm. It was provided to point out that even a non-optimal version of a superior algorithm is usually much better than any version of an inferior algorithm. The next program is also based on the new algorithm, but it uses a FOR loop with the STEP option, which makes it considerably faster than the GOTO loop above:

*Program 1.7*

```
10 FOR S = 7 TO 1000 STEP 7
20 PRINT S;
30 NEXT
```

It is surprising that as many as 6 run-time reduction rules can be applied to a program as simple as *Program 1*. Figure 5 summarizes the time improvements achieved by applying these 6 rules.

**FIGURE 5**
Execution Times After Application of RULES 1-6.

| Rule | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| Before rules | 22.9 | 21.1 | 24.2 | 39.1 |
| 1 | 19.9 | 15.7 | 21.0 | 37.1 |
| 2 | 12.5 | 11.5 | 13.4 | 29.3 |
| 3 | 12.3 | 11.1 | 13.1 | -.-* |
| 4 | 11.1 | 11.0 | 11.8 | 28.5 |
| 5 | 10.9 | 11.1 | 11.7 | 28.5 |
| 6 | 1.7 | 2.6 | 2.1 | 7.7 |

*Not applicable to TI BASIC or Extended BASIC

**FIGURE 6**
Execution Times for RULE 7

| | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| Before | 53.1 | 7.3 | 53.6 | 22.4 |
| After | 6.6 | 5.6 | 6.6 | 16.4 |

**FIGURE 7**
Execution Times for RULE 8

| | Apple II | IBM | C-64 | TI-XB |
|---|---|---|---|---|
| Before | 6.0 | 4.9 | 6.1 | 14.8 |
| After | 5.7 | 4.5 | 5.8 | 14.4 |

**RULE 7: Replace exponentiations, such as A^2 and A^3 with the corresponding multiplications, A*A and A*A*A.**

This idea may have occurred to you when you noticed in Figure 3 that exponentiation was the most expensive operation time-wise. The programs below illustrate the potential time savings. Both programs compute the area of a circle with a radius of 14 in a loop with 1000 repetitions. See Figure 6 for the rather marked improvement in execution time when RULE 7 is applied.

```
Before RULE 7                 After RULE 7
10 LET PI = 3.14159           10 LET PI = 3.14159
20 LET R = 14                 20 LET R = 14
30 FOR N = 1 TO 1000          30 FOR N = 1 TO 1000
40 LET AREA = PI*R ^ 2        40 LET AREA = PI*R*R
50 NEXT                       50 NEXT
```

**RULE 8: Remove unnecessary parentheses.**

By gaining a clear understanding of operation precedence, it is often possible to remove parentheses which serve no purpose. For example, compare the execution time of the segments below:

```
Before RULE 8                 After RULE 8
10 LET A = 2                  10 LET A = 2
20 LET B = 3                  20 LET B = 3
30 FOR N = 1 TO 1000          30 FOR N = 1 TO 1000
40 LET C = (A*B) - N          40 LET C = A*B - N
50 NEXT                       50 NEXT
```

In this case (see Figure 7), a 5 to 8 percent improvement is achieved, merely by knowing that in the absence of parentheses, multiplications are done before subtractions. There are some machine-specific precedence details, but in general, the order of evaluation is:

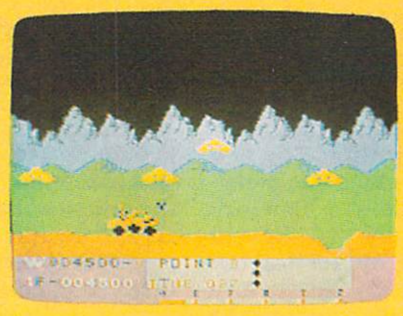
1) Functions, such as ABS, INT, etc.
2) Exponentiation
3) Multiplication and division
4) Addition and subtraction
5) Relational operators, such as < =

**Keep these rules in mind whenever you program, and you'll find that many seconds can be shaved from a program's execution time. Stay tuned for a follow-up in the next issue as we will detail more rules about conditionals and subroutines to aid you in *Speeding Up a BASIC Program*.** HCM

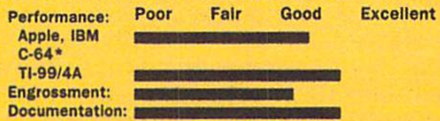
# Moon Patrol

**A Review by Steve Nelson**
*HCM Staff*

## HCM Review



Name: Moon Patrol
Program Type: Arcade game
Machine: TI-99/4A, C-64, Apple, IBM PC
Distributor: Atarisoft
1265 Borregus Ave
Sunnyvale, CA 94066
Price: Cartridge (TI & C-64) or Disk (Apple & IBM) Prices vary under $35
System Requirements: TI-99/4A & C-64 console only. Apple *II*, II+, *II*e or *II*c with disk drive. IBM PC with 128K RAM, color graphics board



* Review copy unavailable from publisher.

***Don't miss this career opportunity with the Lunar City Police Department. Applications now accepted . . .***

*"I remember when I first saw the Lunar City Police Department's ad for replacement gunners on the moon. I was a three-year veteran of the L.A. Police Department (inventory control) and was bored with my unexciting job. When I joined the ranks, I expected excitement, danger, challenges—anything but being stuck behind a desk all day while a jerk like Lieutenant Weedleman threatened me with disciplinary action just because I violated departmental policy number 7438.8 section 574.2 paragraph 16—of the dress code (I had a button missing on my shirt). I lied about my combat experience and caught a freighter the next day. I wasn't sure what I was getting myself into, but I figured that I had nothing to lose—except maybe my life."*

*Moon Patrol*, by Atarisoft, is an arcade style game that puts you in the driver's seat of a Lunar City patrol car. Your assignment is to patrol Sector 9, a dangerous beat if there ever was one.

### Modern Conveniences

Driving a patrol car on the Moon is no picnic, and driving one through Sector 9 is even worse—it's the home of the meanest, nastiest, most ornery thugs in the known galaxy. Fortunately, your patrol car is equipped with all of the latest modern conveniences: anti-gravity buttons, laser cannons, and six-wheel drive to get you over the rough spots—and it can sure get rough out there. Huge craters, moon rocks, patrol-car-eating space plants, rolling boulders, land mines, tanks, and even flying saucers make Sector 9 a real challenge.

***"Driving your patrol car through Sector 9 is extremely dangerous—it's the home of the meanest, nastiest, most ornery thugs in the known galaxy."***

Players may choose from a beginner's level for green recruits, and a championship level for experienced cops. Each course is divided into 26 subsections marked by points A-Z; each of these subsections are grouped together into 5 larger sections. Getting through each region requires all of your driving skills. Both levels are similar in appearance, and are quite difficult as you get near the end of either one. Points are scored each time you jump over a crater, blast or jump over a moon rock, and shoot down enemy saucers and tanks.

### Bombs Away

From the moment you begin your patrol, you are attacked by UFO's piloted by space thugs who indiscriminately drop bombs all over the sector. There are tons of them, and only one of you—so evasive action and a quick trigger finger are necessary. You must shoot down the attacking space ships and dodge the bombs, all the while remaining on the lookout for space craters and moon rocks. Using your patrol car's anti-gravity button (on the joystick), you can jump over meteor craters and other obstacles. Once you make it through the first section, you may be awarded bonus points if your time is fast enough. Then you begin the next section. Each time you advance, the pace of the attacks and the level of difficulty increases.

The game's pace is very fast, and you must maintain absolute control over your patrol car or you will crash into a crater or a moon rock. You can control the car using the keyboard or joysticks. I recommend joysticks because using the keyboard is somewhat awkward—there are 5 keys required to control the car.

The graphics are decent, even though my patrol car's wheels appeared to be designed with Leggo blocks. As you are patroling Sector 9, three "layers" of distant mountains scroll by at different speeds, giving you the illusion of depth, although you can really move in only two dimensions. The attacking space ships are almost too easy to destroy, but there are so many of them that it is still difficult to get through a section without losing several patrol cars. Occasionally, my laser bullet would blast right through an attacking ship with no effect—but I'd just get it on the next pass.

The sound effects are better than average, with a crazy tune that never did get on my nerves as much as a lot of video-game music does. The documentation adequately provides you with the basics of how the game is played (you don't need to know much).

*Moon Patrol* has several features that make it more enjoyable than most run-of-the-mill video games. The game keeps track of the fastest time on a given sector (in memory only), as well as your elapsed time in the sector just completed, and the average time it takes to complete the block—giving you an indication of how your performance shapes up. The game shows one of its friendliest features when you lose a patrol car: You don't have to start all over again at the beginning—you simply begin again at the point where you lost your car.

I tested versions of *Moon Patrol* for the TI-99/4A, the Apple II, and the IBM PC and they were essentially the same, except for the differences in graphics and sound effects—the TI version was the clear winner here.

*Moon Patrol* stacks up as a first rate imitation of the real arcade game. It provides you with a difficult mission, good graphics and sound, and special features that tend to draw you right into the game. Whether you are just looking for a new game with which to pass the time, or, if you are a maniacal video gamer, *Moon Patrol* promises to make life a little more exciting for you. HCM

# HOME COMPUTER™ product news

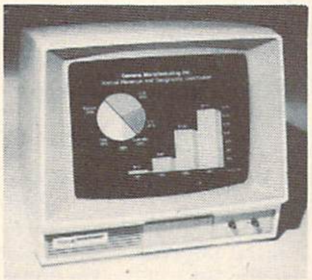
*Each month we publish items of interest and news of recently or soon-to-be released computer products. Our publication of information from manufacturers of computers, peripherals, software, and accessories is not to be construed as product endorsement. Prices quoted are the manufacturers' suggested retail prices and are subject to change.*

*Send press releases to:*

**Product News Editor**
**Home Computer Magazine**
**1500 Valley River Drive., Suite 250**
**Eugene, OR 97401**

## Persistence Pays Off

Phosphor Reduces Screen Flicker



Tecmar, Inc. has introduced a "long persistence," high resolution color monitor for use with IBM PC and PCjr systems. The 640 x 480 resolution and long persistence phosphor is designed to reduce screen flicker. Tecmar's Color Monitor features 16 colors, and costs $795.

**Tecmar, Inc.**
**6225 Cochran Rd.**
**Solon, OH 44139-3377**
**(216) 349-0600**

## Graphx From the Land Down Under

High-Powered Graphics for the 99/4A

Graphx of Sidney, Australia has announced a new graphic design program which bears the company name. Graphx provides MacPaint-like features, such as free-hand drawing, a Zoom option, an easy-to-use color pallete, shape-filling, typewriter mode, and flexible cut-and-paste. Straight lines, circles, and ellipses are also provided for. Drawings—in whole or in part—can be stored on a clipboard, on disk, or even used in assembly language programs as background, etc. Whole collections of often-used shapes, special alphabets and the like can be stored in this fashion and used later. The Clipboard can even be used to create animation. An additional Printer Utility System allows letterhead printing, and giant-sized Poster Prints. Versions are available for Mini-Memory, Editor Assembler, or Extended Basic modules. 32K Expansion and disk drive required. The developer's price is $50.

**Graphx**
**P.O. Box C568**
**Clarence Street**
**Sydney, N.S.W., Australia 2000**

## Brainfood for Apples

Improve Your Memory by Computer

Brainworks has changed its name to Brainpower and released three new software products. Thinkfast, for both the Macintosh and the Apple *II* Family of computers, is a program designed to strengthen one's memory for numeric and verbal information (left brain), and abstract graphic/visual material (right brain). TeleChess, for the Apple *II* Family, offers real-time animated graphics permiting chess play by telephone. Torpedo Run, for the IBM PC, is a simulation game presenting eight historical World War II submarine missions.

**Brainpower Inc.**
**24009 Ventura Blvd.**
**Calabasas, CA. 91302**

## A Light Pen For Kids

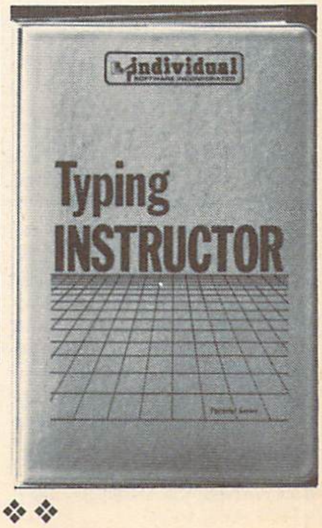
Simplifying the Educational Process

MicroEd introduces a light-pen reading series designed to provide learning-disabled and young students with beginning word skills normally taught in kindergarten through third grade. Initially available for the Commodore 64, the programs use 800 full-screen drawings, and can be purchased in four packages consisting of 14 disks. Packages One and Two have 3 disks and retail for $74.95 each. Packages Three and Four have 4 disks, and retail for $99.95 each.

**MicroEd Incorporated**
**P.O. Box 444005**
**Eden Prarie, MN 55344**
**1-800-MicroEd**

## Three For The IBM PC

New Tutorials on Computer Concepts

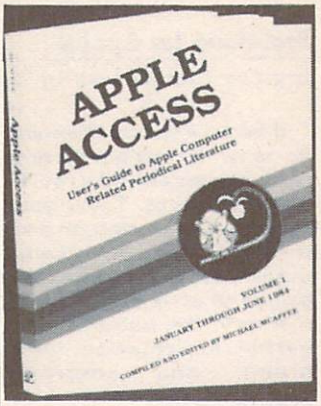


Individual Software Inc. has introduced three new tutorials for the IBM PC and PCjr. They are: a new version of Professor DOS, a program designed to teach the use of DOS 1 and 2; Typing Instructor, which offers interactive training, drill and practice, a fast action game, and word processing concepts; and Professor Pixel, which teaches you graphics, sound, and animation.

**Individual Software Inc.**
**1163-I Chess Dr.**
**Foster City, CA 94404**
**(415) 341-6116**

## Everything About Apples

A User's Guide to Apple Periodicals


APPLE ACCESS
User's Guide to Apple Computer Related Periodical Literature
Apple Access
VOLUME 1
JANUARY THROUGH JUNE 1984
COMPILED AND EDITED BY MICHAEL MCAFFEE

Stony Point Publications has announced the release of a new book, Apple Access—a user's guide to Apple-computer-related periodical literature. Apple Access is the first volume in a series of semi-annual guides directed at documenting most of what is written, and what will be written about Apple computers. Each book will contain more than 9000 entries. In addition, there will be two catch-up volumes covering the period from 1977 through 1982, and another for 1983. The books will retail for $19.95.

**Stony Point Publications**
**Box 4467**
**Petaluma, CA. 94953**

## A New Speaker

Speech Synthesizer with a Loud Voice

Votrax, Inc. has upgraded its line of Type-'N-Talk speech-to-text synthesizers by adding a built-in speaker system. Type-'N-Talk will still retain an external speaker jack. The Type-'N'Talk synthesizer will operate on any personal computer that has an RS-232 serial port. The complete system retails for $265.

**Votrax, Inc**
**1394 Rankin**
**Troy, MI 48083**
**(313) 588-2050**

## A Game Just Like the Movies

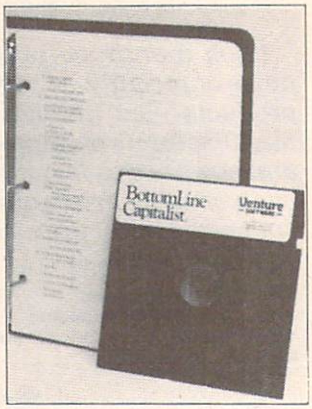
And A Mystery That Teaches Geography

Two new games have been released by Broderbund Software. Where In The World Is Carmen Sandiego? is an adventure mystery game that teaches users about world geography with the help of a 1985 World Almanac and Book of Facts that accompanys the program. The mystery scenario varies in every game because there are 10 different villains and 30 different countries. It requires an Apple *II* Family computer with at least 64K, and it retails for $39.95. Karateka was designed to emulate an action-packed movie. It has a story line, characters, animation, and sound, some of which was recorded and transfered to disk. In the game, a young karate master must use his karate skills to fight an evil warlord to rescue his bride-to-be. Karateka is available for Apple *II* Family ($34.95) and Commodore 64 ($29.95) computers.

**Broderbund Software**
**17 Paul Dr.**
**San Rafael, CA 94903-2101**
**(415) 479-1170**

## Capitalism: the Bottomline

Market Strategy Taught By Game



Bottomline Capitalist, a business-simulation program for IBM PC and Apple computers, challenges players or teams to compete as entrepreneurs in the marketplace. Through 40 quarters of business activity, players make decisions on budgeting, raising capital, marketing, developing managers and staff, raising capital, and more. The goal is to foster growth of the business enterprise, increase profitability, and maintain or gain marketshare. Bottomline Capitalist retails for $129.95.

**Venture Software**
**16200 Ventura Blvd.**
**Encino, CA 91436**
**(818) 986-4110**

## A Synphonix of Words

A Real Speech Synthesizer for Junior

SynPhonix jr is a speech synthesizer board by Artic Technologies that plugs into the internal modem slot of the IBM PCjr. It includes an on-board speaker and an external speaker jack. In addition, the Sonix jr Speech Operating System and the Text-to-Speech jr translation program come with SynPhonix jr. This support software allows users to generate vocabulary from phonemes or strings of English text. Unlimited speech, music, and sound effects may then be incorporated into user-written programs with BASIC statements. The SynPhonix jr retails for $175.

**Artic Technologies**
**2234 Star Court**
**Auburn Heights, MI 48057**
**(313) 852-8344**

## Teenagers in Space

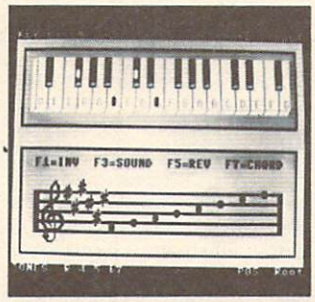
14 Year-Old Writes TI Game in Forth

Spacestation Pheta is a complex, graphics-filled game available for the TI-99/4A from T & T Software, a small company headed by 14 year-old Tad Woods. Written in Wycove Forth, this new game features 80 different playing screens. A built-in editor allows the user to design and add an unlimited number of original screens. Players explore a mysterious abandoned space station with a limited supply of oxygen, encountering strange objects that can be used to survive. High scores can be saved on disk, and the editor also allows flexible playing speeds and starting points. A disk controller and drive, 32K Memory Expansion, and one of 4 modules—Extended Basic, Mini Memory, Editor Assembler, or TI-Writer—are required. Spacestation Pheta is available directly from the company. Price is $14.95.

**T ¿ T Software**
**109 Tee Circle**
**Salem, VA 24153**

## Music Making to Scale

Computer Drills Teach Music Theory



The Keyboard Chord/Scale Master by Valhala Software is designed to enhance the user's keyboard abilities through sight and sound reinforcements in its chord and scale display, review, and compare modes. Adaptable to color or monochrome displays, it offers drills on chords or key signatures, quizzes, and a music game. Available in disk media for the Commodore 64, Keyboard Chord/Scale Master is $39.95.

Valhala Software
205 E. Hazelhurst
Ferndale, MI 48220

❖❖❖❖

## Mac Gets a New Desk

Desk Organizer Expands Mac Office

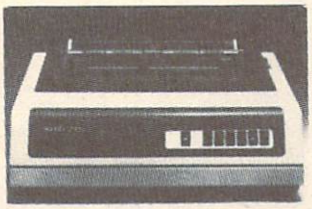
The Desk Organizer from Warner Software, Inc. is now available for the Apple Macintosh. This program is billed as a complete desk management system—and, in this case, "a perfect hardware/software marriage." Desk Organizer runs on both models of the Macintosh, with the 512K Mac able to switch instantly between this program and other Macintosh Software. Features include a full set of "desk tools" to organize notes, letters, and Rolodex information—as well as an expanded Mac notepad, automatic modem dialing, cross-referencing, time-management, calculating, printing, and a sophisticated filing and correspondence system. It retails for $99.

Warner Software
666 Fifth Ave.
New York, NY 10103
(212) 484-3070

❖❖❖❖

## Printer Has Instant IBM-Compatibility

Fast, and Letter Quality



Okidata's Pacemark 2410 printer is now available for instant interface with IBM PC and PC-compatible computers. It has graphics ability, letter-quality printing, two-color printing, and a choice of three print modes. It offers full access to most software for the IBM PC, including graphics packages and spread sheets. The printer is available with a Centronics-compatible parallel interface for $2395, and an RS-232C interface for $2475.

Okidata
532 Fellowship Rd.
Mt. Laurel, NJ 08054
(609) 235-2600

❖❖❖❖

## Another Language Comes Forth

The Fourth Forth is for Mac

MicroMotion has announced the release of MasterForth for the Macintosh computer. It is a state-of-the-art implementation of the Forth programming language and exactly matches the Forth-83 standard dialect described in the book Mastering Forth, which is included in the package. The package provides a complete programming environment for the Macintosh, including a 68000 macro-assembler, support for the mouse, finders, menus, and a graphics toolbox. MasterForth is also available for the IBM PC, the Apple II Family, and the Commodore 64. It retails for $125.

MicroMotion
12077 Wilshire Blvd. *506
Los Angeles, CA 90025

❖❖❖❖

## Call of the Wild

Simulating Survival by Computer

Wilderness: A Survival Adventure combines "the realism of a simulation with the excitement of an adventure game." Developed by Eduware, a division of Peachtree Software, the program recreates a wilderness survival experience set in the Sierra Nevada mountains, and an adventure for a lost city of gold. Three-dimensional graphics and data bases dealing with the weather, topography, animal and plant life, and health-related situations for each region are featured in this program, and in separate disks that explore Bolivia, Burma, British Columbia, Chile, or New Guinea. Wilderness: A Survival Adventure is available for Apple II Family computers at a retail price of $49.95.

Eduware
3445 Peachtree Rd. N.E.
Atlanta, GA 30326-1276
(404) 239-3000

❖❖❖❖

## Digitize Your Monitor

Grab Video Signals Off the Air

I/O Video announces Mac Private Eye, a video digitizer that converts video signals into high-resolution images using the Apple Macintosh computer. Mac Private Eye accepts images from any standard NTSC video source, and unlike conventional video digitizers, works with moving images, capturing a complete frame in 1/30th of a second. Each video frame is converted into a 512 x 512 pixel image, viewable on the Mac with a moveable window. The image can be stored on disk, and manipulated with MacPaint. Mac Private Eye retails for $595.

IÒ Video, Inc.
222 Third St.
Cambridge, MA 02142
(617) 547-4141

❖❖❖❖

## Pull-Down Menu Magic

Four New Kids On the TI Block

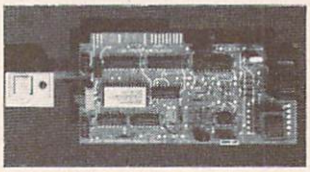
Asgard Software announces the release of four new programs for the TI-99/4A computer. The products are: an easy-to-use schedule maker with pull-down menus and 20 different menu options for a variety of tasks in the home or business; a stamp-manager program for philatelists, also featuring pull-down menus and a printer option; a music synthesizer designed with the musical novice in mind, that allows users to create and merge compositions; and an arcade-style game, Balloon War. All four programs require Extended BASIC.

**Asgard Software**
**P.O. Box 10306**
**Rockville, MD 20850**

## More Modems Make Market

Data Exchange Systems For Apple II

Telstar Communications has released its Networker Modem along with its Zoom/Modem *IIe* for Apple II, II +, and IIe systems. The Networker Starter Set ($99) includes a plug-in 300-baud modem board and software to store conversations to disk in DOS or ProDOS format. Networker with advanced software ($149) adds an advanced text editor, and the ability to transfer Apple text or ASCII files to other computers over the phone. The Zoom/Modem IIe ($149) has all the features of the Networker, plus auto-dial, auto-answer, monitor speaker, and Hayes Micromodem IIe compatibility. The Zoom/Modem IIe is also available with advanced software for $199.



**Telstar Communications**
**145 Lincoln St.**
**Boston, MA 02111**
**1-800-344-3311**

## Invest to Learn, Learn to Invest

College Subjects, Computer-Taught

Useful Software has packaged a number of programs of interest to college students and investors onto disks for the Commodore 64. The College Pak contains more than 20 programs for computer-aided instruction in college-level math, physics, chemistry, engineering, language, history, and medicine. The Investors Pak contains more than 20 business and investment programs in Real Estate, Mortgages, Bonds, Loans, Syndications, Leases, Shelters, and other investment analysis subjects. The College Pak and The Investors Pak are $25 each.

**Useful Software Co.**
**P.O. Box 54-H**
**Scarsdale NY 10583**
**(914) 633-8427**

## Keeping Track and Keeping Fit

Two Products Aid Weight Loss

Journal and Belly Buddy for Weight Loss are the two latest releases from New Found Software for Apple *II* Family machines. Journal allows users to daily record thoughts and ideas as in a diary, or as an organizer. It includes a perpetual calendar and an appointment calendar. Journal is password but not copy-protected for personal use. It costs $15.95. Belly Buddy for Weight Loss computes calories, carbohydrate grams, protein, and 13 other nutrients for 425 food items. Users have the option to add more, and can track their food intake, set weight goals, and keep a 5-year weigh-in history. It costs $28.95.

**New Found Software**
**9040 Aileen Dr.**
**Mentor, OH 44060**

## Starting Smart, Staying Smart

Utility Speeds Up C-64

Muse Software has announced the release of Smart Start, a program for the Commodore 64 that generates BASIC code automatically. It is designed for use with a disk drive to make writing, saving, loading, and running programs as simple as moving the cursor and pushing a button. Smart Start can be used immediately by computer neophytes to create and save pictures, music and sound effects programs, and more. It retails for $39.95.

**Muse Software**
**347 N. Charles St.**
**Baltimore, MD 21201**
**(301) 659-7212**

## Expense Accounting

Program Tracks Outflow of Money>

Sapana Micro Software has announced the release of its Expense-Track II, a personal and small-business accounting program to store and retrieve expense information, and print checks. It runs on the IBM PC and PCjr. The program accommodates 255 main expense categories and 255 subcategories, and can sort by month, day, year, and category. A calculator mode is included for numerical values, and summary reports can be produced in bar graph and table form. Expense-Track II is priced at $69.95.

**Sapana Micro Software**
**1305 S. Rouse**
**Pittsburg, KS 66762**
**(316) 231-5023**

## More Help For River City

Musicman Composes on IBM, Apple

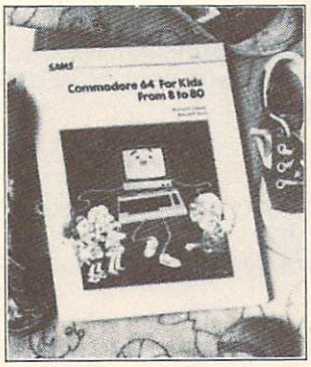
A new program from Zepher Services allows the user to compose and play music on the IBM PC/XT/PCjr or Apple *II* Family computers. With Musicman, you can compose music on screen using standard musical methods. Compositions can be saved on disk for later replay or revision. Sample music is also provided with the program. A musical staff of 5 lines appears on the screen where you can place whole, half, quarter, eighth, or sixteenth notes, or rests. Then, you can set a desired key and any accidental sharps or flats. Notes can be dotted for extended duration. Once the composition is done, you can change the tempo or pitch to test the variations. Legato can also be varied. Musicman is available for $29.95 (plus $2.00 shipping) directly from the publisher.

**Zepher Services**
**306 South Homewood Ave.**
**Pittsburgh, PA 15208**
**(412) 247-5915**

❖❖❖❖

## C-64 BASIC Made Easy

Step-by-Step Guide to Programming



Howard W. Sams & Co. is releasing a new addition to their popular 8-to-80 series entitled Commodore 64 for Kids 8 to 80. The book is designed to work hands-on to create short BASIC programs in Commodore BASIC. Commodore 64 for Kids 8 to 80 is fully illustrated, and includes exercizes and review material useful to both beginning and experienced programmers. The content of the book is based on concepts used at National Computer Camps by the authors.

**Howard W. Sams ¿ Co.**
**4300 W. 62nd St.**
**Indianapolis, IN 46268**
**(317) 298-5400**

❖❖❖❖

## Financial Planning the TI Way

Key-in Business Application Programs

The Chilton Company has released the book Financial Analysis on TI Computers. The book contains 18 chapters on various topics of financial interest. Each chapter contains a brief explanation of the analysis technique, example applications, and program listings. All programs are written in TI BASIC. Specific topics include financial ratio analysis, cost-volume-profit determination, inventory-level analysis, and many more. The book is also available for the IBM PC.

**Chilton Book Company**
**Radnor, PA**
**1-800-345-1214**

❖❖❖❖

## When it Rains—Have Fun

10 Programs Especially for Kids

Rainy Day Fun
GAMES, PUZZLES AND FUN ACTIVITIES TO PLAY, PRINT OUT AND SHARE
NEW! Ten Creative Activities for Kids

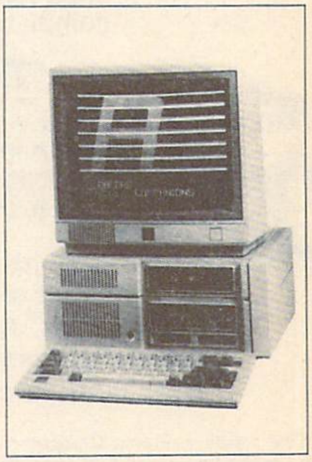
Thorn EMI Compter Software Inc. has announced the first in its new Computer Clubhouse Series of software products available for the Apple II Family of computers and the Commodore 64. Rainy Day Fun programs are designed for children aged 5 through 9, and are educational in nature. Ten programs on the disk use icons to lead children through the menus. Rainy Day Fun retails for $39.95.

**Thorn EMI**
**Computer Software Inc.**
**3187C Airway Ave.**
**Costa Mesa, CA 92626**

❖❖❖❖

## Little Brother's Growing Up

PCjr Gets More Memory



Racore Corp. has introduced two new products for the IBM PCjr that give it many capabilities of the IBM PC and PC AT and XT. The new products are a 1.2MB floppy drive called the Racore ATcessory, and a network product, Racore-Net, enabling the PCjr to communicate with the IBM PC AT. The floppy disk drive includes a parallel printer port, a clock calendar, a direct memory access drive controller, and a PC/PCjr mode switch. ATcessory makes the PCjr even more PC-compatible, allowing it to run Framework, Wordstar 2000, and other software previously not usable on the PCjr. Racore-Net allows up to 16 PCjrs, PCs, XTs, and ATs to be linked together. ATcessory sells for $895, and Racor-Net retails for $200.

**Racore Corp.**
**10 Victor Square**
**Scotts Valley, CA 95066**
**(408) 438-7255**

❖❖❖❖

**[Special Note: We are hungry for information on TI-related products for inclusion in HCM Product News! Product developers for the TI-99/4A, please send us your press releases.]**

# Want to Get Published?
# Fame, Fortune, Recognition! See Your Name in Print!

Home Computer Magazine is looking for articles and programs in all areas of interest relevant to Apple, IBM, Commodore, and Texas Instruments home computers. Here are some of the kinds of material we would like you to submit:

## Software

Have you written any programs in the areas of home productivity, education, or entertainment? Perhaps you've created unique software to help monitor personal finances, or a new contribution to computer-assisted instruction (CAI). Maybe you have an unusual new game—or a routine that makes certain computer operations easier to perform. Don't be shy. Even if you think your piece is "unpolished," it may still be a good idea. We will be glad to follow through with your concept—enhancing the program and converting it to work on the other machines we cover.

## Product Reviews

Have you recently purchased a piece of hardware or software that hasn't come up to your expectations—or has, on the other hand, impressed you with its performance? We're looking for comprehensive product reviews from different perspectives.

## Hardware Tips

Perhaps you've modified your microcomputer or have interfaced it with some unique or useful hardware. Send us your how-to-do-it story, complete with photos and/or diagrams.

## Tutorials

Many of our articles are purely instructional. If you have extensive experience in some area of programming or other computer application, put your specialized knowledge down on paper and let us pass it on to our readers.

These are just some ideas. Perhaps you have others. If you're not a professional writer, don't worry. Our friendly editorial staff stands ready to help you polish your manuscripts. And we'll be more than happy to send you a copy of our author guidelines. Here are some comments from happy writers who have already published their work in our magazine:

*"The people at Home Computer Magazine are fun to work with. And it's sure nice to get paid for writing about my favorite subject."*
*—Patricia Swift*
Author of "Multiplan Medium" and other articles

*"The artwork and layout are creative and contribute a lot to the presentation of my articles."*
*—Roger Kirchner*
Author of "Missionary Impossible" and other articles

*"It was gratifying to finally see my name in print after all the work I've done on my computer."*
*—Brian Lee*
Author of "Market Madness"

*"I was extremely impressed with the way my program was printed in HCM. It was very interesting to see the way the program was translated into the languages of the other popular computers and to read the comments of the people who reviewed the program. Truly a first class job! Thank you!"*
*—Craig Blazakis*
Author of "Bird Brain"

*"I was very pleased with the final presentation of my article. It is gratifying to see such judicious handling of an outside submission. The HCM staff fixed a program bug and expanded the application of the article to three other computers, while preserving the style of the article as submitted. The illustration added to the overall readability."*
*—Andrew Keith*
Author of "Build a LOGO Adventure"

Please send your double-spaced, typed or printed manuscripts, photos, and disks or cassettes (recorded on both sides) if the article includes program material, to:

**Attn: Editorial Submissions**
**Home Computer Magazine**
1500 Valley River Drive, Suite 250
Eugene, OR 97401

# HOME COMPUTER™
## PROGRAM LISTINGS

PROGRAM LISTING

## CONTENTS

# Your Guide to Typing in Programs from *HCM*

Within these pages is a software bonanza: entertainment, education, home and business applications, utilities, and tutorials—just for you. All you need to do is type them into your computer. *HCM* has taken most of the strain out of this process:

- Typeset listings with numbers in boldface.
- A bold, double vertical bar separating the line numbers from the program statements in BASIC listings.
- A vertical background grid to aid entry of the spaces.

Look at the Key-in-Reference (Figure 1 below) see how each character actually appears in the listing. By checking any questionable characters with the Key-in Reference, you can reduce errors to a minimum.

## Figure 1: Key-in Reference

```
      1 0 0  R E M     1 2 3 4 5 6 7 8 9 0 ! ? @ # $ % & * ( ) { } + = — / : ; , .
             A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [ ] _ | ☛☚  < > ^
             a b c d e f g h i j k l m n o p q r s t u v w x y z ` \ ~ ← ↑ £ π ' "
```

## Before You Begin

Since *HCM* publishes for several different computers, the first thing you should do is make sure that you are looking at the listing designed for your machine. If, for example, you have an Apple IIe, make sure you look for the following black bar above the listing: APPLE II Family The computer model name will likewise appear on each subsequent page of each listing, so always look for the name before you begin typing from a new page of listings.

Before you begin typing in the program, you will want to set up a system to save your program. Whether you are using a cassette or diskette storage system, now is the time to be certain it is properly connected, powered up, and loaded with a blank cassette or an initialized disk. As you type in your program, you should get in the habit of saving your work after every twenty or so lines.

One of the most common errors in entering a listing is typing one symbol for another. These transpositions include substituting the letter O for the number 0, the letter l for the number 1, the letter S for the $, and the uppercase B for the number 8. The last error is especially likely when working in hexadecimal numbers which are composed of 0-9 and the uppercase letter A-F.

The listings in *HCM* are always the same number of characters wide, but the number of characters put on any line of the video display will vary from computer to computer. Don't try to make your listings *look like* the type-set listing—instead make sure you key in the listings character for character and space for space.

## A Special Note on C-64 Listings

Commodore uses more than 90 special symbols to represent various keyboard operations: for instance, the symbol ♥ in a program represents the operation of holding down the [SHIFT] key and pressing the key which has CLR on its upper half (second key from the right on the top row). This operation clears the screen.

Rather than reproducing these symbols, *HCM's* listings include key-stroke instructions, between two hands with pointing fingers. For example, when you find ☛SHIFT CLR☚ in an *HCM* listing, you will know to hold down the [SHIFT] key and press the key with CLR on it.

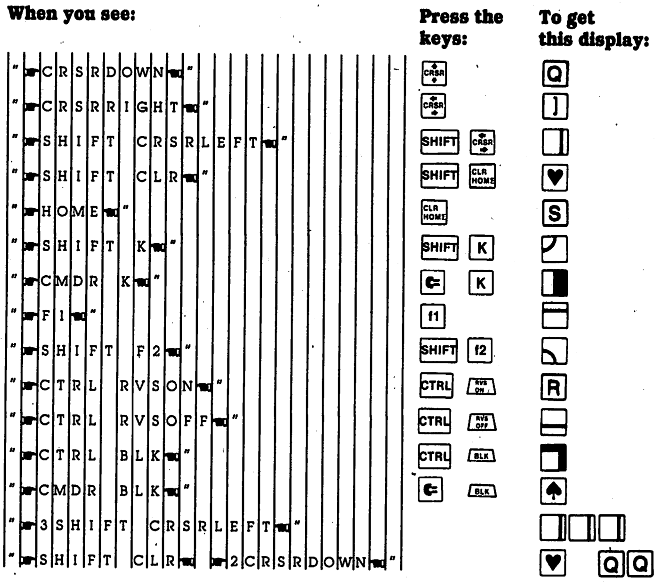
A number is included if you need to repeat the operation: ☛8SHIFT CRSRLEFT☚ tells you to hold the [SHIFT] key down and press the cursor left key (on the bottom right of the keyboard) eight times.

When you come to the hand symbols, remember:

- Each operation is enclosed in its own set of hand symbols.
- If any key action requires you to press two keys, press the control key or the Commodore key or the shift key *first* and hold it down before pressing the second key.
- Everything between a pair of hand symbols is set in a different place.

In Figure 2 below, we have included a chart showing you a representative sample of the symbols that appear when you use keystrokes enclosed by the hand symbols. (Notice that the hand symbols always appear within quotation marks—as in a print statement.)

## Figure 2: C-64 Special symbols



| When you see: | Press the keys: | To get this display: |
|---|---|---|
| "☛CRSRDOWN☚" | CRSR | Q |
| "☛CRSRRIGHT☚" | CRSR | ] |
| "☛SHIFT CRSRLEFT☚" | SHIFT CRSR | ▕ |
| "☛SHIFT CLR☚" | SHIFT CLR HOME | ♥ |
| "☛HOME☚" | CLR HOME | S |
| "☛SHIFT K☚" | SHIFT K | ◜ |
| "☛CMDR K☚" | C= K | ▐ |
| "☛F1☚" | f1 | ▔ |
| "☛SHIFT F2☚" | SHIFT f2 | ◟ |
| "☛CTRL RVSON☚" | CTRL RVS ON | R |
| "☛CTRL RVSOFF☚" | CTRL RVS OFF | ▁ |
| "☛CTRL BLK☚" | CTRL BLK | ▜ |
| "☛CMDR BLK☚" | C= BLK | ♠ |
| "☛3SHIFT CRSRLEFT☚" | | ▕▕▕ |
| "☛SHIFT CLR☚ ☛2CRSRDOWN☚" | | ♥ QQ |

## Program Identification

Each program header (the first few lines of the program) contains information giving the language the program is written in (e.g., TI Extended BASIC, Applesoft, etc.) and any special system components that are required (special memory cards, Speech Synthesizer, etc.). The first two digits of the version number tell you in which volume and issue of *HCM* the program *initially* appeared. The third digit of the version number indicates the version of the program. When a program initially appears, in *HCM*, it is version 1. Any subsequent revisions to the program if later published in the magazine or in the software available on magnetic medium from *HCM* will bear a revised version number.

5 . 3 . 1

**Volume no.**
**Issue**
**Version**
**1 = original program**
**2 ⋮ n } = no. of update**

**End of Program = HCM**

# BUDGETRON

```
100 REM  *************
110 REM  * BUDGETRON *
120 REM  *************
130 REM  COPYRIGHT 1985
140 REM  EMERALD VALLEY PUBLISHING CO
150 REM  BY SCOTT WILLIAMS
160 REM  AND THE HCM STAFF
170 REM  HOME COMPUTER MAGAZINE
180 REM  VERSION 5.3.1
190 REM  APPLE II FAMILY APPLESOFT
200 REM
210 PD = 0: IF PEEK (48905) = 76 AND
    PEEK (48911) = 0 THEN PD = 1
220 ONERR GOTO 4430
230 GOSUB 380
240 GOSUB 530
250 IF IN$ = "5" OR IN$ = ESC$ THEN 280
260 ON VAL (IN$) GOSUB 630,730,970,111
    0
270 GOTO 240
280 HOME
290 VTAB 10: HTAB 1
300 PRINT "ARE YOU SURE YOU WISH TO END
     THE"
310 PRINT "PROGRAM AND DESTROY ALL DATA
     IN"
320 PRINT "MEMORY?  (Y/N)"
330 VC = 12:HC = 18: GOSUB 4370
340 IF IN$ < > "Y" THEN 240
350 HOME
360 END
370 REM  INITIALIZATION
380 ESC$ = CHR$ (27):LF$ = CHR$ (8):R
    T$ = CHR$ (21):CR$ = CHR$ (13):A$
     = "A":Z$ = "Z":BL$ = CHR$ (7):CM$
     = ","
390 DIM ET$(30)
400 MC = 16
410 MD = 100
420 DIM CT$(MC): DIM NI(MC)
430 CT$(1) = "$$  INCOME  $$  "
440 DIM BD(MC): DIM EX(MC)
450 NC = 1
460 DIM BI$(MD)
470 BK$ = "                    "
480 FOR IT = 1 TO MD:BI$(IT) = BK$: NEX
    T
490 FOR IT = 1 TO MC:NI(IT) = 0: NEXT
500 NR = 0:NC = 0
510 RETURN
520 REM  MAIN MENU
530 HOME
540 VTAB 1: HTAB 12: PRINT "BUDGETRON"
550 VTAB 4: HTAB 5: PRINT "1. WORK WITH
     BUDGET"
560 VTAB 6: HTAB 5: PRINT "2. REPORTS".
570 VTAB 8: HTAB 5: PRINT "3. BAR CHART
    "
580 VTAB 10: HTAB 5: PRINT "4. FILES"
590 VTAB 12: HTAB 5: PRINT "5. EXIT THE
     PROGRAM"
600 UL$ = "5": GOSUB 4330
610 RETURN
620 REM  WORK WITH BUDGET
630 HOME
640 VTAB 1: HTAB 10: PRINT "WORK WITH B
    UDGET"
650 VTAB 6: HTAB 5: PRINT "1. BUDGET CA
    TEGORIES"
660 VTAB 9: HTAB 5: PRINT "2. INCOME AN
    D EXPENSES"
670 VTAB 12: HTAB 5: PRINT "3. EXIT"
680 UL$ = "3": GOSUB 4330
690 IF IN$ = "3" OR IN$ = ESC$ THEN RE
    TURN
700 ON VAL (IN$) GOSUB 1560,1680: GOTO
     630
710 RETURN
720 REM  REPORTS
730 HOME : VTAB 1: HTAB 10: PRINT "REPO
    RTS"
740 FOR IT = 1 TO 4: ON IT GOSUB 910,92
    0,930,940: NEXT
750 UL$ = "4": GOSUB 4330
760 IF IN$ = ESC$ OR IN$ = "4" THEN RE
    TURN
770 INVERSE : ON VAL (IN$) GOSUB 910,9
    20,930: NORMAL : FOR DI = 1 TO 1000
    : NEXT
780 RX = VAL (IN$)
790 HOME : VTAB 1: HTAB 10: PRINT "REPO
    RTS"
800 VTAB 6: HTAB 5: PRINT "1.  PRINT TO
     SCREEN"
810 VTAB 10: HTAB 5: PRINT "2.  PRINT T
    O PRINTER"
820 UL$ = "2": GOSUB 4330
830 IF IN$ = ESC$ THEN 730
840 PX = VAL (IN$)
850 SF = 60 - 42 * (PX - 1): GOSUB 4110
860 IF PX = 2 THEN GOSUB 4240
870 ON RX GOTO 880,890,900
880 GOSUB 3170: GOSUB 3480: GOTO 730
890 GOSUB 3480: GOTO 730
900 GOSUB 3350: GOSUB 3480: GOTO 730
910 VTAB 5: HTAB 4: PRINT " 1. ALL CATE
    GORIES AND ITEMS": RETURN
920 VTAB 7: HTAB 4: PRINT " 2. MONTHLY
    SUMMARY ": RETURN
930 VTAB 9: HTAB 4: PRINT " 3. PRINT CH
    ART ": RETURN
940 VTAB 11: HTAB 4: PRINT " 4. EXIT ":
     RETURN
950 RETURN
960 REM  BAR CHART
970 IF NC < 2 THEN RETURN
980 SF = 38: GOSUB 4110
990 HOME :CP = 0
1000 FOR IT = 2 TO NC
1010 PRINT CT$(IT): PRINT ""
1020 FOR JI = 1 TO INT (BD(IT) * SC + .
     5001): PRINT "B";: NEXT JI: PRINT "
     "
1030 C$ = "E": IF EX(IT) > BD(IT) THEN C
     $ = "*": INVERSE
1040 FOR JI = 1 TO INT (EX(IT) * SC + .
     5001): PRINT C$;: NEXT JI: NORMAL :
      PRINT "": PRINT ""
1050 PRINT ""
1060 CP = CP + 1: IF CP = 3 THEN GOSUB
     3640
1070 NEXT IT
1080 IF CP > 0 THEN GOSUB 3640
1090 RETURN
1100 REM  FILES
1110 HOME
1120 VTAB 1: HTAB 10: PRINT "FILES"
1130 VTAB 6: HTAB 5: PRINT "1. LOAD DATA
     "
1140 VTAB 9: HTAB 5: PRINT "2. SAVE DATA
     "
1150 VTAB 12: HTAB 5: PRINT "3. EXIT"
1160 UL$ = "3": GOSUB 4330
1170 IF IN$ = ESC$ OR IN$ = "3" THEN RE
     TURN
1180 CH = VAL (IN$)
1190 HOME : VTAB 1: HTAB 10: ON CH GOTO
     1200,1210
1200 PRINT "LOAD DATA FROM A FILE": GOTO
      1220
1210 PRINT "SAVE DATA TO A FILE"
1220 VTAB 6: PRINT "INPUT FILE NAME:": G
     OSUB 4790
1230 IF T$ = ESC$ OR T$ = "" THEN RETUR
     N
1240 GOSUB 4260
1250 FL$ = T$: HOME
1260 VTAB 10: HTAB 1
1270 ON CH GOTO 1280,1290
1280 PRINT "LOADING ";: GOTO 1300
1290 PRINT "SAVING ";
1300 PRINT FL$
1310 PRINT ""
1320 PRINT CHR$ (4)"OPEN ";FL$",D";DR$
1330 IF CH = 2 THEN 1430
1340 PRINT CHR$ (4)"READ "FL$
1350 INPUT X$:NC = VAL (X$)
1360 FOR IT = 1 TO NC: INPUT CT$(IT): NE
     XT
1370 FOR IT = 1 TO NC: INPUT X$:BD(IT) =
       VAL (X$): NEXT
1380 INPUT X$:NR = VAL (X$)
1390 IF NR = 0 THEN 1420
1400 FOR IT = 1 TO NR: INPUT BI$(IT): IN
     PUT X$:NI(IT) = VAL (X$): NEXT
1410 PRINT CHR$ (4)"CLOSE"
1420 RETURN
1430 PRINT CHR$ (4)"CLOSE "FL$
1440 PRINT CHR$ (4)"DELETE "FL$
1450 PRINT CHR$ (4)"OPEN "FL$
1460 PRINT CHR$ (4)"WRITE "FL$
1470 PRINT STR$ (NC)
1480 FOR IT = 1 TO NC: PRINT CT$(IT): NE
     XT
1490 FOR IT = 1 TO NC: PRINT STR$ (BD(I
     T)): NEXT
1500 PRINT STR$ (NR)
1510 IF NR = 0 THEN 1530
1520 FOR IT = 1 TO NR: PRINT BI$(IT): PR
     INT STR$ (NI(IT)): NEXT
1530 PRINT CHR$ (4)"CLOSE"
1540 RETURN
1550 REM  BUDGET CATEGORIES
1560 HOME
1570 VTAB 1: HTAB 10: PRINT "BUDGET CATE
     GORIES"
1580 VTAB 5: HTAB 5: PRINT "1. INITIALIZ
     E BUDGET"
1590 VTAB 7: HTAB 5: PRINT "2. ADD A CAT
     EGORY"
1600 VTAB 9: HTAB 5: PRINT "3. CHANGE A
     CATEGORY"
1610 VTAB 11: HTAB 5: PRINT "4. EXIT"
1620 UL$ = "4": GOSUB 4330
1630 IF IN$ = "4" OR IN$ = ESC$ THEN RE
     TURN
1640 ON VAL (IN$) GOSUB 1800,1920,2180
1650 GOTO 1560
1660 RETURN
1670 REM  INCOME AND EXPENSES
```

Continued

```
1680 HOME
1690 VTAB 1: HTAB 10: PRINT "INCOME AND EXPENSES"
1700 VTAB 5: HTAB 5: PRINT "1. INITIALIZE ITEMS"
1710 VTAB 7: HTAB 5: PRINT "2. ADD ITEM"
1720 VTAB 9: HTAB 5: PRINT "3. CHANGE ITEM"
1730 VTAB 11: HTAB 5: PRINT "4. EXIT"
1740 UL$ = "4": GOSUB 4330
1750 IF IN$ = "4" OR IN$ = ESC$ THEN RETURN
1760 ON VAL (IN$) GOSUB 2420,2530,2750
1770 GOTO 1680
1780 RETURN
1790 REM    INITIALIZE THE BUDGET
1800 HOME
1810 VTAB 5: HTAB 5: PRINT "INITIALIZE THE BUDGET:"
1820 VTAB 10: HTAB 1
1830 PRINT "ARE YOU SURE?? THIS WILL ERASE"
1840 PRINT "ALL BUDGET DATA (Y/N)"
1850 VC = 11:HC = 23: GOSUB 4370
1860 IF IN$ < > "Y" THEN RETURN
1870 FOR IT = 2 TO MC:CT$(IT) = "                    "
1880 BD(IT) = 0:EX(IT) = 0: NEXT IT
1890 BD(1) = 0:NC = 1
1900 RETURN
1910 REM      ADD A CATEGORY
1920 HOME
1930 VTAB 1: HTAB 10
1940 PRINT "ADD A CATEGORY"
1950 IF NC < MC THEN 1990
1960 VTAB 10: HTAB 1: PRINT "CATEGORIES ARE FULL."
1970 FOR DI = 1 TO 1000: NEXT
1980 RETURN
1990 IF CT$(NC + 1) = "" THEN CT$(NC + 1) = "                "
2000 CN$ = CT$(NC + 1)
2010 GOSUB 3700
2020 VTAB 21: HTAB 1: PRINT "CATEGORY #" STR$ (NC + 1)":";
2030 IF NC = 0 THEN VTAB 21: HTAB 16: PRINT CT$(1)
2040 VTAB 23: HTAB 1: PRINT "ENTER BUDGET AMOUNT:"
2050 IF NC = 0 THEN 2090
2060 HB = 15: GOSUB 3960
2070 IF IN$ = ESC$ THEN RETURN
2080 CT$(NC + 1) = CN$
2090 CN$ = LEFT$ ( STR$ (BD(NC + 1)) + "       ",6)
2100 N1 = 23:N2 = 22:N3 = 6: GOSUB 3800
2110 IF IN$ = ESC$ THEN RETURN
2120 BD(NC + 1) = CX:NC = NC + 1
2130 VTAB 19: HTAB 1: CALL - 868: PRINT "ANOTHER CATEGORY? (Y/N)"
2140 VC = 19:HC = 25: GOSUB 4370
2150 IF IN$ = "Y" THEN VTAB 19: HTAB 1: CALL - 868: GOTO 1920
2160 RETURN
2170 REM    CHANGE A CATEGORY
2180 HOME
2190 VTAB 1: HTAB 10
2200 PRINT "CHANGE A CATEGORY"
2210 GOSUB 3700
2220 VTAB 19: HTAB 1: PRINT "ENTER A CATEGORY (1 - " STR$ (NC)"):"
2230 CN$ = "1 "
2240 N1 = 19:N2 = 30:N3 = 2: GOSUB 3800: IF IN$ = ESC$ THEN RETURN
2250 IF CX < 1 OR CX > NC THEN PRINT BL$;: GOTO 2240
2260 CN$ = "                "
2270 VTAB 21: HTAB 1: PRINT "CATEGORY #" STR$ (CX)":"
2280 IF CX = 1 THEN VTAB 23: HTAB 1: PRINT "ENTER BUDGET INCOME:": GOTO 2300
2290 VTAB 23: HTAB 1: PRINT "ENTER BUDGET AMOUNT:"
2300 IF CX = 1 THEN VTAB 21: HTAB 16: PRINT CN$;: GOTO 2340
2310 HB = 15: GOSUB 3960: IF IN$ = ESC$ THEN RETURN
2320 IF LEFT$ (CN$,3) = "/D/" THEN GOSUB 4680: RETURN
2330 CT$(CX) = CN$
2340 CH = CX
2350 CN$ = "        ":N1 = 23:N2 = 22:N3 = 6: GOSUB 3800: IF IN$ = ESC$ THEN RETURN
2360 BD(CH) = CX
2370 VTAB 19: HTAB 1: CALL - 868: PRINT "ANOTHER CATEGORY? (Y/N)"
2380 VC = 19:HC = 25: GOSUB 4370
2390 IF IN$ = "Y" THEN VTAB 19: HTAB 1: CALL - 868: GOTO 2180
2400 RETURN
2410 REM    INITIALIZE ITEMS
2420 HOME
2430 VTAB 1: HTAB 7: PRINT "INITIALIZE BUDGET ITEMS"
2440 VTAB 10: HTAB 1: PRINT "ARE YOU SURE?? THIS WILL ERASE ALL"
2450 PRINT "ITEMS IN MEMORY!! (Y/N)"
2460 VC = 11:HC = 27: GOSUB 4370
2470 IF IN$ < > "Y" THEN RETURN
2480 FOR IT = 1 TO MD:BI$(IT) = BK$: NEXT
2490 FOR IT = 1 TO MC:NI(IT) = 0: NEXT
2500 NR = 0
2510 RETURN
2520 REM    ADD ITEM
2530 HOME : VTAB 1: HTAB 10: PRINT "ADD ITEM"
2540 IF NR < MD THEN GOSUB 3700: GOTO 2590
2550 VTAB 10: HTAB 1: PRINT "THE FILE IS FULL."
2560 PRINT "THERE ARE "STR(MD)" ITEMS."
2570 FOR DI = 1 TO 1000: NEXT
2580 RETURN
2590 GOSUB 3100: IF IN$ = ESC$ THEN RETURN
2600 VTAB 19: HTAB 1: PRINT "BUDGET CATEGORY #" STR$ (CH)": "CT$(CH)
2610 CN$ = LEFT$ (BK$,12):HB = 24
2620 VTAB 21: HTAB 1: CALL - 868: PRINT "ITEM #" STR$ (NR + 1)" DESCRIPTION:"
2630 VTAB 23: HTAB 11: CALL - 868: PRINT "ITEM AMOUNT:     0"
2640 GOSUB 3960: IF IN$ = ESC$ THEN RETURN
2650 BI$(NR + 1) = CHR$ (100 + CH) + CN$ + "0      "
2660 CN$ = "0      ":N1 = 23:N2 = 25:N3 = 6: GOSUB 3800: IF IN$ = ESC$ THEN RETURN
2670 BI$(NR + 1) = LEFT$ (BI$(NR + 1),13) + CN$
2680 NR = NR + 1:NI(CH) = NI(CH) + 1: IF NR = MD THEN RETURN
2690 IF NR = MD THEN RETURN
2700 VTAB 19: HTAB 1: CALL - 868: PRINT "ANOTHER ITEM? (Y/N)"
2710 VC = 19:HC = 21: GOSUB 4370
2720 IF IN$ = "Y" THEN VTAB 3:: HTAB 1: CALL - 958:CH = O: GOTO 2590
2730 RETURN
2740 REM    CHANGE ITEM
2750 HOME : VTAB 1: HTAB 10: PRINT "CHANGE ITEMS"
2760 IF NR > 0 THEN 2790
2770 VTAB 10: HTAB 1: PRINT "THERE ARE NO ITEMS TO CHANGE."
2780 FOR DI = 1 TO 1000: NEXT : RETURN
2790 VTAB 3: HTAB 1: CALL - 958: GOSUB 3700: GOSUB 3100: IF IN$ = ESC$ THEN RETURN
2800 IF NI(CH) > 0 THEN 2830
2810 VTAB 19: HTAB 1: CALL - 868: PRINT "THERE ARE NO ITEMS IN THIS CATEGORY."
2820 FOR DI = 1 TO 1000: NEXT : VTAB 19: HTAB 1: CALL - 868: GOTO 2790
2830 VTAB 3: CALL - 958:CN = 0: FOR IT = 1 TO NR: IF ASC (BI$(IT)) - 100 < > CH THEN 2900
2840 PRINT STR$ (IT); TAB( 3);")"; MID$ (BI$(IT),2,12); TAB( 20);"$"; VAL ( RIGHT$ (BI$(IT),7)):CN = CN + 1
2850 IF INT (CN / 8) < > CN / 8 THEN 2900
2860 VTAB 20: HTAB 7: PRINT "PRESS SPACE BAR TO CONTINUE": VTAB 22: HTAB 10: PRINT "PRESS RETURN TO SELECT":CN = 0
2870 GET K$: IF K$ < > CR$ AND K$ < > " " THEN 2870
2880 VTAB 19: CALL - 958: VTAB 3
2890 IF K$ = CR$ THEN 2910
2900 NEXT IT
2910 VTAB 19: HTAB 1: PRINT "BUDGET CATEGORY #" STR$ (CH)": "CT$(CH)
2920 VTAB 23: HTAB 1: CALL - 868
2930 VTAB 21: HTAB 1: CALL - 868: PRINT "ITEM #"
2940 N1 = 21:N2 = 7:N3 = 3:CN$ = "   "
2950 GOSUB 3800: IF IN$ = ESC$ THEN RETURN
2960 Z1 = CX: IF Z1 < 1 OR Z1 > NR THEN PRINT BL$;: GOTO 2950
2970 IF CH < > ASC (BI$(Z1)) - 100 THEN PRINT "ITEM #";Z1;" NOT IN THIS CATEGORY";BL$;: FOR DE = 1 TO 1000: NEXT : GOTO 2950
2980 VTAB 21: HTAB 9: CALL - 868: PRINT " DESCRIPTION:"
2990 VTAB 23: HTAB 11: CALL - 868: PRINT "ITEM AMOUNT:      "
3000 CN$ = "            ":HB = 24: GOSUB 3960: IF IN$ = ESC$ THEN RETURN
3010 IF LEFT$ (CN$,3) = "/D/" THEN GOSUB 4610: GOTO 3050
3020 BI$(Z1) = LEFT$ (BI$(Z1),1) + CN$ + RIGHT$ (BI$(Z1),6)
```

*Continued*

```
3030 CN$ = "          ":N1 = 23:N2 = 25:N3 =
      6: GOSUB 3800: IF IN$ = ESC$ THEN
      RETURN
3040 BI$(Z1) = LEFT$ (BI$(Z1),13) + CN$
3050 VTAB 19: HTAB 1: CALL - 868: PRINT
      "ANOTHER ITEM? (Y/N) "
3060 VC = 19:HC = 21: GOSUB 4370
3070 IF IN$ = "Y" THEN 2790
3080 RETURN
3090 REM    ENTER CATEGORY CHOICE
3100 VTAB 19: HTAB 1: CALL - 958: VTAB
     19: HTAB 3: PRINT "ENTER CATEGORY C
     HOICE (1 - " STR$ (NC) ") :"
3110 CN$ = "1 ":N1 = 19:N2 = 35:N3 = 2
3120 GOSUB 3800: IF IN$ = ESC$ THEN RET
     URN
3130 IF CX < 1 OR CX > NC THEN  PRINT BL
     $;: GOTO 3120
3140 CH = CX
3150 RETURN
3160 REM    PRINT ALL DETAILS
3170 CP = 0:LM = 20
3180 FOR IT = 1 TO NC
3190 PRINT " ": PRINT CT$(IT)" " RIGHT$
     ((" " + STR$ (BD(IT))),6)
3200 CP = CP + 2: IF PX = 2 AND CP > LM
     THEN GOSUB 3640
3210 IF NI(IT) = 0 THEN 3300
3220 FOR JI = 1 TO NR
3230 IF ASC ( LEFT$ (BI$(JI),1)) - 100
     < > IT THEN 3260
3240 PRINT "  " MID$ (BI$(JI),2,12); RI
     GHT$ ((" " + STR$ ( VAL (
      RIGHT$ (BI$(JI),6)))),10)
3250 CP = CP + 1: IF PX = 1 AND CP > LM
     THEN GOSUB 3640
3260 NEXT JI
3270 PRINT "            ----------"
3280 PRINT "            "; RIGHT$ (("
          " + STR$ (EX(IT))),10)
3290 CP = CP + 3: IF CP > LM AND PX = 1
     THEN GOSUB 3640
3300 NEXT IT
3310 PRINT " ": IF PX = 1 AND CP > 0 THE
     N GOSUB 3640
3320 PRINT " "
3330 RETURN
3340 REM    PRINT CHART
3350 IF NC < 2 THEN  RETURN
3360 CP = 0:LM = 5
3370 FOR IT = 2 TO NC
3380 PRINT CT$(IT);" : ";
3390 FOR JI = 1 TO  INT (BD(IT) * SC + .
     5001): PRINT "B";: NEXT JI: PRINT "
     "
3400 C$ = "E": IF EX(IT) > BD(IT) THEN C
     $ = "*"
3410 PRINT "           : ";
3420 FOR JI = 1 TO  INT (EX(IT) * SC + .
     5001): PRINT C$;: NEXT JI: PRINT "
     ": PRINT " "
3430 CP = CP + 1: IF CP = LM AND PX = 1
     THEN GOSUB 3640
3440 NEXT IT
3450 IF CP > 0 AND PX = 1 THEN  GOSUB 36
     40
3460 RETURN
3470 REM    MONTHLY SUMMARY REPORT
3480 PRINT " "
3490 PRINT "BUDGET INCOME:      " RIGHT$
      ((" " + STR$ (BD(1))),10
     )
3500 PRINT "ACTUAL INCOME:      " RIGHT$
      ((" " + STR$ (EX(1))),10
     )
3510 PRINT " "
3520 PRINT "ACTUAL EXPENSES:    " RIGHT$
      ((" " + STR$ (TE)),10)
3530 PRINT "CASH BALANCE:       " RIGHT$
      ((" " + STR$ (EX(1) - TE
     )),10)
3540 PRINT " "
3550 PRINT "BUDGETED EXPENSES:  " RIGHT$
      ((" " + STR$ (TB)),10)
3560 PRINT " "
3570 PRINT "DEVIATION FROM"
3580 PRINT "BUDGETED EXPENSES:  " RIGHT$
      ((" " + STR$ (TB - TE)),
     10)
3590 PRINT " "
3600 IF PX = 2 THEN  PRINT CHR$ (4)"PR#
     0"
3610 IF PX = 1 THEN  GOSUB 3640
3620 RETURN
3630 REM    LIST CONTINUATION PROMPT
3640 PRINT "PRESS ANY KEY TO CONTINUE:
     ";
3650 IN$ = "": GET IN$: IF IN$ = "" THEN
      3650
3660 CP = 0
3670 PRINT CR$
3680 RETURN
3690 REM    DISPLAY CATEGORIES
3700 VTAB 3: FOR IT = 1 TO NC
3710 IF NC > = IT THEN  HTAB 1: PRINT
     RIGHT$ ((" " + STR$ (IT)),2)". "C
     T$(IT); TAB( 20);"$"; BD(IT): PRINT
3720 IF IT / 8 < > INT (IT / 8) THEN 3
     770
3730 VTAB 20: HTAB 7: PRINT "PRESS SPACE
      BAR TO CONTINUE": VTAB 22: HTAB 10
     : PRINT "PRESS RETURN TO SELECT"
3740 GET K$: IF K$ < > CR$ AND K$ < >
     " " THEN 3740
3750 VTAB 19: CALL - 958: IF K$ = CR$ T
     HEN 3780
3760 IF IT < NC THEN  VTAB 3: CALL - 95
     8: VTAB 3
3770 NEXT IT
3780 RETURN
3790 REM    NUMBER ENTRY
3800 VTAB N1: HTAB N2: PRINT CN$
3810 FOR IT = 1 TO N3:ET$(IT) = MID$ (C
     N$,IT,1): NEXT
3820 HX = N2
3830 IF HX < N2 THEN HX = N2
3840 IF HX > N2 + N3 - 1 THEN HX = N2 +
     N3 - 1
3850 HC = HX:VC = N1: GOSUB 4370
3860 IF IN$ = ESC$ THEN  RETURN
3870 IF IN$ = LF$ THEN HX = HX - 1: GOTO
      3830
3880 IF IN$ = RT$ THEN HX = HX + 1: GOTO
      3830
3890 IF IN$ = CR$ THEN 3920
3900 IF (IN$ > = "0" AND IN$ < = "9")
     OR IN$ = " " THEN ET$(HX - N2 + 1)
     = IN$:HX = HX + 1: GOTO 3830
3910 PRINT BL$;: GOTO 3830
3920 IN$ = "": FOR IT = 1 TO N3:IN$ = IN
     $ + ET$(IT): NEXT :CX = VAL (IN$)
3930 CN$ = LEFT$ (( STR$ (CX) + "
     "),N3): VTAB N1: HTAB N2: PRINT CN
     $
3940 RETURN
3950 REM    CATEGORY OR ITEM NAME EDIT
3960 VTAB 21: HTAB HB + 1: PRINT CN$
3970 CL = LEN (CN$)
3980 FOR IT = 1 TO CL:ET$(IT) = MID$ (C
     N$,IT,1): NEXT
3990 HX = 1
4000 IF HX < 1 THEN HX = 1
4010 IF HX > CL THEN HX = CL
4020 VC = 21:HC = HB + HX: GOSUB 4370
4030 IF IN$ = ESC$ THEN  RETURN
4040 IF IN$ = LF$ THEN HX = HX - 1: GOTO
      4000
4050 IF IN$ = RT$ THEN HX = HX + 1: GOTO
      4000
4060 IF IN$ = CR$ THEN 4080
4070 ET$(HX) = IN$:HX = HX + 1: GOTO 400
     0
4080 CN$ = "": FOR IT = 1 TO CL:CN$ = CN
     $ + ET$(IT): NEXT
4090 RETURN
4100 REM    FIGURE TOTALS
4110 HOME : VTAB 10: HTAB 10: PRINT "GET
     TING DATA..."
4120 TE = 0:TB = 0: FOR IT = 1 TO NC:EX(
     IT) = 0: NEXT
4130 FOR IT = 1 TO NR:X = ASC (BI$(IT))
      - 100:EX(X) = EX(X) + VAL ( RIGHT
     $ (BI$(IT),6)): NEXT
4140 SC = 0
4150 FOR IT = 2 TO NC:TE = TE + EX(IT):T
     B = TB + BD(IT)
4160 IF SC < EX(IT) THEN SC = EX(IT)
4170 IF SC < BD(IT) THEN SC = BD(IT)
4180 NEXT
4190 IF SC < > 0 THEN SC = SF / SC
4200 IF SC = 0 THEN SC = 1
4210 HOME
4220 RETURN
4230 REM    TURN PRINTER ON
4240 PRINT " ": PRINT CHR$ (4)"PR#1"
4250 RETURN
4260 DR$ = "1"
4270 VTAB 14: HTAB 10: PRINT "DRIVE: "DR
     $
4280 VC = 14:HC = 17: GOSUB 4370: IF IN$
      < > "1" AND IN$ < > "2" AND IN$
     < > CR$ AND IN$ < > ESC$ THEN  PR
      INT BL$;: GOTO 4280
4290 IF IN$ < > CR$ THEN DR$ = IN$
4300 IF PD THEN  GOSUB 4990
4310 RETURN
4320 REM    MENU ITEM SELECT CHECK
4330 VC = 15:HC = 7: GOSUB 4370
4340 IF (IN$ < "1" OR IN$ > UL$) AND IN$
      < > ESC$ THEN  PRINT BL$;: GOTO 4
     330
4350 RETURN
4360 REM    GET CHARACTER
4370 VTAB VC: HTAB HC
4380 IN$ = ""
4390 GET IN$: IF IN$ = "" THEN 4390
4400 IF ASC (IN$) > 31 THEN  VTAB VC: H
     TAB HC: PRINT IN$;
4410 RETURN
4420 REM    ERROR HANDLER
4430 HOME : VTAB 7: HTAB 1
4440 X = PEEK (222)
4450 IF X = 22 THEN 240
```

*Continued*

## BUDGETRON *Continued*

APPLE // Family

```
4460 ON (X = 5 + 2 * (X = 9 OR X = 17) + 3 * (X = 8)) GOTO 4550,4580,4590
4470 PRINT "ERROR NUMBER "X" ";
4480 PRINT "AT LINE "( PEEK (218) + PEEK (219) * 256)
4490 VTAB 12: HTAB 1
4500 PRINT "PRESS ANY KEY TO CONTINUE: ";
4510 GET IN$: IF IN$ = "" THEN 4510
4520 POP : POP : POP : POP : POP : POP : POP : POP : POP : POP : POP : POP : POP : POP : POP
4530 GOTO 240
4540 RETURN
4550 PRINT "'"FL$"'"
4560 PRINT " IS NOT ON DRIVE " RIGHT$ (DR$,1)
4570 GOTO 4490
4580 PRINT "DISK IS FULL": GOTO 4490
4590 PRINT "IS THE DRIVE DOOR OPEN?": GOTO 4490
4600 REM DELETE ITEM
4610 FOR I = Z1 TO NR
4620 IF I = NR THEN 4640
4630 BI$(I) = BI$(I + 1): NEXT I
4640 NR = NR - 1
4650 RETURN
4660 REM DELETE CATEGORY
4670 REM DELETE ALL ITEMS IN CATEGORY
4680 Z1 = 1
4690 IF (NR < Z1) THEN 4730
4700 IF ASC (BI$(Z1)) - 100 = CX THEN GOSUB 4610:Z1 = Z1 - 1
4710 Z1 = Z1 + 1: GOTO 4690
4720 REM DELETE THE CATEGORY
4730 IF NC = 2 THEN NC = 1:CT$(2) = "": RETURN
4740 FOR I = CX TO NC - 1
4750 CT$(I) = CT$(I + 1)
4760 FOR J = 1 TO NR: IF ASC (BI$(J)) - 100 = I + 1 THEN BI$(J) = CHR$ (I + 100) + RIGHT$ (BI$(J), LEN (BI$(J)) - 1)
4770 NEXT J: NEXT I:CT$(NC) = "":NC = NC - 1: RETURN
4780 REM FILENAME ENTRY
4790 CR$ = CHR$ (13)
4800 T$ = "":VT = 7:HT = 1
4810 IF PD THEN MX = 15: GOTO 4830
4820 MX = 30
4830 VTAB VT: HTAB HT: PRINT T$; SPC( MX - LEN (T$));: HTAB LEN (T$) + HT
4840 IF LEN (T$) = MX THEN HTAB MX
4850 IF T$ = "" THEN GOSUB 4920: GOTO 4870
4860 GOSUB 4940
4870 IF K$ = CR$ THEN 4970
4880 IF K$ = LF$ AND LEN (T$) < 2 THEN T$ = "": GOTO 4830
4890 IF K$ = LF$ THEN T$ = LEFT$ (T$, LEN (T$) - 1): GOTO 4830
4900 IF LEN (T$) < MX THEN T$ = T$ + K$: GOTO 4830
4910 T$ = LEFT$ (T$,MX - 1) + K$: GOTO 4830
4920 GET K$: IF NOT (K$ = LF$ OR K$ = CR$ OR (K$ > = "A" AND K$ < = "Z")) THEN PRINT CHR$ (7);: GOTO 4920
4930 RETURN
4940 IF PD THEN GET K$: IF NOT (K$ = LF$ OR K$ = CR$ OR (K$ > = "A" AND K$ < = "Z") OR (K$ > = "0" AND K$ < = "9") OR K$ = ".") THEN PRINT CHR$ (7);: GOTO 4940
4950 IF PD = 0 THEN GET K$: IF NOT (K$ = LF$ OR K$ = CR$ OR (K$ > = " " AND K$ < = "+") OR (K$ > = "-" AND K$ < = "~")) THEN PRINT CHR$ (7);: GOTO 4950
4960 RETURN
4970 RETURN : REM LINK TO DISK ACCESS ROUTINE
4980 REM GET PREFIX
4990 PRINT CHR$ (4);"PREFIX,D";DR$: RETURN
```

HCM

## BUDGETRON

COMMODORE 64

```
100 REM *************
110 REM * BUDGETRON *
120 REM *************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM BY SCOTT WILLIAMS
160 REM AND THE HCM STAFF
170 REM HOME COMPUTER MAGAZINE
180 REM VERSION 5.3.1
190 REM C-64 BASIC
200 REM
210 REM DISPLAY TITLE SCREEN
220 POKE 53281,15:POKE 53280,15:POKE 646,0:POKE 53272,21:POKE 657,128
230 T$="BUDGETRON"
240 PRINT"☞SHIFT CLR☜☞10CRSRDOWN☜  **"T$"**"
250 PRINT"☞12CRSRDOWN☜         PRESS☞SPACE☜☞CTRL RVSON☜RETURN☞CTRL RVSOFF☜☞SPACE☜TO☞SPACE☜CONTINUE":X=15:Y=10:S=1
260 GET K$:GOSUB3320:S=ABS(S-1):POKE 199,S
270 FOR I=1 TO 50:NEXT:PRINTT$:IF K$="" THEN260
280 IF ASC(K$)<>13 THEN260
290 REM INITIALIZE VARIABLES
300 MD=100:MC=16:DIM A$(MD),B(MC),E(MC),N$(MC):N$(1)="$$ INCOME $$":NC=1:NR=0
310 SD=54272:POKE SD,48:POKE SD+1,4:POKE SD+3,140:POKE SD+6,240:POKE SD+24,15
320 REM MAIN MENU
330 PRINT"☞SHIFT CLR☜"TAB(15)T$:PRINTTAB(15)"MAIN MENU☞6CRSRDOWN☜"
340 PRINTTAB(8)"1) WORK WITH BUDGET☞CRSRDOWN☜":PRINTTAB(8)"2) REPORTS☞CRSRDOWN☜"
350 PRINTTAB(8)"3) BAR CHART☞CRSRDOWN☜":PRINTTAB(8)"4) FILES☞CRSRDOWN☜"
360 X=5:GOSUB3030:ON K GOTO370,510,2040,620,2970
370 REM WORK WITH BUDGET MENU
380 PRINT"☞SHIFT CLR☜"TAB(15)T$:PRINTTAB(9)"WORK WITH BUDGET MENU☞6CRSRDOWN☜"
390 PRINTTAB(8)"1) BUDGET CATEGORIES☞CRSRDOWN☜":PRINTTAB(8)"2) INCOME AND EXPENSES☞CRSRDOWN☜"
400 X=3:GOSUB3030:ON K GOTO410,460,320
410 REM BUDGET CATEGORIES MENU
420 PRINT"☞SHIFT CLR☜"TAB(15)T$:PRINTTAB(8)"BUDGET CATEGORIES MENU☞6CRSRDOWN☜"
430 PRINTTAB(8)"1) INITIALIZE BUDGET☞CRSRDOWN☜":PRINTTAB(8)"2) ADD A CATEGORY☞CRSRDOWN☜"
440 PRINTTAB(8)"3) CHANGE A CATEGORY☞CRSRDOWN☜"
450 X=4:GOSUB3030:ON K GOSUB660,800,870,370:GOTO410
460 REM INCOME AND EXPENSES MENU
470 PRINT"☞SHIFT CLR☜"TAB(15)T$:PRINTTAB(8)"INCOME AND EXPENSES MENU☞6CRSRDOWN☜"
480 PRINTTAB(8)"1) INITIALIZE ITEMS☞CRSRDOWN☜":PRINTTAB(8)"2) ADD ITEM☞CRSRDOWN☜"
490 PRINTTAB(8)"3) CHANGE ITEM☞CRSRDOWN☜"
500 X=4:GOSUB3030:ON K GOSUB730,1080,1160,370:GOTO460
510 REM REPORTS MENU
520 IF NC=1 OR NR=0 THEN GOTO2060
530 PRINT"☞SHIFT CLR☜"TAB(15)T$:PRINTTAB(13)"REPORTS MENU☞6CRSRDOWN☜"
540 PRINTTAB(8)"1) ALL CATEGORIES AND ITEMS☞CRSRDOWN☜":PRINTTAB(8)"2) MONTHLY SUMMARY☞CRSRDOWN☜"
550 PRINTTAB(8)"3) PRINT CHART☞CRSRDOWN☜"
560 X=4:GOSUB3030:V=K:IF V=4 THEN320
570 PRINT"☞SHIFT CLR☜"TAB(12)"CHOOSE OUTPUT☞6CRSRDOWN☜"
580 PRINTTAB(8)"1) SCREEN☞CRSRDOWN☜":PRINTTAB(8)"2) PRINTER☞CRSRDOWN☜"
590 X=3:GOSUB3030:IF K=3 THEN320
600 IF K=1 THEN ON V GOSUB2210,2260,2310:GOTO320
610 OPEN4,4:CMD4:ON V GOSUB2400,2420,2440:PRINT#4:CLOSE4:GOTO320
620 REM FILES MENU
630 PRINT"☞SHIFT CLR☜"TAB(15)T$:PRINTTAB(14)"FILES MENU☞6CRSRDOWN☜"
640 PRINTTAB(8)"1) LOAD DATA☞CRSRDOWN☜":PRINTTAB(8)"2) SAVE DATA☞CRSRDOWN☜"
650 X=3:GOSUB3030:ON K GOSUB1550,1730:GOTO320
660 REM INITIALIZE BUDGET
670 POKE 198,0:PRINT"☞SHIFT CLR☜       ARE YOU SURE?? THIS WILL☞CRSRDOWN☜"
680 PRINT"     ERASE ALL THE BUDGET DATA (Y/N)☞3CRSRDOWN☜"
690 GET K$:IF K$<>"Y" AND K$<>"N" THEN690
700 IF K$="N" THEN RETURN
710 PRINT"INITIALIZING.";
720 FOR I=2 TO MC:PRINT".";:N$(I)="":B(I)=0:E(I)=0:NEXT:B(1)=0:NC=1:RETURN
730 REM INITIALIZE ITEMS
740 POKE 198,0:PRINT"☞SHIFT CLR☜       ARE YOU SURE?? THIS WILL ERASE☞CRSRDOWN☜"
750 PRINT"     ALL THE ITEMS IN MEMORY (Y/N)☞3CRSRDOWN☜"
```

*Continued*

## BUDGETRON *Continued*

```
 760 GET K$:IF K$<>"Y" AND K$<>"N" THEN7
     60
 770 IF K$="N" THEN RETURN
 780 PRINT"INITIALIZING.";
 790 FOR I=1 TO MD:PRINT".";:A$(I)="":NE
     XT:NR=0:RETURN
 800 REM ADD A CATEGORY ROUTINE
 810 IF NC=MC THEN PRINT"☞SHIFT CLR☜
       ALL CATEGORIES ARE FULL":GOTO2950
 820 IF NC<>1 OR B(1)<>0 THEN840
 830 PRINT"☞SHIFT CLR☜☞14CRSRDOWN☜ ENTER
      BUDGET INCOME:  ";:GOSUB2870:B(1)=V
 840 GOSUB2740:NC=NC+1:C=NC:GOSUB940:IF
     S$="" THEN NC=NC-1:RETURN
 850 IF K$="Y" THEN800
 860 RETURN
 870 REM CHANGE CATEGORY ROUTINE
 880 IF NC=1 AND B(1)=0 THEN GOTO2920
 890 GOSUB2740:GOSUB1030:IF S$="" THEN R
     ETURN
 900 IF C=0 THEN890
 910 GOSUB940:IF S$="" THEN RETURN
 920 IF K$="Y" THEN890
 930 RETURN
 940 REM INPUT HEADING AND AMOUNT
 950 IF C<>1 THEN PRINT"ENTER CATEGORY H
     EADING":PRINT">";:GOSUB2900
 960 IF S$="" THEN RETURN
 970 IFS$="/D/"THENGOSUB3370:S$="":RETUR
     N
 980 IF C<>1 THEN N$(C)=S$
 990 PRINT:PRINT"ENTER BUDGET AMOUNT":PR
     INT">";:GOSUB2870:B(C)=V
1000 PRINT:PRINT"ANOTHER CATEGORY (Y/N)?
     "
1010 GET K$:IF K$<>"Y" AND K$<>"N" THEN1
     010
1020 RETURN
1030 REM INPUT CATEGORY NUMBER
1040 PRINT" ENTER CATEGORY NUMBER:  ";:L=
     2:B=48:T=57:GOSUB3080:PRINT
1050 IF S$="" THEN RETURN
1060 C=VAL(S$):IF C>0 AND C<=NC THEN RET
     URN
1070 PRINT:PRINT"** CATEGORY NOT FOUND *
     *":C=0:GOTO2950
1080 REM ADD ITEMS
1090 IF NR=MD THEN PRINT"☞SHIFT CLR☜THE
     FILE IS FULL. THERE ARE"MD"ITEMS":
     GOTO2950
1100 IFNC=1ANDB(1)=0 THEN PRINT"☞SHIFT C
     LR☜ YOU NEED TO ENTER CATEGORIES F
     IRST":GOTO2950
1110 GOSUB2740:GOSUB1030:IF S$="" THEN R
     ETURN
1120 IF C=0 THEN1110
1130 NR=NR+1:ZZ=NR:GOSUB1470:IF S$="" TH
     EN NR=NR-1:RETURN
1140 IF K$="Y" THEN1090
1150 RETURN
1160 REM CHANGE ITEMS
1170 IFNC=1ANDB(1)=0 THEN PRINT"☞SHIFT C
     LR☜ YOU NEED TO ENTER CATEGORIES F
     IRST":GOTO2950
1180 IF NR=0 THEN PRINT"☞SHIFT CLR☜ YOU
     HAVE NO ITEMS IN MEMORY TO CHANGE":
     GOTO2950
1190 GOSUB2740:GOSUB1030:IF S$="" THEN R
     ETURN
1200 IF C=0 THEN1190
1210 CT=0:POKE 198,0:PRINT"☞SHIFT CLR☜☞C
     RSRDOWN☜ CATEGORY:"C"☞SHIFT CRSRLEF
     T☜)  "N$(C)
1220 PRINT"☞CRSRDOWN☜      ☞CTRL RVSON☜IT
     EM☞CTRL RVSOFF☜
      ☞SPACE☜☞CTRL RVSON☜AMOUNT☞CTRL RVS
     OFF☜☞CRSRDOWN☜"
1230 FOR I=1 TO NR:IF ASC(A$(I))-100<>C
     THEN1260
1240 CT=CT+1:TX=ASC(MID$(A$(I),2,1))-200
1250 PRINTI"☞SHIFT CRSRLEFT☜)  "MID$(A$(I
     ),3,TX-2)TAB(31)VAL(MID$(A$(I),TX+1
     ,LEN(A$(I))-TX))
1260 IF ((CT/5)<>INT(CT/5)) OR (I=NR) OR
      (CT=0) THEN1350
1270 X=0:Y=23:GOSUB3320:PRINT"     PRESS
      THE SPACE BAR FOR MORE"
1280 PRINT"      PRESS ☞CTRL RVSON☜RETU
     RN☞CTRL RVSOFF☜ TO CONTINUE";
1290 GET K$:IF K$<>CHR$(13) AND K$<>" "
     THEN1290
1300 PRINT"☞2SHIFT CRSRUP☜":PRINT"
                                  "
1310 PRINT"
         ";
1320 IF K$=CHR$(13) THEN I=NR:GOTO1350
1330 PRINT"☞SHIFT CLR☜☞CRSRDOWN☜ CATEGOR
     Y:"C"☞SHIFT CRSRLEFT☜)  "N$(C)
1340 PRINT"☞CRSRDOWN☜      ☞CTRL RVSO
     N☜ITEM☞CTRL RVSOFF☜
      ☞SPACE☜☞CTRL RVSON☜AMOUNT☞CTRL RVS
     OFF☜☞CRSRDOWN☜"
1350 NEXT:IF CT<>0 THEN1370
1360 PRINT"☞2SHIFT CRSRUP☜ THERE ARE NO
      ITEMS IN THIS CATEGORY  ":GOSUB295
     0:GOTO1160
1370 X=0:Y=14:GOSUB3320
1380 PRINT"ENTER ITEM NUMBER:  ";:L=3:B=4
     8:T=57:GOSUB3080:PRINT
1390 IF S$="" THEN1160
1400 V=VAL(S$):IF V>MD THEN1420
1410 IF ASC(A$(V))-100=C THEN1430
1420 PRINT"ITEM IS NOT IN THIS CATEGORY"
     :GOSUB2950:GOTO1160
1430 ZZ=V:GOSUB1470:IF S$="" THEN RETURN
1440 IFS$="/D/"THENGOSUB3340:RETURN
1450 IF K$="Y" THEN1160
1460 RETURN
1470 REM INPUT ITEM DESC AND AMOUNT
1480 PRINT"ENTER ITEM DESCRIPTION":PRINT
     ">";:GOSUB2900:IF S$="" THEN RETURN
1490 IFS$="/D/"THENGOSUB3340:S$="":RETUR
     N
1500 A$(ZZ)=S$:PRINT:PRINT"ENTER ITEM AM
     OUNT":PRINT">";:GOSUB2870
1510 A$(ZZ)=CHR$(C+100)+CHR$(LEN(A$(ZZ))
     +202)+A$(ZZ)+STR$(V):PRINT
1520 PRINT"ANOTHER (Y/N)?"
1530 GET K$:IF K$<>"Y" AND K$<>"N" THEN1
     530
1540 RETURN
1550 REM LOAD FILE ROUTINE
1560 PRINT"☞SHIFT CLR☜"TAB(16)"LOAD FILE
     ☞5CRSRDOWN☜":GOSUB1920:IF S$="" THE
     N RETURN
1570 IF K$="D" THEN OPEN1,8,8,"0:"+F$+",
     S,R":GOSUB2000:IF E THEN1700
1580 IF K$="T" THEN OPEN1,1,0,F$
1590 PRINT"☞CRSRDOWN☜LOADING";
1600 INPUT#1,NC
1610 FOR I=1 TO NC:PRINT".";
1620 INPUT#1,N$(I)
1630 INPUT#1,B(I)
1640 NEXT:IF K$="D" THEN GOSUB2020:IF E
     THEN1700
1650 INPUT#1,NR:IF NR=0 THEN1690
1660 FOR I=1 TO NR:PRINT".";
1670 INPUT#1,A$(I)
1680 NEXT
1690 IF K$="D" THEN GOSUB2020
1700 CLOSE1
1710 IF K$="D" THEN CLOSE15
1720 RETURN
1730 REM SAVE FILE ROUTINE
1740 IF NC=1 AND B(1)=0 THEN GOTO2920
1750 PRINT"☞SHIFT CLR☜"TAB(16)"SAVE FILE
     ☞5CRSRDOWN☜":GOSUB1920:IF S$="" THE
     N RETURN
1760 IF K$="D" THEN OPEN1,8,8,"@0:"+F$+"
     ,S,W":GOSUB2000:IF E THEN1890
1770 IF K$="T" THEN OPEN1,1,1,F$
1780 PRINT"☞CRSRDOWN☜SAVING";
1790 PRINT#1,NC
1800 FOR I=1 TO NC:PRINT".";
1810 PRINT#1,N$(I)
1820 PRINT#1,B(I)
1830 NEXT:IF K$="D" THEN GOSUB2020:IF E
     THEN1890
1840 PRINT#1,NR:IF NR=0 THEN1880
1850 FOR I=1 TO NR:PRINT".";
1860 PRINT#1,A$(I)
1870 NEXT
1880 IF K$="D" THEN GOSUB2020
1890 CLOSE1
1900 IF K$="D" THEN CLOSE15
1910 RETURN
1920 REM INPUT FILE NAME
1930 PRINT"INPUT FILE NAME:  ";
1940 L=14:B=32:T=90:GOSUB3080:F$=S$+".B"
     :IF S$="" THEN RETURN
1950 PRINT:PRINT"☞2CRSRDOWN☜TAPE OR DISK
      (T/D)☞4SHIFT CRSRLEFT☜";
1960 GET K$:IF K$<>"T" AND K$<>"D" THEN1
     960
1970 IF K$="D" THEN PRINT"☞2CRSRRIGHT☜☞C
     TRL RVSON☜D"
1980 IF K$="T" THEN PRINT"☞CTRL RVSON☜T"
1990 RETURN
2000 REM READ DISK ERROR
2010 OPEN 15,8,15
2020 INPUT#15,E,E$:IF E<>0 THEN PRINT "*
     * "E$" **":GOSUB2950
2030 RETURN
2040 REM DISPLAY BAR CHART
2050 IF NC<>1 AND NR<>0 THEN2080
2060 PRINT"☞SHIFT CLR☜        THERE IS NOT
      ENOUGH DATA TO☞CRSRDOWN☜"
2070 PRINT"          CREATE A CHART OR A REP
     ORT":GOSUB2950:GOTO320
2080 POKE 198,0:PRINT"☞SHIFT CLR☜
            THINKING":GOSUB2460:RG=CT/19
     :PRINT"☞SHIFT CLR☜"
2090 FOR I=2 TO NC:PRINT"☞CTRL BLK☜"N$(I
     )TAB(20)"☞CTRL RVSON☜";:IF B(I)=0
     THEN2110
2100 FOR J=1 TO INT(B(I)/RG):PRINT"☞CMDR
      T☜";:NEXT
2110 PRINT:IF E(I)>B(I) THEN PRINT"☞CTRL
      RED☜";
2120 PRINTTAB(20)"☞CTRL RVSON☜";:IF E(I)
     =0 THEN2140
```

Continued

## BUDGETRON *Continued*

```
2130 FOR J=1 TO INT(E(I)/RG):PRINT"☛CMDR
       T☚";:NEXT
2140 PRINT:PRINT:IF I=NC OR INT((I-1)/6)
     <>(I-1)/6 THEN2180
2150 X=6:Y=23:GOSUB3320:PRINT"☛CTRL BLK☚
     PRESS ☛CTRL RVSON☚RETURN☛CTRL RVSOF
     F☚ TO CONTINUE"
2160 GET K$:IF K$<>CHR$(13) THEN2160
2170 PRINT"☛SHIFT CLR☚"
2180 NEXT:X=9:Y=23:GOSUB3320:PRINT"☛CTRL
       BLK☚PRESS ☛CTRL RVSON☚RETURN☛CTRL
     RVSOFF☚ TO EXIT"
2190 GET K$:IF K$<>CHR$(13) THEN2190
2200 GOTO320
2210 REM DISPLAY CATEGORIES AND ITEMS
2220 PRINT"☛SHIFT CLR☚":GOSUB2460:FOR I=
     1 TO NC:GOSUB2670:X=5:Y=23:GOSUB332
     0
2230 PRINT"PRESS ☛CTRL RVSON☚RETURN☛CTRL
       RVSOFF☚ TO CONTINUE"
2240 GET K$:IF K$<>CHR$(13) THEN2240
2250 PRINT"☛SHIFT CLR☚":NEXT
2260 REM DISPLAY MONTHLY SUMMARY
2270 PRINT"☛SHIFT CLR☚":GOSUB2460:GOSUB2
     530:X=9:Y=23:GOSUB3320
2280 PRINT"PRESS ☛CTRL RVSON☚RETURN☛CTRL
       RVSOFF☚ TO EXIT"
2290 GET K$:IF K$<>CHR$(13) THEN2290
2300 RETURN
2310 REM DISPLAY CHART
2320 PRINT"☛SHIFT CLR☚":GOSUB2460:RG=CT/
     17:FOR I=2 TO NC:GOSUB2590
2330 IF INT((I-1)/6)<>(I-1)/6 THEN2370
2340 X=5:Y=23:GOSUB3320:PRINT"PRESS ☛CTR
     L RVSON☚RETURN☛CTRL RVSOFF☚ TO CONT
     INUE"
2350 GET K$:IF K$<>CHR$(13) THEN2350
2360 PRINT"☛SHIFT CLR☚"
2370 NEXT:X=9:Y=23:GOSUB3320:PRINT"☛CTRL
       BLK☚PRESS ☛CTRL RVSON☚RETURN☛CTRL
     RVSOFF☚ TO EXIT"
2380 GET K$:IF K$<>CHR$(13) THEN2380
2390 RETURN
2400 REM PRINT CATEGORIES AND ITEMS
2410 GOSUB2460:FOR I=1 TO NC:GOSUB2670:N
     EXT:PRINT:PRINT
2420 REM PRINT MONTHLY SUMMARY
2430 GOSUB2460:GOSUB2530:RETURN
2440 REM PRINT CHART
2450 GOSUB2460:RG=CT/58:FOR I=2 TO NC:GO
     SUB2590:NEXT:RETURN
2460 REM TOTAL EXPENSES AND FIND HIGHEST
2470 FOR I=1 TO MC:E(I)=0:NEXT:FOR I=1 T
     O NR:TX=ASC(MID$(A$(I),2,1))-200
2480 R=ASC(A$(I))-100:E(R)=E(R)+VAL(MID$
     (A$(I),TX+1,LEN(A$(I))-TX)):NEXT
2490 BT=0:ET=0:FOR I=2 TO MC:BT=BT+B(I):
     ET=ET+E(I):NEXT:CT=0:FOR I=2 TO NC
2500 IF CT<B(I) THEN CT=B(I)
2510 IF CT<E(I) THEN CT=E(I)
2520 NEXT:RETURN
2530 REM OUTPUT MONTHLY SUMMARY
2540 PRINT"MONTHLY BUDGET SUMMARY:":PRIN
     T
2550 PRINT"BUDGET INCOME:       "B(1):PRIN
     T"ACTUAL INCOME:       "E(1):PRINT
2560 PRINT"ACTUAL EXPENSES:     "ET:PRINT"
     CASH BALANCE:         "E(1)-ET
2570 PRINT:PRINT"BUDGETED EXPENSES:   "BT:
     PRINT:PRINT"DEVIATION FROM"
2580 PRINT"BUDGETED EXPENSES:   "BT-ET:RET
     URN
2590 REM OUTPUT CATEGORY FOR CHART
2600 PRINTN$(I);:FOR J=1 TO 21-LEN(N$(I)
     ):PRINT" ";:NEXT:PRINT":";
2610 IF B(I)=0 THEN2630
2620 FOR J=1 TO INT(B(I)/RG):PRINT"B";:N
     EXT:E$="E":IF E(I)>B(I) THEN E$="*"
2630 PRINT:PRINT"                     :"
     ;
2640 IF E(I)=0 THEN2660
2650 FOR J=1 TO INT(E(I)/RG):PRINTE$;:NE
     XT
2660 PRINT:PRINT"                     :"
     :RETURN
2670 REM OUTPUT CATEGORY AND ITEMS
2680 PRINTN$(I);:FOR J=1 TO 21-LEN(N$(I)
     ):PRINT" ";:NEXT:PRINTB(I):PRINT
2690 FOR J=1 TO NR:IF ASC(A$(J))-100<>I
     THEN2720
2700 TX=ASC(MID$(A$(J),2,1))-200:PRINT"
      "MID$(A$(J),3,TX-2);
2710 FOR K=1 TO 25-TX:PRINT" ";:NEXT:PRI
     NTMID$(A$(J),TX+1,LEN(A$(J))-TX)
2720 NEXT:PRINT"                        —
     ———————"
2730 PRINT"                        "E(I
     ):RETURN
2740 REM LIST CATEGORIES ROUTINE
2750 CT=0:POKE 198,0:PRINT"☛SHIFT CLR☚☛C
     RSRDOWN☚     ☛CTRL RVSON☚CATEGORY☛C
     TRL RVSOFF☚           ☛SPACE☚
     ☛CTRL RVSON☚AMOUNT☛CTRL RVSOFF☚☛CRS
     RDOWN☚"
2760 FOR I=1 TO NC
2770 PRINT"  "LEFT$("☛2SHIFT CRSRLEFT☚",I
     /10)I"☛SHIFT CRSRLEFT☚)  "N$(I)TAB(3
     1)B(I):IF I=NC THEN2860
2780 CT=CT+1:IF CT<>8 THEN2860
2790 X=0:Y=23:GOSUB3320:PRINT"      PRESS
      THE SPACE BAR FOR MORE"
2800 PRINT"          PRESS ☛CTRL RVSON☚RETU
     RN☛CTRL RVSOFF☚ TO CONTINUE";
2810 GET K$:IF K$<>CHR$(13) AND K$<>" "
     THEN2810
2820 PRINT"☛2SHIFT CRSRUP☚":PRINT"
                                 "
2830 PRINT"
          ";
2840 IF K$=CHR$(13) THEN I=NC:GOTO2860
2850 PRINT"☛SHIFT CLR☚☛CRSRDOWN☚      ☛CT
     RL RVSON☚CATEGORY☛CTRL RVSOFF☚
                   ☛SPACE☚☛CTRL RVSON☚AMOU
     NT☛CTRL RVSOFF☚☛CRSRDOWN☚"
2860 NEXT:X=0:Y=14:GOSUB3320:RETURN
2870 REM INPUT A NUMERIC AMOUNT
2880 L=6:B=48:T=57:GOSUB3080:V=VAL(S$):I
     F S$="" THEN S$="0"
2890 RETURN
2900 REM INPUT TEXT
2910 L=20:B=32:T=90:GOSUB3080:RETURN
2920 REM ERROR MESSAGE
2930 PRINT"☛SHIFT CLR☚  YOU NEED TO CREA
     TE A BUDGET BEFORE☛CRSRDOWN☚"
2940 PRINT"          CHOOSING THIS OPTION"
2950 REM BEEP AND PAUSE
2960 POKE SD+4,65:FOR I=1 TO 250:NEXT:PO
     KE SD+4,0:FOR I=1 TO 2000:NEXT:RETU
     RN
2970 REM EXIT PROGRAM
2980 POKE 198,0:PRINT"☛SHIFT CLR☚☛CRSRDO
     WN☚         ARE YOU SURE THAT YOU☛CRS
     RDOWN☚"
2990 PRINT"    WISH TO EXIT THE PROGRAM (
     Y/N)?"
3000 GET K$:IF K$<>"Y" AND K$<>"N" THEN3
     000
3010 IF K$="N" THEN320
3020 PRINT"☛SHIFT CLR☚BYE...":POKE 657,0
     :END
3030 REM PRINT'EXIT' AND GET MENU OPTION
3040 PRINTTAB(7)X"☛SHIFT CRSRLEFT☚) EXIT
     ":B=49:T=B+X-1:L=1:X=8:Y=23:GOSUB33
     20
3050 PRINT"CHOOSE ONE:   ";
3060 GOSUB3080:IF S$="" THEN3060:K=VAL(S
     $)
3070 K=VAL(S$):RETURN
3080 REM INPUT ROUTINE
3090 X=0:S$="":S=PEEK(214)*40+PEEK(211)+
     1024:POKE 213,PEEK(211)+L+1
3100 GOSUB3290:IF K=13 THEN3270
3110 GOSUB3120:GOTO3100
3120 IF X=0 THEN3150
3130 IF K=20 THEN K$="":I=-1:GOTO3230
3140 IF K=157 THEN I=-1:GOTO3260
3150 IF X=L THEN3280
3160 IF X=LEN(S$) THEN3190
3170 IF K=29 THEN I=1:GOTO3260
3180 IF K=148 AND LEN(S$)<L THEN K$="  "+
     MID$(S$,X+1,1):GOTO3220
3190 IF K=34 OR K<B OR K>T THEN3280
3200 IF K=34 OR K=36 OR K=42 OR K=44 OR
     K=58 OR K=63 OR K=64 THEN3280
3210 I=0
3220 X=X+1
3230 S$=LEFT$(S$,X-1)+K$+MID$(S$,X+1)
3240 IF K$="" THEN K$=CHR$(20)
3250 IF LEN(K$)=2 THEN K$=CHR$(148):I=-1
3260 PRINT K$;:POKE 216,0
3270 POKE 204,1:POKE S+X,PEEK(S+X) AND 1
     27:X=X+I
3280 RETURN
3290 REM INPUT ONE CHARACTER
3300 POKE 204,0:POKE 207,0:GET K$:IF K$=
     "" THEN3300
3310 K=ASC(K$):RETURN
3320 REM PLACE CURSOR AT X,Y
3330 POKE 781,Y:POKE 782,X:POKE 783,0:SY
     S 65520:RETURN
3340 FORZZ=VTONR:IFA$(ZZ+1)=""THEN3360
3350 A$(ZZ)=A$(ZZ+1):NEXT:A$(NR)=""
3360 NR=NR-1:RETURN
3370 FORDC=CTONC:IFDC<>NCTHENN$(DC)=N$(D
     C+1)
3380 NEXT:N$(NC)="":B(NC)=0:NC=NC-1:IFNR
     =0THENRETURN
3390 DI=1
3400 IFASC(A$(DI))-100<>CTHEN3430
3410 V=DI:GOSUB3340:IFNR>=DITHENGOTO3400
3420 RETURN
3430 IFASC(A$(DI))-100<CTHEN3450
3440 A$(DI)=CHR$(ASC(A$(DI))-1)+RIGHT$(A
     $(DI),LEN(A$(DI))-1)
3450 DI=DI+1:IFDI>NRTHENRETURN
3460 GOTO3400
```

# BUDGETRON

```
100 ' *************
110 ' * BUDGETRON *
120 ' *************
130 ' COPYRIGHT 1985
140 ' EMERALD VALLEY PUBLISHING CO
150 ' BY SCOTT WILLIAMS
160 ' AND THE HCM STAFF
170 ' HOME COMPUTER MAGAZINE
180 ' VERSION 5.3.1
190 ' IBM PC W/ BASICA AND
           COLOR/GRAPHICS MONITOR ADAPT
    ER
200 ' IBM PCjr CARTRIDGE BASIC
210 '
220 ' INITIALIZATION *****
230 KEY OFF:OPTION BASE 1:MAXDATA=500:M
    AXCAT=16:ENT$=CHR$(17)+CHR$(217):CR
    $=CHR$(13):SP$=CHR$(32):BS$=CHR$(8)
    :LT$=CHR$(0)+CHR$(75):RT$=CHR$(0)+C
    HR$(77):DEL$=CHR$(0)+CHR$(83):INS$=
    CHR$(0)+CHR$(82):TI$="BUDGETRON"
240 DIM A$(MAXDATA),B(MAXCAT),E(MAXCAT)
    ,N$(MAXCAT),ERCD(14),ERM$(14):RESTO
    RE 2350:FOR I=1 TO 14:READ ERCD(I):
    READ ERM$(I):NEXT
250 CLS:SCREEN 0:WIDTH 40:N$(1)="$$ INC
    OME $$":NC=1
260 T$=TI$:ROW=10:GOSUB 1870:GOSUB 1890
270 ' MAIN MENU     *****
280 CLS:ROW=2:T$=TI$:GOSUB 1870:ROW=4:T
    $="MAIN MENU":GOSUB 1870:RESTORE 22
    40:FOR I= 8 TO 16 STEP 2:LOCATE I,8
    :READ MEN$:PRINT I/2-3;") ";MEN$:NE
    XT I:LOCATE 19,8:PRINT "YOUR CHOICE
    : "
290 K$="":WHILE K$<"1" OR K$>"5":LOCATE
     19,21,1:GOSUB 1850:WEND:MD=VAL(K$)
300 ON MD GOSUB 330,1110,1390,1670,2220
310 GOTO 280
320 'WORK WITH BUDGET *****
330 CLS:RESTORE 2250:ROW=2:T$=TI$:GOSUB
     1870:ROW=4:T$="WORK WITH BUDGET":G
    OSUB 1870:FOR I= 8 TO 12 STEP 2:LOC
    ATE I,8:READ MEN$:PRINT I/2-3;") ";
    MEN$:NEXT I:LOCATE 15,8,1:PRINT "YO
    UR CHOICE:   "
340 K$="":WHILE K$<"1" OR K$>"3":LOCATE
     15,21,1:GOSUB 1850:WEND:MD1=VAL(K$
    )
350 IF MD1=3 THEN RETURN ELSE ON MD1 GO
    SUB 360,510:GOTO 330
360 CLS:RESTORE 2260:ROW=2:T$=TI$:GOSUB
     1870:ROW=4:T$="BUDGET CATEGORIES":
    GOSUB 1870:FOR I=8 TO 14 STEP 2:LOC
    ATE I,8:READ MEN$:PRINT I/2-3;") ";
    MEN$:NEXT I:LOCATE 17,8,1:PRINT "YO
    UR CHOICE:  ";
370 K$="":WHILE K$<"1" OR K$>"4":LOCATE
     17,21,1:GOSUB 1850:WEND:MD2=VAL(K$
    )
380 IF MD2=4 THEN RETURN:ELSE ON MD2 GO
    SUB 400,440,480:GOTO 360
390 ' INITIALIZE BUDGET *****
400 CLS:LOCATE 10,1,0:PRINT "ARE YOU SU
    RE YOU WANT TO ERASE ALL THE DATA
    IN THE PRESENT BUDGET? (Y/N)":K$=""
    :WHILE NOT(K$="Y" OR K$="y" OR K$="
    N" OR K$="n"):LOCATE 10,36:GOSUB 18
    50:WEND
410 IF K$="N" OR K$="n" THEN RETURN
420 FOR Z=2 TO MAXCAT:N$(Z)="":B(Z)=0:E
    (Z)=0:NEXT:B(1)=0:NC=1:RETURN
430 ' ADD A CATEGORY *****
440 CLS:IF NC=MAXCAT THEN ROW=10:T$="TH
    E CATEGORIES ARE FULL":GOSUB 1870:G
    OSUB 1890:RETURN
450 ROW=2:T$="ADD CATEGORIES":GOSUB 187
    0:IF NC=1 AND B(1)=0 THEN LOCATE 4,
    3:PRINT "ENTER BUDGET INCOME: $";:R
    OW=4:COL=25:MAXLEN=6:IN$="":GOSUB 1
    920:IF IN$="" THEN RETURN ELSE B(1)
    =VAL(IN$)
460 GOSUB 700:NC=NC+1:C=NC:GOSUB 760:IF
     IN$="" THEN NC=NC-1:RETURN ELSE GO
    SUB 800:IF K$="Y" OR K$="y" THEN 44
    0 ELSE RETURN
470 ' CHANGE A CATEGORY *****
480 CLS:ROW=2:T$="CHANGE CATEGORIES":GO
    SUB 1870
490 GOSUB 700:C=0:GOSUB 680:IF C=0 THEN
     RETURN ELSE GOSUB 760:IF IN$="" OR
     IN$="/D/" THEN RETURN ELSE GOSUB 8
    00:IF K$="Y" OR K$="y" THEN 480 ELS
    E RETURN
500 'INCOME AND EXPENSES *****
510 CLS:RESTORE 2270:ROW=2:T$=TI$:GOSUB
     1870:ROW=4:T$="INCOME AND EXPENSES
    ":GOSUB 1870:FOR I= 8 TO 14 STEP 2:
    LOCATE I,8:READ MEN$:PRINT I/2-3;")
     ";MEN$:NEXT:LOCATE 17,8,1:PRINT "Y
    OUR CHOICE:   "
520 K$="":WHILE K$<"1" OR K$>"4":LOCATE
     17,21,1:GOSUB 1850:WEND:MD2=VAL(K$
    )
530 IF MD2=4 THEN RETURN ELSE ON MD2 GO
    SUB 550,590,640:GOTO 510
540 ' INITIALIZE ITEMS *****
550 CLS:LOCATE 10,1,0:PRINT "ARE YOU SU
    RE YOU WANT TO ERASE ALL THE DATA
    IN THE PRESENT BUDGET? (Y/N)":K$=""
    :WHILE NOT(K$="Y" OR K$="y" OR K$="
    N" OR K$="n"):LOCATE 10,36:GOSUB 18
    50:WEND
560 IF K$="N" OR K$="n" THEN RETURN
570 FOR Z=2 TO MAXDATA:A$(Z)="":NEXT:NR
    =0:RETURN
580 ' ADD ITEMS *****
590 CLS:ROW=2:T$="ADD ITEMS":GOSUB 1870
    :IF NR=MAXDATA THEN ROW=4:T$="THE F
    ILE IS FULL":GOSUB 1870:ROW=6:T$="T
    HERE ARE ";MAXDATA;" ITEMS IN FILE"
    :GOSUB 1870:GOSUB 1890:RETURN
600 GOSUB 700:C=0:GOSUB 680:IF C=0 THEN
     RETURN ELSE NR=NR+1:LOCATE 18,1:PR
    INT"ENTER ITEM DESCRIPTION:":IN$=""
    :ROW=20:COL=5:MAXLEN=25:GOSUB 2030:
    IF IN$="" THEN NR=NR-1:RETURN ELSE
    A$(NR)=IN$
610 LOCATE 22,5:PRINT"ENTER ITEM AMOUNT
    : $";:ROW=22:COL=25:MAXLEN=6:IN$=""
    :GOSUB 1920:IF IN$="" THEN 610 ELSE
     A=VAL(IN$)
620 A$(NR)=CHR$(200+C)+CHR$(100+LEN(A$(
    NR))+2)+A$(NR)+STR$(A):GOSUB 820:IF
     K$="Y" OR K$="y" THEN 590 ELSE RET
    URN
630 ' CHANGE ITEM *****
640 CLS:ROW=2:T$="CHANGE ITEMS":GOSUB 1
    870:
650 GOSUB 700:C=0:GOSUB 680:IF C=0 THEN
     RETURN
660 GOSUB 840:IF K$="Y" OR K$="y" THEN
    640 ELSE RETURN
670 ' CATEGORY NUMBER *****
680 LOCATE 22,1:PRINT SPACE$(39):LOCATE
     24,1:PRINT SPACE$(39);:LOCATE 17,1
    :PRINT "ENTER CATEGORY #:";:IN$="":
    MAXLEN=2:ROW=17:COL=18:GOSUB 1920:I
    F IN$="" THEN RETURN ELSE C=VAL(IN$
    ):IF C=0 OR C>NC THEN 680 ELSE RETU
    RN
690 ' LIST CATEGORIES *****
700 LOCATE 7,1,0:CNT=0:FOR Z=1 TO NC:PR
    INT MID$(SP$+STR$(Z),LEN(STR$(Z)),2
    );") ";N$(Z);TAB(20);B(Z):IF Z=NC T
    HEN RETURN
710 CNT=CNT+1:IF CNT/8<>INT(CNT/8) THEN
     740
720 ROW=22:T$="PRESS SPACE BAR FOR MORE
    ":GOSUB 1870:LOCATE 24,10:PRINT "PR
    ESS ";ENT$;" TO CONTINUE";:GOSUB 18
    50
730 IF K$<>CR$ AND K$<>SP$ THEN 720 ELS
    E IF K$=CR$ THEN RETURN ELSE FOR I=
    7 TO 14:LOCATE I,1:PRINT SPACE$(39)
    :NEXT:CNT=0:LOCATE 7,1
740 NEXT Z:RETURN
750 ' GET CATEGORY INFO *****
760 IF C=1 THEN 780
770 ROW=22:T$=SPACE$(39):GOSUB 1870:LOC
    ATE 24,1:PRINT SPACE$(39);:LOCATE 1
    8,1:PRINT "ENTER CATEGORY HEADING:
    ":IN$="":MAXLEN=25:ROW=20:COL=5:GOS
    UB 2030:IF IN$="" THEN RETURN ELSE
    IF (MD2=3) AND IN$="/D/" THEN GOSUB
     1050:RETURN ELSE N$(C)=IN$
780 LOCATE 22,5:PRINT "ENTER BUDGET AMO
    UNT: $";:ROW=22:COL=27:MAXLEN=6:IN$
    ="":GOSUB 1920:IF IN$="" THEN 780 E
    LSE B(C)=VAL(IN$):RETURN
790 ' ANOTHER CATEGORY PROMPT *****
800 LOCATE 24,5:PRINT "ANOTHER CATEGORY
    ? (Y/N)";:K$="":WHILE NOT(K$="Y" OR
     K$="y" OR K$="N" OR K$="n"):LOCATE
     24,29:GOSUB 1850:WEND:RETURN
810 ' ANOTHER ITEM PROMPT *****
820 LOCATE 24,5:PRINT "ANOTHER ITEM? (Y
    /N)";:K$="":WHILE NOT(K$="Y" OR K$=
    "y" OR K$="N" OR K$="n"):LOCATE 24,
    26:GOSUB 1850:WEND:RETURN
830 'DISPLAY ITEMS AND CHANGE *****
840 TZ=0:GOSUB 990
850 FOR Z=1 TO NR:IF ASC(A$(Z))-200<>C
    THEN 870
860 TX=ASC(MID$(A$(Z),2,1))-100:PRINT S
    TR$(Z);TAB(3);")";MID$(A$(Z),3,TX-2
    );TAB(30);VAL(MID$(A$(Z),TX+1,LEN(A
    $(Z))-TX)):CNT=CNT+1
870 IF CNT=0 THEN 970 ELSE IF CNT<5 AND
     Z<>NR THEN 970
880 ROW=22:T$="PRESS SPACE BAR FOR MORE
    ":GOSUB 1870:LOCATE 24,10:PRINT"PRE
    SS ";ENT$;" TO SELECT";:GOSUB 1850:
    LOCATE 7,1,0
890 IF K$<>CR$ AND K$<>SP$ THEN 880 ELS
    E IF K$=SP$ THEN 960
900 GOSUB 1000:LOCATE 17,1,0:PRINT "ENT
    ER ITEM #:";:IN$="":MAXLEN=3:ROW=17
    :COL=14:GOSUB 1920:IF IN$="" THEN 9
    40 ELSE A=VAL(IN$):IF A=O THEN RETU
    RN ELSE IF ASC(A$(A))-200<>C THEN G
    OSUB 980:GOTO 970
```

Continued

## BUDGETRON *Continued*

```
910 LOCATE 18,1,0:PRINT "ENTER ITEM DESCRIPTION: ":ROW=20:COL=5:MAXLEN=25:IN$="":GOSUB 2030:IF IN$="" THEN 940 ELSE TEMP$=IN$:IF TEMP$="/D/" THEN GOSUB 1020:RETURN
920 LOCATE 22,5,0:PRINT"ENTER ITEM AMOUNT:$":ROW=22:COL=24:IN$="":GOSUB 1920:IF IN$="" THEN 920
930 A$(A)=CHR$(200+C)+CHR$(LEN(TEMP$)+102)+TEMP$+STR$(VAL(IN$))
940 LOCATE 24,10,0:PRINT "ANOTHER ITEM? (Y/N) ";:K$="":WHILE NOT(K$="Y" OR K$="y" OR K$="N" OR K$="n"):LOCATE 24,28:GOSUB 1850:WEND
950 IF K$="N" OR K$="n" THEN RETURN ELSE Z=A-1
960 GOSUB 990
970 NEXT Z:ROW=10:T$="NO MORE IN CATEGORY":GOSUB 1870:ROW=12:T$="WANT ANOTHER? (Y/N) ":GOSUB 1870:K$="":WHILE NOT(K$="Y" OR K$="y" OR K$="N" OR K$="n"):LOCATE 24,28:GOSUB 1850:WEND:RETURN
980 ROW=22:T$="ITEM NOT IN THIS CATEGORY":GOSUB 1870:GOSUB 1890:T$=SPACE$(39):ROW=17:GOSUB 1870:GOSUB 1000:RETURN
990 CLS:ROW=2:T$="CHANGE ITEMS":GOSUB 1870:CNT=0:LOCATE 7,1,0:RETURN
1000 ROW=22:T$=SPACE$(39):GOSUB 1870:LOCATE 24,1:PRINT SPACE$(39);:RETURN
1010 ' DELETE ITEM *****
1020 FOR I=A TO NR:IF I=NR THEN 1030 ELSE A$(I)=A$(I+1):NEXT
1030 NR=NR-1:RETURN
1040 'DELETE A CATEGORY *****
1050 A=1:WHILE NR>=A:IF ASC(A$(A))-200=C THEN GOSUB 1020:A=A-1
1060 A=A+1:WEND
1070 IF NC=2 THEN NC=1:N$(2)="":RETURN
1080 FOR I=C TO NC-1:N$(I)=N$(I+1):FOR J=1 TO NR:IF ASC(A$(J))-200=I+1 THEN A$(J)=CHR$(I+200)+RIGHT$(A$(J),LEN(A$(J))-1)
1090 NEXT J:NEXT I:N$(NC)="":NC=NC-1:RETURN
1100 'REPORTS *****
1110 CLS:RESTORE 2290:ROW=2:T$=TI$:GOSUB 1870:ROW=4:T$="REPORTS":GOSUB 1870:FOR I= 8 TO 14 STEP 2:LOCATE I,8:READ MEN$:PRINT I/2-3;") ";MEN$:NEXT I:LOCATE 17,8,1:PRINT "YOUR CHOICE:"
1120 K$="":WHILE K$<"1" OR K$>"4":LOCATE 17,21,1:GOSUB 1850:WEND:MD1=VAL(K$):IF MD1=4 THEN RETURN
1130 CLS:ROW=2:T$=TI$:GOSUB 1870:ROW=4:T$="REPORTS":GOSUB 1870:FOR I= 8 TO 10 STEP 2:LOCATE I,8:READ MEN$:PRINT I/2-3;") ";MEN$:NEXT I:LOCATE 13,8,1:PRINT "YOUR CHOICE:"
1140 K$="":WHILE NOT(K$="1" OR K$="2" OR K$=CR$):LOCATE 13,21,1:GOSUB 1850:WEND:IF K$=CR$ THEN RETURN ELSE IF K$="1" THEN DEV$="SCRN:":F=0: ELSE DEV$="LPT1:":F=1
1150 GOSUB 2130:OPEN DEV$ FOR OUTPUT AS #2:ON MD1 GOSUB 1170,1210,1310:CLOSE #2:GOTO 1110
1160 'PRINT ALL CATEGORIES AND ITEMS *****
1170 FOR C=1 TO NC:IF F=0 THEN CLS
1180 GOSUB 2160:IF F=0 THEN GOSUB 1890
1190 NEXT C:GOSUB 1210:RETURN
1200 'PRINT MONTHLY SUMMARY *****
1210 IF F=0 THEN CLS
1220 PRINT #2,CR$+"MONTHLY BUDGET SUMMARY"+CR$+CR$
1230 PRINT #2,"BUDGET INCOME:";TAB(20);"$";B(1)
1240 PRINT #2,"ACTUAL INCOME:";TAB(20);"$";E(1);CR$+CR$
1250 PRINT #2,"ACTUAL EXPENSES:";TAB(20);"$";ETOT
1260 PRINT #2,"CASH BALANCE:";TAB(20);"$";E(1)-ETOT;CR$
1270 PRINT #2,"BUDGETED EXPENSES:"TAB(20);"$";BTOT
1280 PRINT #2,"BUDGET DEVIATION:";TAB(20);"$";BTOT-ETOT
1290 GOSUB 1890:RETURN
1300 'PRINT CHART *****
1310 IF F=0 THEN CLS
1320 GOSUB 1470:IF F=0 THEN RNG=CNT/39 ELSE RNG=CNT/80
1330 FOR I=2 TO NC:PRINT #2,N$(I):PRINT #2,"BUDGETED: $";B(I);TAB(20);"EXPENDED: $";E(I)
1340 FOR J=1 TO INT(B(I)/RNG+.50001):PRINT #2,"B";:NEXT J:PRINT #2,"":IF E(I)>B(I) THEN EC$="*" ELSE EC$="E"
1350 FOR J=1 TO INT(E(I)/RNG+.50001):PRINT #2,EC$;:NEXT J:PRINT #2,CR$
1360 IF F=0 AND INT((I-1)/4)=(I-1)/4 THEN GOSUB 1890:CLS
1370 NEXT I:GOSUB 1890:RETURN
1380 ' BAR CHART *****
1390 CLS:T$="GETTING DATA. . .":ROW=10:GOSUB 1870:GOSUB 2130:SCREEN 1:COLOR 0,0:ENTDR$="E3 D1 G2 D1 E2 D1 G1 D1 F1 U2 E1 R6 U3 L1 D2 L1 U2":CNT=0:GOSUB 1470
1400 RNG=240/CNT:FOR Z=2 TO MAXCAT:IF B(Z)=0 AND N$(Z)="" THEN 1460 ELSE GOSUB 1510:IF PASS THEN PRINT Z;") ";N$(Z) ELSE PASS=2:GOTO 1450
1410 GOSUB 1560:IF CF THEN LINE (0,RS)-(CF,RS+4),1,BF
1420 IF E(Z)>B(Z) THEN COLR=2 ELSE COLR=1
1430 GOSUB 1580:GOSUB 1600:IF CF THEN LINE (0,RS)-(CF,RS+4),COLR,BF
1440 GOSUB 1620
1450 NEXT
1460 GOSUB 1640:SCREEN 0:RETURN
1470 FOR Z=2 TO MAXCAT:IF B(Z)<=CNT THEN 1480 ELSE CNT=B(Z)
1480 IF E(Z)<=CNT THEN 1490 ELSE CNT=E(Z)
1490 NEXT Z:RETURN
1500 ' SELECT TEXT LOCATION *****
1510 IF Z>9 THEN 1530
1520 ROW=(Z-2)*3+1:LOCATE ROW,1:PASS=1:RETURN
1530 IF PASS=1 THEN GOSUB 1640:Z=Z-1:CLS:PASS=0:RETURN
1540 ROW=(Z-10)*3+1:LOCATE ROW,1:RETURN
1550 ' SELECT B(Z) BAR LOCATION *****
1560 RS=ROW*8+2:CF=RNG*B(Z):RETURN
1570 PRINT B(Z) AMOUNT *****
1580 LOCATE ROW+1,32:PRINT "$";B(Z);:RETURN
1590 ' SELECT E(Z) BAR LOCATION *****
1600 RS=(ROW+1)*8+2:CF=RNG*E(Z):RETURN
1610 ' PRINT E(Z) AMOUNT *****
1620 LOCATE ROW+2,32:PRINT "$";E(Z);:RETURN
1630 ' ENTER PROMPT *****
1640 LOCATE 25,10,0:PRINT "PRESS ";:PSET (120,195):DRAW ENTDR$:LOCATE 25,19:PRINT "TO CONTINUE";
1650 K$="":WHILE K$<>CR$:K$=INKEY$:WEND:LOCATE 25,1:PRINT SPACE$(39);:RETURN
1660 'FILES ROUTINE *****
1670 CLS:RESTORE 2280:ROW=2:T$=TI$:GOSUB 1870;ROW=4:T$="FILES":GOSUB 1870:FOR I= 8 TO 12 STEP 2:LOCATE I,8:READ MEN$:PRINT I/2-3;") ";MEN$:NEXT I:LOCATE 15,8,1:PRINT "YOUR CHOICE:"
1680 K$="":WHILE K$<"1" OR K$>"3":LOCATE 15,21,1:GOSUB 1850:WEND:MD1=VAL(K$)
1690 IF MD1=3 THEN RETURN
1700 ON ERROR GOTO 2310:ON MD1 GOSUB 1720,1770:ON ERROR GOTO 0:IF DONE=1 THEN RETURN ELSE 1670
1710 ' LOAD DATA *****
1720 CLS:ROW=2:T$="LOAD FILES":GOSUB 1870:GOSUB 1820:IF FI$="" THEN RETURN
1730 OPEN FI$ FOR INPUT AS #2:LOCATE 8,5,0:PRINT"LOADING FILE. . ."
1740 INPUT #2,NR,NC:FOR Z=1 TO MAXCAT:INPUT #2,B(Z),E(Z),N$(Z):NEXT:FOR Z=1 TO NR:INPUT #2,A$(Z):NEXT:CLOSE #2:DONE=1:RETURN
1750 RETURN
1760 ' SAVE DATA *****
1770 CLS:ROW=2:T$="SAVE FILES":GOSUB 1870:GOSUB 1820:IF FI$="" THEN RETURN
1780 OPEN FI$ FOR OUTPUT AS #2:LOCATE 8,5,0:PRINT"SAVING FILE. . ."
1790 WRITE #2,NR,NC:FOR Z=1 TO MAXCAT:WRITE #2,B(Z),E(Z),N$(Z):NEXT:FOR Z=1 TO NR:WRITE #2,A$(Z):NEXT:CLOSE #2:DONE=1:RETURN
1800 RETURN
1810 ' GET FILENAME *****
1820 LOCATE 6,1:PRINT "INPUT FILE NAME:[          ]";:ROW=6:COL=18:MAXLEN=10:IN$="":GOSUB 2030:FI$=IN$
1830 IF FI$="" THEN RETURN ELSE IF MID$(FI$,2,1)<>":" THEN FI$=LEFT$(FI$,8)+".BDG":RETURN ELSE RETURN
1840 'INKEY SUBROUTINE *****
1850 K$="":WHILE K$="":K$=INKEY$:WEND:RETURN
1860 'CENTER SCREEN TITLE *****
1870 LOCATE ROW,(40-LEN(T$))/2,0:PRINT T$:RETURN
1880 ' PRESS ENT$ TO CONTINUE *****
1890 LOCATE 24,10,0:PRINT "PRESS "+ENT$+" TO CONTINUE";
1900 GOSUB 1850:IF K$<>CR$ THEN 1900 ELSE RETURN
1910 'NUMERIC INPUT ROUTINE *****
1920 PT=1
1930 LOCATE ROW,COL,0:PRINT IN$;SPACE$(MAXLEN-LEN(IN$));:LOCATE ROW,COL+(PT-1),1:GOSUB 1850
1940 IF K$=CR$ THEN RETURN
```

*Continued*

## BUDGETRON *Continued*

IBM PC & IBM PCjr

```
1950 IF K$>="0" AND K$<="9" THEN IN$=LEF
     T$(IN$,PT-1)+K$+MID$(IN$,PT+1):PT=P
     T+1:IF PT>MAXLEN THEN PT=MAXLEN:GOT
     O 1930 ELSE 1930
1960 IF K$=BS$ AND PT>1 THEN IN$=LEFT$(I
     N$,PT-2)+MID$(IN$,PT):PT=PT-1:GOTO
     1930
1970 IF K$=DEL$ THEN IN$=LEFT$(IN$,PT-1)
     +MID$(IN$,PT+1):GOTO 1930
1980 IF K$=INS$ AND LEN(IN$)<MAXLEN THEN
      IN$=LEFT$(IN$,PT-1)+SP$+MID$(IN$,P
     T):GOTO 1930
1990 IF K$=RT$ AND LEN(IN$)>=PT THEN PT=
     PT+1:IF PT>MAXLEN THEN PT=MAXLEN:BE
     EP:GOTO 1930 ELSE 1930
2000 IF K$=LT$ AND LEN(IN$)>1 THEN PT=PT
     -1:IF PT<1 THEN PT=1 :GOTO 1930 ELS
     E 1930
2010 GOTO 1930
2020 ' DESCRIPTION INPUT ROUTINE *****
2030 PT=1
2040 LOCATE ROW,COL,0:PRINT IN$;SPACE$(M
     AXLEN-LEN(IN$));:LOCATE ROW,COL+(PT
     -1),1:GOSUB 1850
2050 IF K$=CR$ THEN RETURN
2060 IF K$>=SP$ AND K$<="~" AND K$<>CHR$
     (34) AND K$<>"," THEN IN$=LEFT$(IN$
     ,PT-1)+K$+MID$(IN$,PT+1):PT=PT+1:IF
      PT>MAXLEN THEN PT=MAXLEN:GOTO 2040
      ELSE 2040
2070 IF K$=BS$ AND PT>1 THEN IN$=LEFT$(I
     N$,PT-2)+MID$(IN$,PT):PT=PT-1:GOTO
     2040
2080 IF K$=DEL$ THEN IN$=LEFT$(IN$,PT-1)
     +MID$(IN$,PT+1):GOTO 2040
2090 IF K$=INS$ AND LEN(IN$)<MAXLEN THEN
      IN$=LEFT$(IN$,PT-1)+SP$+MID$(IN$,P
     T):GOTO 2040
2100 IF K$=RT$ AND LEN(IN$)>=PT THEN PT=
     PT+1:IF PT>MAXLEN THEN PT=MAXLEN:BE
     EP:GOTO 2040 ELSE 2040
2110 IF K$=LT$ AND LEN(IN$)>1 THEN PT=PT
     -1:IF PT<1 THEN PT=1 :GOTO 2040 ELS
     E 2040
2120 GOTO 2040
2130 FOR Z=1 TO MAXCAT:E(Z)=0:NEXT:FOR Z
     =1 TO NR:TX=ASC(MID$(A$(Z),2,1))-10
     0:R=ASC(A$(Z))-200:E(R)=E(R)+VAL(MI
     D$(A$(Z),TX+1,LEN(A$(Z))-TX)):NEXT
     Z
2140 BTOT=0:ETOT=0:FOR Z=2 TO MAXCAT:BTO
     T=BTOT+B(Z):ETOT=ETOT+E(Z):NEXT Z:R
     ETURN
2150 'PRINT CATEGORIES *****
2160 IF F=0 THEN CLS:CNT=0
2170 PRINT #2,N$(C);TAB(13);"$";B(C)
2180 FOR I=1 TO NR:IF ASC(A$(I))-200<>C
     THEN 2200 ELSE TX=ASC(MID$(A$(I),2,
     1))-100:PRINT #2,TAB(3);MID$(A$(I),
     3,TX-2);TAB(30);"$";MID$(A$(I),TX+1
     ,LEN(A$(I))-TX)
2190 IF F=1 THEN 2200 ELSE CNT=CNT+1:IF
     CNT<16 THEN 2200 ELSE GOSUB 1890
2200 NEXT I:PRINT #2,TAB(29);"--------":
     PRINT #2,TAB(30);"$";(E(C)):RETURN
2210 'EXIT ROUTINE *****
2220 CLS:LOCATE 10,1,0:PRINT "ARE YOU SU
     RE YOU WANT TO EXIT NOW? (Y/N)":K$=
     "":WHILE NOT(K$="Y" OR K$="y" OR K$
     ="N" OR K$="n"):LOCATE 10,36:GOSUB
     1850:WEND
2230 IF K$="Y" OR K$="y" THEN CLS:PRINT
     "BYE":KEY ON:END ELSE RETURN
2240 DATA WORK WITH BUDGET,REPORTS,BAR C
     HART,FILES,EXIT PROGRAM
2250 DATA BUDGET CATEGORIES,INCOME AND E
     XPENSES,EXIT
2260 DATA INITIALIZE BUDGET,ADD A CATEGO
     RY,CHANGE A CATEGORY,EXIT
2270 DATA INITIALIZE ITEMS,ADD ITEM,CHAN
     GE ITEM,EXIT
2280 DATA LOAD DATA,SAVE DATA,EXIT
2290 DATA ALL CATEGORIES AND ITEMS,MONTH
     LY SUMMARY,PRINT CHART,EXIT,SCREEN,
     PRINTER
2300 ' ERROR ROUTINES *****
2310 CLOSE:LOCATE 25,1,0:R=ERR:L=ERL:FOR
      Z=1 TO 14:IF ERCD(Z)=R THEN 2330
2320 NEXT:PRINT "ERROR #";R;" IN LINE #"
     ;L;:GOTO 2340
2330 PRINT ERM$(Z);"-- #";R;
2340 SOUND 110,20:FOR TD=1 TO 4000:NEXT:
     LOCATE 25,1:PRINT SPACE$(39);:IF MD
     1=1 THEN RESUME 1720 ELSE RESUME 17
     70
2350 DATA 64,BAD FILE NAME,69,COMMUNICAT
     IONS BUFFER OVERFLOW,25,DEVICE FAUL
     T,57,DEVICE I/O ERROR,24,DEVICE TIM
     EOUT,68,DEVICE UNAVAILABLE,61,DISKE
     TTE IS FULL,72,DISK MEDIA ERROR,71,
     DISK NOT READY,70,THIS DISK IS WRIT
     E PROTECTED
2360 DATA 53,FILE IS NOT ON THE DISK,14,
     DATA STORAGE AREA FULL--START NEW F
     ILE,67,TOO MANY FILES ON THIS DISK,
     52,BAD FILE NUMBER OR NAME
```

## BUDGETRON

TI-99/4A

```
100 REM ************
110 REM * BUDGETRON *
120 REM ************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM BY SCOTT WILLIAMS
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 5.3.1
180 REM TI BASIC OR
190 REM TI EXTENDED BASIC
200 REM
210 OPTION BASE 1
220 DIM A$(35),B(16),E(16),N$(16)
230 N$(1)="$$ INCOME $$"
240 NC=1
250 MAXDATA=35
260 MAXCAT=16
270 FOR Z=1 TO 8
280 READ Q$
290 CALL CHAR(127+Z,Q$)
300 CALL CHAR(135+Z,Q$)
310 NEXT Z
320 CALL COLOR(13,2,1)
330 CALL COLOR(14,7,1)
340 CALL CLEAR
350 PRINT TAB(10);"BUDGETRON"::::::::"
       PRESS ENTER TO START"
360 GOSUB 3980
370 CALL CLEAR
380 PRINT TAB(10);"BUDGETRON"::TAB(10);
    "MAIN MENU"::::
390 PRINT "1) WORK WITH BUDGET"::"2) RE
    PORTS"::"3) BAR CHART"::"4) FILES":
    :"5) EXIT PROGRAM":::::::
400 GOSUB 3980
410 IF (K<49)+(K>53)THEN 400
420 CALL CLEAR
430 MD=K-48
440 ON MD GOTO 450,2650,2260,3540,4370
450 CALL CLEAR
460 PRINT TAB(6);"WORK WITH BUDGET":::"
    1) BUDGET CATEGORIES"::"2) INCOME A
    ND EXPENSES"::"3) EXIT"::::::::
470 GOSUB 3980
480 IF (K<49)+(K>51)THEN 470
490 MD1=K-48
500 IF MD1=3 THEN 370
510 ON MD1 GOTO 590,520
520 CALL CLEAR
530 PRINT "    INCOME AND EXPENSES":::"
    1) INITIALIZE ITEMS"::"2) ADD ITEM"
    ::"3) CHANGE ITEM"::"4) EXIT":::::
540 GOSUB 3980
550 MD2=K-48
560 IF MD2=4 THEN 450
570 ON MD2 GOSUB 3890,1470,1470
580 GOTO 520
590 CALL CLEAR
600 PRINT TAB(5);"BUDGET CATEGORIES":::
    "1) INITIALIZE BUDGET"::"2) ADD A C
    ATEGORY"::"3) CHANGE A CATEGORY"
610 PRINT :"4) EXIT":::::::
620 GOSUB 3980
630 IF (K<49)+(K>52)THEN 620
640 IF K=52 THEN 450
650 MD2=K-48
660 ON MD2 GOSUB 680,1080,800
670 GOTO 590
680 PRINT "ARE YOU SURE?? THIS WILL":"E
    RASE ALL BUDGET DATA (Y/N)"
690 GOSUB 3980
700 IF (K=89)+(K=121)THEN 720
710 RETURN
720 FOR Z=2 TO MAXCAT
730 N$(Z)=""
740 B(Z)=0
750 E(Z)=0
760 NEXT Z
770 B(1)=0
780 NC=1
790 RETURN
800 CALL CLEAR
810 GOSUB 1220
820 GOSUB 1370
830 IF ((C>1)*(C<=NC)*(MD2=2))+((C>0)*(
    C<=NC)*(MD2=3))THEN 850
840 RETURN
850 IF C=1 THEN 990
860 PRINT "ENTER CATEGORY HEADING:"
870 INPUT "":N$(C)
880 IF N$(C)<>"" THEN 920
890 IF MD2=3 THEN 910
900 NC=NC-1
910 RETURN
920 IF N$(C)<>"/D/" THEN 950
930 GOSUB 4520
940 GOTO 1020
950 IF LEN(N$(C))<13 THEN 990
```

Continued

## BUDGETRON *Continued*

```
 960 N$(C)=SEG$(N$(C),1,12)
 970 PRINT "INPUT TOO LONG!":"TRUNCATED
     TO:":N$(C)
 980 GOSUB 4330
 990 INPUT "ENTER BUDGET AMOUNT:":Q$
1000 GOSUB 4170
1010 B(C)=A
1020 PRINT "ANOTHER CATEGORY (Y/N)?"
1030 GOSUB 3980
1040 IF (K<>89)*(K<>121)THEN 1060
1050 IF MD2=2 THEN 1080 ELSE 800
1060 IF (K<>78)*(K<>110)THEN 1030
1070 RETURN
1080 IF NC<MAXCAT THEN 1120
1090 PRINT "CATEGORIES ARE FULL"::
1100 GOSUB 4330
1110 RETURN
1120 CALL CLEAR
1130 IF (NC<>1)+(B(1)<>0)THEN 1180
1140 INPUT "ENTER BUDGET INCOME:":Q$
1150 GOSUB 4170
1160 B(1)=A
1170 CALL CLEAR
1180 GOSUB 1220
1190 NC=NC+1
1200 C=NC
1210 GOTO 850
1220 CNT=0
1230 FOR Z=1 TO NC
1240 PRINT SEG$(" "&STR$(Z),LEN(STR$(Z))
     ,2);") ";N$(Z);TAB(18);B(Z)
1250 IF Z=NC THEN 1350
1260 CNT=CNT+1
1270 IF CNT/8<>INT(CNT/8)THEN 1340
1280 PRINT ::"PRESS SPACE BAR FOR MORE":
     "PRESS ENTER TO CONTINUE"
1290 GOSUB 3980
1300 IF K=32 THEN 1330
1310 IF K<>13 THEN 1290
1320 GOTO 1350
1330 CNT=0
1340 NEXT Z
1350 PRINT :::
1360 RETURN
1370 INPUT "ENTER CATEGORY #":Q$
1380 IF (Q$="")+(LEN(Q$)>2)THEN 1420
1390 IF (ASC(Q$)<49)+(ASC(Q$)>57)THEN 14
     20
1400 IF LEN(Q$)=1 THEN 1440
1410 IF (ASC(SEG$(Q$,2,1))>47)*(ASC(SEG$
     (Q$,2,1))<51)THEN 1440
1420 C=0
1430 RETURN
1440 C=VAL(Q$)
1450 IF (C<1)+(C>MAXCAT)THEN 1420
1460 RETURN
1470 CALL CLEAR
1480 IF ((NR<MAXDATA)*(MD2=2))+(MD2=3)TH
     EN 1520
1490 PRINT ::::"THE FILE IS FULL":"THERE
      ARE";MAXDATA;" ITEMS"::::
1500 GOSUB 4330
1510 RETURN
1520 GOSUB 1220
1530 GOSUB 1370
1540 IF C>0 THEN 1560
1550 RETURN
1560 IF MD2=3 THEN 1710
1570 NR=NR+1
1580 INPUT "ITEM DESCRIPTION:
     ":A$(NR)
1590 IF LEN(A$(NR))<13 THEN 1630
1600 A$(NR)=SEG$(A$(NR),1,12)
1610 PRINT "ITEM DESCRIPTION TOO LONG!":
     "TRUNCATED TO:":A$(NR)
1620 GOSUB 4330
1630 INPUT "ITEM AMOUNT:":Q$
1640 GOSUB 4170
1650 A$(NR)=CHR$(200+C)&CHR$(100+LEN(A$(
     NR))+2)&A$(NR)&STR$(A)
1660 PRINT "ANOTHER (Y/N)?"
1670 GOSUB 3980
1680 IF (K=89)+(K=121)THEN 1470
1690 IF (K<>78)*(K<>110)THEN 1670
1700 RETURN
1710 CALL CLEAR
1720 CNT=0
1730 TZ=0
1740 FOR Z=1 TO NR
1750 IF ASC(A$(Z))-200=C THEN 1770
1760 IF Z=NR THEN 1830 ELSE 2200
1770 TX=ASC(SEG$(A$(Z),2,1))-100
1780 PRINT STR$(Z);TAB(3);")";SEG$(A$(Z)
     ,3,TX-2);TAB(18);VAL(SEG$(A$(Z),TX+
     1,LEN(A$(Z))-TX))
1790 PRINT
1800 CNT=CNT+1
1810 IF (CNT<5)*(Z<>NR) THEN 2200
1820 IF CNT=0 THEN 2200
1830 PRINT "PRESS SPACE BAR FOR MORE":"P
     RESS ENTER TO SELECT ITEM"
1840 GOSUB 3980
1850 IF K=32 THEN 2170
1860 IF K<>13 THEN 1840
1870 INPUT "ITEM NUMBER:":Q$
1880 GOSUB 4170
1890 IF A=0 THEN 1470
1900 IF ASC(A$(A))-200<>C THEN 2120
1910 PRINT "ENTER DESCRIPTION:"
1920 INPUT "":Q$
1930 IF Q$<>"" THEN 1950
1940 RETURN
1950 IF Q$<>"/D/" THEN 1980
1960 GOSUB 4450
1970 GOTO 2070
1980 IF LEN(Q$)<13 THEN 2020
1990 Q$=SEG$(Q$,1,12)
2000 PRINT "ENTRY TRUNCATED TO:":Q$
2010 GOSUB 4330
2020 R=A
2030 A$(R)=CHR$(200+C)&CHR$(LEN(Q$)+102)
     &Q$
2040 INPUT "ENTER VALUE:":Q$
2050 GOSUB 4170
2060 A$(R)=A$(R)&STR$(A)
2070 PRINT "ANOTHER (Y/N)?"
2080 GOSUB 3980
2090 IF (K=89)+(K=121)THEN 2170
2100 IF (K<>78)*(K<>110)THEN 2080
2110 GOTO 1470
2120 PRINT "ITEM IS NOT IN THIS CATEGORY
     "::
2130 GOSUB 4330
2140 Z=TZ
2150 CNT=0
2160 GOTO 2200
2170 CNT=0
2180 CALL CLEAR
2190 TZ=Z
2200 NEXT Z
2210 PRINT "CHANGE ANOTHER (Y/N)?"
2220 GOSUB 3980
2230 IF (K=89)+(K=121)THEN 1470
2240 IF (K<>78)*(K<>110)THEN 2220
2250 RETURN
2260 CALL CLEAR
2270 PRINT "GETTING DATA..."
2280 GOSUB 4020
2290 CALL CLEAR
2300 CNT=0
2310 GOSUB 2330
2320 GOTO 2400
2330 FOR Z=2 TO MAXCAT
2340 IF B(Z)<=CNT THEN 2360
2350 CNT=B(Z)
2360 IF E(Z)<=CNT THEN 2380
2370 CNT=E(Z)
2380 NEXT Z
2390 RETURN
2400 RNG=(CNT+2)/128
2410 CNT=0
2420 FOR Z=2 TO MAXCAT
2430 IF (B(Z)=0)*(N$(Z)="")THEN 2480
2440 PRINT N$(Z)
2450 CALL HCHAR(23,15,135,INT(B(Z)/(RNG*
     8)))
2460 CALL HCHAR(23,15+INT(B(Z)/(RNG*8)),
     INT(128+((B(Z)/RNG)-INT(B(Z)/(RNG*8
     ))*8))))
2470 PRINT
2480 IF E(Z)=0 THEN 2550
2490 IF E(Z)<=B(Z)THEN 2520
2500 CO=136
2510 GOTO 2530
2520 CO=128
2530 CALL HCHAR(23,15,CO+7,INT(E(Z)/(RNG
     *8)))
2540 CALL HCHAR(23,15+INT(E(Z)/(RNG*8)),
     INT(CO+((E(Z)/RNG)-INT(E(Z)/(RNG*8)
     )*8))))
2550 CNT=CNT+1
2560 IF (CNT<6)*(Z<>NC)THEN 2620
2570 PRINT ::"PRESS ENTER TO CONTINUE"
2580 GOSUB 3980
2590 IF Z=NC THEN 370
2600 CNT=0
2610 CALL CLEAR
2620 PRINT :
2630 NEXT Z
2640 GOTO 370
2650 CALL CLEAR
2660 PRINT TAB(10);"REPORTS":::
2670 PRINT "1) ALL CATEGORIES AND ITEMS"
     ::"2) MONTHLY SUMMARY"::"3) PRINT C
     HART"::"4) EXIT":::::::::
2680 GOSUB 3980
2690 IF (K<49)+(K>52)THEN 2680
2700 PM=K-48
2710 IF PM=4 THEN 370
2720 F=0
2730 PRINT "1) SCREEN"::"2) PRINTER"::
2740 GOSUB 3980
2750 IF (K<49)+(K>50)THEN 2740
2760 IF K=49 THEN 2800
2770 INPUT "ENTER DEVICE:":PRNTR$
2780 F=1
2790 OPEN #F:PRNTR$
2800 GOSUB 4020
2810 ON PM GOSUB 2850,2940,3040
2820 IF F=0 THEN 2650
2830 CLOSE #F
2840 GOTO 2650
```

Continued

## BUDGETRON *Continued*

```
2850 FOR C=1 TO NC
2860 GOSUB 3350
2870 IF F=1 THEN 2910
2880 PRINT ::"NEXT CATEGORY - PRESS ENTE
     R"
2890 GOSUB 3980
2900 CALL CLEAR
2910 NEXT C
2920 GOSUB 2940
2930 RETURN
2940 IF F=1 THEN 2970
2950 CALL CLEAR
2960 PRINT #F::"MONTHLY BUDGET SUMMARY:"
     ::
2970 PRINT #F:"BUDGET INCOME:";TAB(20);B
     (1):"ACTUAL INCOME:";TAB(20);E(1)::
2980 PRINT #F:"ACTUAL EXPENSES:";TAB(20)
     ;ETOT:"CASH BALANCE:";TAB(20);E(1)-
     ETOT
2990 PRINT #F::"BUDGETED EXPENSES:";TAB(
     20);BTOT::"BUDGET DEVIATION";TAB(20
     );BTOT-ETOT
3000 IF F=1 THEN 3030
3010 PRINT :::"PRESS ENTER FOR MENU"::
3020 GOSUB 3980
3030 RETURN
3040 IF F=1 THEN 3060
3050 CALL CLEAR
3060 GOSUB 2330
3070 IF F=1 THEN 3100
3080 RNG=CNT/15
3090 GOTO 3110
3100 RNG=CNT/60
3110 FOR Z=2 TO NC
3120 PRINT #F:TAB(13);"|":N$(Z);TAB(13);
     "|";
3130 FOR X=1 TO INT(B(Z)/RNG+.50001)
3140 PRINT #F:"B";
3150 NEXT X
3160 PRINT #F:TAB(13);"|";
3170 IF E(Z)<=B(Z)THEN 3200
3180 EC$="*"
3190 GOTO 3210
3200 EC$="E"
3210 FOR X=1 TO INT(E(Z)/RNG+.50001)
3220 PRINT #F:EC$;
3230 NEXT X
3240 IF (INT(Z/5)<>Z/5)+(F=1)THEN 3280
3250 PRINT ::"PRESS ENTER TO CONTINUE"
3260 GOSUB 3980
3270 CALL CLEAR
3280 NEXT Z
3290 PRINT #F::::
3300 IF F=1 THEN 3330
3310 PRINT "PRESS ENTER TO CONTINUE"
3320 GOSUB 3980
3330 GOSUB 2940
3340 RETURN
3350 IF F=1 THEN 3380
3360 CALL CLEAR
3370 CNT=0
3380 PRINT #F:N$(C);TAB(13);B(C)
3390 FOR X=1 TO NR
3400 IF ASC(A$(X))-200<>C THEN 3490
3410 TX=ASC(SEG$(A$(X),2,1))-100
3420 PRINT #F:TAB(3);SEG$(A$(X),3,TX-2);
     TAB(18);SEG$(A$(X),TX+1,LEN(A$(X))-
     TX)
3430 IF F=1 THEN 3490
3440 IF (CNT<16)+(F=1)THEN 3490
3450 CNT=0
3460 PRINT ::"PRESS ENTER TO CONTINUE";
3470 GOSUB 3980
3480 CNT=0
3490 NEXT X
3500 PRINT #F:TAB(17);"--------":TAB(18)
     ;STR$(E(C))::
3510 IF F=1 THEN 3530
3520 PRINT #F:""
3530 RETURN
3540 CALL CLEAR
3550 PRINT TAB(11);"FILES":::"1) LOAD DA
     TA"::"2) SAVE DATA"::"3) EXIT":::::
     :
3560 GOSUB 3980
3570 IF (K<49)+(K>51)THEN 3560
3580 IF K=51 THEN 370
3590 IF K=50 THEN 3630
3600 MD1=1
3610 PRINT "** LOAD DATA **":::
3620 GOTO 3650
3630 MD1=2
3640 PRINT "** SAVE DATA **":::
3650 INPUT "ENTER DEVICE & FILE NAME
     OR DEVICE:":FN$
3660 IF FN$<>"" THEN 3680
3670 GOTO 3540
3680 IF MD1=2 THEN 3790
3690 OPEN #2:FN$,INTERNAL,FIXED 64,INPUT
3700 INPUT #2:NR,NC
3710 FOR Z=1 TO MAXCAT
3720 INPUT #2:B(Z),E(Z),N$(Z)
3730 NEXT Z
3740 FOR Z=1 TO NR
3750 INPUT #2:A$(Z)
3760 NEXT Z
3770 CLOSE #2
3780 GOTO 370
3790 OPEN #2:FN$,INTERNAL,FIXED 64,OUTPU
     T
3800 PRINT #2:NR;NC
3810 FOR Z=1 TO MAXCAT
3820 PRINT #2:B(Z);E(Z);N$(Z)
3830 NEXT Z
3840 FOR Z=1 TO NR
3850 PRINT #2:A$(Z)
3860 NEXT Z
3870 CLOSE #2
3880 GOTO 370
3890 PRINT :::"ARE YOU SURE?? THIS WILL"
     :"ERASE ALL ITEMS IN MEMORY!!":"(Y/
     N)"
3900 GOSUB 3980
3910 IF (K=89)+(K=121)THEN 3930
3920 RETURN
3930 FOR Z=1 TO MAXDATA
3940 A$(Z)=""
3950 NEXT Z
3960 NR=0
3970 RETURN
3980 CALL KEY(0,K,S)
3990 IF S=0 THEN 3980
4000 CALL SOUND(1,880,0)
4010 RETURN
4020 FOR Z=1 TO MAXCAT
4030 E(Z)=0
4040 NEXT Z
4050 FOR Z=1 TO NR
4060 TX=ASC(SEG$(A$(Z),2,1))-100
4070 R=ASC(A$(Z))-200
4080 E(R)=E(R)+VAL(SEG$(A$(Z),TX+1,LEN(A
     $(Z))-TX))
4090 NEXT Z
4100 BTOT=0
4110 ETOT=0
4120 FOR Z=2 TO MAXCAT
4130 BTOT=BTOT+B(Z)
4140 ETOT=ETOT+E(Z)
4150 NEXT Z
4160 RETURN
4170 IF Q$="" THEN 4310
4180 FOR NS=1 TO LEN(Q$)
4190 P=POS("0123456789",SEG$(Q$,NS,1),1)
4200 IF P=0 THEN 4230
4210 NEXT NS
4220 GOTO 4250
4230 IF NS=1 THEN 4310
4240 Q$=SEG$(Q$,1,NS-1)
4250 IF LEN(Q$)<7 THEN 4290
4260 Q$=SEG$(Q$,1,6)
4270 PRINT ::"NUMBER TOO BIG! TRUNCATED
     TO":Q$
4280 GOSUB 4330
4290 A=VAL(Q$)
4300 RETURN
4310 A=0
4320 RETURN
4330 CALL SOUND(100,110,0)
4340 FOR TD=1 TO 500
4350 NEXT TD
4360 RETURN
4370 CALL CLEAR
4380 PRINT "ARE YOU SURE YOU WANT TO":"L
     EAVE NOW (Y/N)?"
4390 GOSUB 3980
4400 IF (K<>89)*(K<>121)THEN 370
4410 CALL CLEAR
4420 END
4430 DATA 0080808080808000,00C0C0C0C0C0C
     000,00E0E0E0E0E0E000,00F0F0F0F0F0F0
     00,00F8F8F8F8F8F800,00FCFCFCFCFCFC0
     0
4440 DATA 00FEFEFEFEFEFE00,00FFFFFFFFFFF
     F00
4450 FOR I=A TO NR
4460 IF A$(I+1)="" THEN 4500
4470 A$(I)=A$(I+1)
4480 NEXT I
4490 A$(NR)=""
4500 NR=NR-1
4510 RETURN
4520 FOR DC=C TO NC
4530 IF DC=NC THEN 4550
4540 N$(DC)=N$(DC+1)
4550 NEXT DC
4560 N$(NC)=""
4570 B(NC)=0
4580 NC=NC-1
4590 IF NR>0 THEN 4610
4600 RETURN
4610 DI=1
4620 IF ASC(A$(DI))-200=C THEN 4660
4630 IF ASC(A$(DI))-200<C THEN 4690
4640 A$(DI)=CHR$(ASC(A$(DI))-1)&SEG$(A$(
     DI),2,LEN(A$(DI))-1)
4650 GOTO 4690
4660 A=DI
4670 GOSUB 4450
4680 GOTO 4620
4690 DI=DI+1
4700 IF DI<=NR THEN 4620
4710 RETURN
```

HCM

# OVER-REACTION

```
 100  REM  *****************
 110  REM  * OVER-REACTION *
 120  REM  *****************
 130  REM  COPYRIGHT 1985
 140  REM  EMERALD VALLEY PUBLISHING CO
 150  REM  BY WILLIAM K. BALTHROP
 160  REM  AND THE HCM STAFF
 170  REM  HOME COMPUTER MAGAZINE
 180  REM  VERSION 5.3.1
 190  REM  APPLE // FAMILY APPLESOFT
 200  REM
 210  IF  PEEK (104) = 64 THEN 240
 220  POKE 104,64: POKE 16384,0
 230  PRINT  CHR$ (4);"RUN OVEREACT"
 240  TEXT : HOME :M$ =  CHR$ (13)
 250  DIM S(4,6),TS(4),BT(4),INST(4),CSYS
      (4),BP(4,2),QP(4,2),FG(4)
 260  HCHAR = 2048:SND = 2156:SHFT$ = "!"
       + CHR$ (34) + "#$%&": IF  PEEK (6
      4435) = 6 THEN SHFT$ = "!@#$%^"
 270  FLASH : FOR X = 5 TO 7: VTAB X: HTA
      B 11: PRINT  SPC( 21): NEXT : NORMA
      L
 280  VTAB 6: HTAB 13: PRINT " OVER - REA
      CTION"
 290  VTAB 12: PRINT  TAB( 13)"< INITIALI
      ZING: >"
 300  FOR X = 1 TO 4: READ OP$(X): NEXT
 310  FOR X = 1 TO 4: READ BP(X,1),BP(X,2
      ): NEXT
 320  FOR X = 1 TO 4: READ QP(X,1),QP(X,2
      ): NEXT
 330  FOR K = 768 TO 825: READ P: POKE K,
      P: NEXT K
 340  READ A$: IF A$ <  > "CHECK" THEN 19
      70
 350  IF  PEEK (2048) <  > 32 THEN  FOR K
       = 2048 TO 2271: READ P: POKE K,P:
      NEXT K
 360  FOR X = 1 TO 4: VTAB X * 2 + 10: HT
      AB 10: PRINT X")" LEFT$ (".......
      ...",20 - LEN (OP$(X)));OP$(X): NE
      XT
 370  VTAB 22: PRINT  TAB( 8)"ENTER SKILL
       LEVEL (1-4) ==>";: GET K$: PRINT K
      $:K =  VAL (K$): IF K < 1 OR K > 4
      THEN 370
 380  FOR X = 1 TO 10: PRINT : NEXT : HTA
      B 18 -  LEN (OP$(K)) / 2: PRINT "**
      *";OP$(K);"***": PRINT
 390  ON K GOSUB 950,980,1010,1050
 400  REM  *** DRAW GAME SCREEN ***
 410  HOME : HGR : SCALE= 1: ROT= 0: POKE
       232,0: POKE 233,3
 420  HCOLOR= 2:NB = 4:X1 = 0:Y1 = 0:X2 =
       279:Y2 = 161: GOSUB 1790
 430  FOR X = 1 TO 4:YY = BP(X,1) - 1:XX
      = BP(X,2) - 1
 440  CALL HCHAR,YY,XX,11,7,0
 450  HCOLOR= 1:NB = 2:X1 = XX * 7 - 2:Y1
       = YY * 8 - 2:X2 = X1 + 59:Y2 = Y1
      + 15: GOSUB 1790
 460  NEXT X
 470  CALL HCHAR,18,3,11,17,0
 480  HCOLOR= 5:NB = 2:X1 = 3 * 7 - 2:Y1
      = 18 * 8 - 2:X2 = X1 + 128:Y2 = Y1
      + 15: GOSUB 1790
 490  HCOLOR= 6:NB = 2:X1 = 34 * 7:Y1 = 3
       * 8 - 2:X2 = X1 + 18:Y2 = 152: GOS
      UB 1790
 500  Z = 1: FOR Y = Y1 TO Y2 - 4 STEP 4:
       HPLOT X1,Y TO X1 + 3 + 2 * Z,Y: HP
      LOT X2,Y TO X2 - 3 - 2 * Z,Y:Z =  A
      BS (Z - 1): NEXT
 510  VTAB 21: HTAB 4: PRINT "58        60
           62"; TAB( 34);"STEAM"
 520  HTAB 4: PRINT " <- FREQUENCY  ->
      [TRIP]"; TAB( 32);"PRESSURE": HTAB
      24: PRINT "[SLAM]"; TAB( 32);"( X10
      0 )"
 530  CALL HCHAR,2,37,1,0,0: CALL HCHAR,2
      ,38,0,0,0: FOR Z = 6 TO 9: CALL HCH
      AR,42 - Z * 4,37,Z,0,0: NEXT
 540  FOR X = 4 TO 1 STEP - 1:YY = QP(X,
      1) - 1:XX = QP(X,2) - 1: HCOLOR= 6
      + (X = 1):RY = X: GOSUB 1770
 550  FOR Y1 = 0 TO 1: FOR X1 = 0 TO 2
 560  HCOLOR= 1: DRAW 4 AT (XX * 7) + (28
       * X1) + 12,(YY * 8) + (19 * Y1) +
      1: CALL HCHAR,YY + 2 * Y1 + 1,XX +
      4 * X1 + 3,Y1 * 3 + X1 + 1,0,0
 570  NEXT X1,Y1,X
 580  REM  *** INIT. VARIABLES ***
 590  FOR RY = 1 TO 4: FOR RX = 1 TO 6:S(
      RY,RX) = 8: NEXT RX
 600  CSYS(RY) = 0:INST(RY) = 0:TS(RY) =
      48:BT(RY) = 48:Z = RY: GOSUB 1430:F
      G(RY) = 0
 610  NEXT RY:SOUT = 0:HOUT = 0:BOUT = 0:
      DNE = 0:LOD = 1:SP = 64:TTS = 64:HZ
       = 60:RY = 1
 620  SDAM = 0:CT = .05:LDP = 1:LN = ( RN
      D (1) * .067) + .967:LP = .05
 630  GOSUB 1530: GOSUB 1570
 640  REM  *** MAIN GAME LOOP ***
 650  GOSUB 1610
 660  SCRE = SCRE + (2 - ( ABS (60 - HZ))
      ): IF DNE THEN 1290
 670  FOR Z = 1 TO 4:F = 20: IF CSYS(Z) =
       1 THEN F = FLUX
 680  IF  INST(Z) = 1 AND FG(Z) = 0 THEN I
      NST(Z) = 0: GOSUB 1470:FG(Z) = 1:IN
      ST(Z) = 1: INVERSE : CALL HCHAR,BP(
      Z,1) - 1,BP(Z,2) - 1,10,7,0: NORMAL
 690  GOSUB 1470:BT(Z) = BT(Z) + ((TS(Z)
      +  SIN ((CT + Z * 1.5)) * F) - BT(Z
      )) / 3
 700  GOSUB 1430
 710  NEXT Z
 720  TTS = ((BT(1) + BT(2) + BT(3) + BT(
      4)) / 3) - SDAM
 730  IF TTS <  > SP THEN  GOSUB 1560: GO
      SUB 1530
 740  GOSUB 1600: GOSUB 1570
 750  IF  INT (CT) = CT THEN LDP = LN:LP
      = .05:LN = ( RND (1) * .05) + .975:
        IF PROBS < 4 THEN LN = ( RND (1) *
       .07) + .965
 760  LOD = (LDP * (1 - LP)) + (LN * LP):
      LP = .05 + LP
 770  IF PROBS <  > 1 AND  INT ( RND (1)
      * 15) = 0 THEN 810
 780  CT = CT + .05:CT =  INT (CT * 100 +
       .5) / 100: IF (SOUT) OR (BOUT) OR
       (HOUT) THEN 1120
 790  GOTO 650
 800  REM  ** PROBLEMS OCCURRED **
 810  ON PROBS - 1 GOSUB 830,910,910
 820  GOTO 650
 830  IF  INT ( RND (1) * 5) > 2 THEN 890
 840  C =  INT ( RND (1) * 4) + 1
 850  IF CSYS(C) = 1 THEN  RETURN
 860  CSYS(C) = 1
 870  FLUX =  INT ( RND (1) * 30) + 20
 880  FOR N = 1 TO 3: CALL SND,63: NEXT :
        RETURN
 890  I =  INT ( RND (1) * 4) + 1: IF INS
      T(I) = 0 THEN INST(I) = 1: FOR N =
      1 TO 3: CALL SND,253: NEXT N
 900  RETURN
 910  P =  INT ( RND (1) * 8) + 1: IF PRO
      BS < 4 THEN 930
 920  ON P GOTO 870,870,870,870,890,890,8
      90,940
 930  ON P GOTO 870,870,870,890,890,940,9
      40,940
 940  SDAM = SDAM +  RND (1) * 6 + .2: CA
      LL SND,193: CALL SND,193: RETURN
 950  PRINT "  YOU MAY ENCOUNTER MINOR PRO
      BLEMS WITHIN": PRINT "THE REACTOR;
      HOWEVER, A COOL HEAD SHOULD": PRINT
       " BE ABLE TO KEEP THINGS UNDER CON
      TROL."
 960  PRINT M$;" WATCH FOR OVERHEATING I
      N THE REACTOR": PRINT M$;"      CORE
      S AND CHANGES IN THE LOAD."
 970  SKILL = 750:PROBS = 1: GOTO 1090
 980  PRINT "    IT'S BEEN ONE OF THOSE DAY
      S!   ELEC-": PRINT M$;" TRICAL PROBL
      EMS IN THE COOLING SYSTEM"
 990  PRINT M$;"AND CONTROL PANEL HAVE YO
      U IN A FRENZY.": PRINT M$;"    WATC
      H REACTOR TEMPERATURES !"
1000  SKILL = 700:PROBS = 2: GOTO 1090
1010  PRINT "      YOUR REACTOR IS IN THE H
      EART OF": PRINT M$;" CALIFORNIA EA
      RTHQUAKE COUNTRY !!!!!"
1020  PRINT M$;"    BE PREPARED FOR DRASTI
      C AND SUDDEN": PRINT M$;" CHANGES I
      N THE REACTOR.   IRREVERSIBLE": PRIN
      T M$;"          DAMAGE MAY RESULT.
      "
1030  PRINT M$;"    WATCH THE STEAM PRESSUR
      E ! INTERNAL": PRINT M$;"LEAKS MAY
       DEVELOP IF AN EARTHQUAKE HITS."
1040  SKILL = 600:PROBS = 3: GOTO 1090
1050  PRINT "    OUR WORST FEARS ARE NOW A
       REALITY.": PRINT M$;"NUCLEAR WAR I
      S ON THE VERGE OF ERUPTING."
1060  PRINT "  THE LOAD ON YOUR REACTOR IS
       TERRIFIC": PRINT M$;" AS THE WORLD
       PREPARES FOR WAR.  RIOTS"
1070  PRINT M$;"          ARE INCREASING !
      !"
1080  SKILL = 450:PROBS = 4
1090  FOR X = 1 TO 5: PRINT : NEXT
1100  VTAB 22: HTAB 3: PRINT "PRESS ANY K
      EY TO START THE GAME ==>";: GET A$
1110  FOR X = 1 TO 24: PRINT : NEXT : RET
      URN
1120  IF BOUT THEN 1160
1130  IF SOUT THEN 1200
1140  IF HOUT THEN 1250
1150  GOTO 650
1160  GOSUB 1810: GOSUB 1810:A$ = " MELTD
      OWN !!": GOSUB 1240: GOSUB 1490
1170  TEXT : HOME : VTAB 7
1180  PRINT "THE REACTOR CORE TEMPERATURE
       HAS EXCEEDEDSAFE LIMITS.   MELTDOW
      N HAS BEGUN AND"
1190  PRINT "SURROUNDING AREA MUST BE EVA
      CUATED.":SCRE = SCRE * .2: GOTO 132
      0
```

*Continued*

## OVER-REACTION *Continued*

APPLE // Family

```
1200 GOSUB 1810: GOSUB 1810:A$ = " STEAM
      LEAK !": GOSUB 1240: GOSUB 1490
1210 TEXT : HOME : VTAB 7
1220 PRINT "THE REACTOR STEAM PRESSURE H
     AS GONE. ITCAUSED A MAJOR LEAK OUT
     SIDE THE SYSTEM"
1230 PRINT "AND A MELTDOWN HAS BEGUN. T
     HE AREA SURROUNDING THE PLANT M
     UST BE EVACUATED.":SCRE = SCRE * .2
     : GOTO 1320
1240 HOME : VTAB 22: HTAB 21 - LEN (A$)
      / 2: FLASH : PRINT A$: NORMAL : FO
     R Z = 1 TO 3: CALL SND,1: CALL SND,
     255: NEXT Z: RETURN
1250 FLASH : VTAB 22: HTAB 25: PRINT "TR
     IP";: NORMAL : FOR K = 1 TO 5: CALL
      SND,127: CALL SND,129: NEXT
1260 TEXT : HOME : VTAB 7
1270 PRINT "YOU WERE UNABLE TO MAINTAIN
     THE": PRINT "OPERATING FREQENCY OF
     THE REACTOR'S": PRINT "POWER OUTPUT
     . THE REACTOR'S MAIN"
1280 PRINT "BREAKER HAS BEEN TRIPPED. Y
     OU WILL BE SEVERELY REPRIMANDED.":
     SCRE = SCRE * .5: GOTO 1320
1290 GOSUB 1380: TEXT : HOME : VTAB 7
1300 PRINT "THE REACTOR HAS SHUT DOWN SA
     FELY.": PRINT : PRINT "YOU MAY REST
     ART THE REACTOR AFTER ALL": PRINT "
     COMPONENTS HAVE BEEN INSPECTED"
1310 PRINT " AND REPAIRED."
1320 VTAB 20: PRINT "YOUR SCORE IS :"; I
     NT (SCRE) * 100
1330 VTAB 23: HTAB 1: PRINT "WOULD YOU L
     IKE TO PLAY AGAIN ? (Y/N) =>";: GET
      A$
1340 IF A$ < > "Y" AND A$ < > "N" THEN
      1330
1350 IF A$ = "N" THEN HOME : TEXT : END
1360 HOME : VTAB 22: HTAB 13: PRINT "REP
     AIRING REACTOR"
1370 GOTO 360
1380 REM *** SHUT-DOWN ***
1390 FLASH : VTAB 23: HTAB 25: PRINT "SL
     AM";: NORMAL : FOR K = 1 TO 5: CALL
      SND,255: NEXT
1400 HCOLOR= 1: FOR RY = 1 TO 4: FOR RX
     = 1 TO 6:Y1 = (RX > 3):X1 = RX - 1
     - (3 * Y1)
1410 XX = (QP(RY,2) - 1) * 7 + (28 * X1)
      + 12:YY = (QP(RY,1) - 1) * 8 + (19
      * Y1) + 1: ROT= 0: DRAW 4 AT XX,YY
     : ROT= 32: DRAW 4 AT XX + 6,YY + 15
1420 CALL SND,248: NEXT RX,RY: ROT= 0: R
     ETURN
1430 REM *** UPDATE GAUGES ***
1440 BM = BT(Z): IF BM < = 1 THEN BM =
     1:BT(Z) = 1: GOTO 1470
1450 IF BM > 90 THEN BOUT = 1: RETURN
1460 IF BM = > 86 THEN CALL SND,254
1470 IF INST(Z) = 0 THEN XDRAW 2 AT (BP
     (Z,2) - 1) * 7 + (BT(Z) * 54 / 90),
     (BP(Z,1) - 1) * 8 + 4
1480 RETURN
1490 REM *** MELTDOWN ! ***
1500 HCOLOR= 5: FOR RY = 1 TO 4: FOR RX
     = 1 TO 6:Y1 = (RX > 3):X1 = RX - 1
     - (3 * Y1)
1510 XX = (QP(RY,2) - 1) * 7 + (28 * X1)
      + 12:YY = (QP(RY,1) - 1) * 8 + (19
      * Y1) + 1: ROT= 0: DRAW 4 AT XX,YY
     : ROT= 32: DRAW 4 AT XX + 6,YY + 15
1520 CALL SND,248: NEXT RX,RY: ROT= 0: R
     ETURN
1530 REM *** STEAM METER ***
1540 SP = SP + (TTS - SP) / 15:ST = INT
      (SP * 4 - 192): IF ST < 0 THEN ST
     = 1
1550 IF ST = > 127 THEN SOUT = 1: RETUR
     N
1560 XDRAW 1 AT 244,149 - ST: RETURN
1570 REM *** FREQ. METER ***
1580 HZ = (SP / 4 + 48) * .9375 * LOD:HM
      = (HZ - 58) * 28
1590 IF HZ < 58 OR HZ > 62 THEN HOUT = 1
1600 XDRAW 2 AT 24 + HM,148: RETURN
1610 REM *** SCAN KEYBOARD ***
1620 K = PEEK ( - 16384) - 128: POKE -
      16368,0: IF K < 0 THEN RETURN
1630 CALL SND,240: IF K = 17 THEN DNE =
     1: RETURN
1640 IF K = > 65 AND K < = 68 THEN HC
     OLOR= 6: GOSUB 1770:RY = K - 64: HC
     OLOR= 7: GOSUB 1770
1650 IF K < 49 OR K > 54 THEN 1690
1660 RX = K - 48: IF S(RY,RX) = 0 THEN
     RETURN
1670 HCOLOR= 0: GOSUB 1750:S(RY,RX) = S(
     RY,RX) - 1
1680 CV = 1: GOSUB 1740: RETURN
1690 FOR RX = 1 TO 6: IF K < > ASC ( M
     ID$ (SHFT$,RX,1)) THEN NEXT : RETU
     RN
1700 IF S(RY,RX) = 16 THEN RETURN
1710 S(RY,RX) = S(RY,RX) + 1: HCOLOR= 1:
      GOSUB 1750
1720 CV = - 1: GOSUB 1740: RETURN
1730 REM ** UPDATE TEMPERATURE **
1740 TS(RY) = TS(RY) + CV: RETURN
1750 REM ** MOVE ROD POSITIONS **
1760 Y1 = (RX > 3):X1 = RX - 1 - 3 * Y1:
      DRAW 3 AT (QP(RY,2) - 1) * 7 + (28
      * X1) + 13,(QP(RY,1) - 1) * 8 + (1
     9 * Y1) + 17 - S(RY,RX): RETURN
1770 REM *** QUADRANT BOX ***
1780 NB = 2:X1 = (QP(RY,2) - 1) * 7 - 2:
     Y1 = (QP(RY,1) - 1) * 8 - 2:X2 = X1
      + 100:Y2 = Y1 + 40: GOSUB 1790: RE
     TURN
1790 REM *** DRAW A BOX ***
1800 FOR Z = 0 TO NB - 1: HPLOT X1 + Z,Y
     1 + Z TO X2 - Z,Y1 + Z TO X2 - Z,Y2
      - Z TO X1 + Z,Y2 - Z TO X1 + Z,Y1
     + Z: NEXT : RETURN
1810 REM *** FLASH THE SCREEN ***
1820 INVERSE : FOR K = 0 TO 19 STEP 2: C
     ALL HCHAR,K,0,10,79,1: NEXT : NORMA
     L : RETURN
1830 REM *** DATA ***
1840 DATA NORMAL OPERATION,FAULTY SYSTEM
     S,EARTHQUAKE COUNTRY,ARMAGEDDON
1850 DATA 2,4,2,24,16,4,16,24
1860 DATA 5,4,5,18,10,4,10,18
1870 REM *** SHAPE TABLES ***
1880 DATA 4,0,10,0,18,0,27,0,31,0,45,45,
     45,62,63,63,63,0,54,54,54,46,36,36,
     36,36,0,45,45,5,0,54,54,54,54,54,54
     ,54,46,37,36,36,36,45,54,54,54,46,3
     7,36,36,36,36,36,36,36,4,0
1890 DATA "CHECK"
1900 REM *** MACHINE SUBR. ***
1910 DATA 32,76,231,134,37,32,76,231,134
     ,36,32,76,231,134,8,32,76,231,134,6
     ,32,76,231,138,41,1,170,189,106,8,1
     41,96,8,165,37,32,34,252,32,51
1920 DATA 8,169,160,32,240,253,198,6,16,
     244,96,169,7,133,7,160,0,36,50,48,2
     ,160,127,132,9,165,8,10,10,10,170,1
     64,36,24,165,40,133,38,165,41
1930 DATA 105,24,133,39,24,165,39,105,4,
     133,39,189,128,8,69,9,17,38,145,38,
     232,198,7,16,235,96,145,81,32,76,23
     1,134,8,169,0,168,136,208,253,44
1940 DATA 48,192,56,229,8,208,244,96,28,
     34,50,42,38,34,28,0,8,12,8,8,8,8,28
     ,0,28,34,32,24,4,2,62,0,62,32,16,24
     ,32,34,28,0
1950 DATA 16,24,20,18,62,16,16,0,62,2,30
     ,32,32,34,28,0,56,4,2,30,34,34,28,0
     ,62,32,16,8,4,4,4,0,28,34,34,28,34,
     34,28,0
1960 DATA 28,34,34,60,32,16,14,0,0,0,0,0
     ,0,0,0,0,17,17,1,1,0,0,0,0,0,0,0,0
1970 HOME : PRINT "THERE IS AN ERROR IN
     THE DATA STATEMENTS"; CHR$ (7): END
```

HCM

## OVER-REACTION

COMMODORE 64

```
100 REM ****************
110 REM * OVER-REACTION *
120 REM ****************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM WILLIAM K. BALTHROP
160 REM AND THE HCM STAFF
170 REM HOME COMPUTER MAGAZINE
180 REM VERSION 5.3.1
190 REM C-64 BASIC
200 REM DISPLAY TITLE SCREEN
210 POKE 56,128:CLR:POKE 53281,15.:POKE
     53280,15:T$="OVER-REACTION":X=RND(
    -TI)
220 POKE 650,0:PRINT"{SHIFT CLR}{CTRL B
    LK}{10CRSRDOWN}          **"T$" **"
230 PRINT"{12CRSRDOWN}          PRESS RETU
    RN TO CONTINUE":X=12:Y=10:Z=1
240 GET K$:GOSUB2450:Z=ABS(Z-1):POKE 19
    9,Z
250 FOR I=1 TO 50:NEXT:PRINTT$:IF K$=""
     THEN240
260 IF ASC(K$)<>13 THEN240
270 REM INITIALIZE PROGRAM AND INPUT LE
    VEL OF DIFFICULTY
280 DIM S(4,6),BP(4,2):FOR I=1 TO 4:REA
    D BP(I,1),BP(I,2):NEXT
290 FOR I=1 TO 4:READ ED$(I):NEXT:V1=54
    276:V2=V1+7:V3=V2+7:S1=129:S2=85:S3
    =65
300 FOR I=54272 TO 54296:READ D:POKE I,
    D:NEXT:L$="{CMDR G}{SHIFT G}{SHIFT
    H}{CMDR N}":U$="{CMDR P}{SHIFT F}{S
    HIFT D}{CMDR Y}"
310 IF PEEK(53272)=49 THEN400
320 PRINT"{SHIFT CLR}{10CRSRDOWN}
    PLEASE WAIT WHILE I DISPOSE"
330 PRINT"{CRSRDOWN}          OF SOME NU
    CLEAR WASTE"
```

*Continued*

```
340 POKE 56334,PEEK(56334)AND254:POKE 1,PEEK(1)AND251
350 FOR I=0 TO 2047:POKE 32768+I,PEEK(53248+I):NEXT
360 POKE 1,PEEK(1)OR4:POKE 56334,PEEK(56334)OR1
370 READ CH:IF CH=-1 THEN390
380 FOR I=0 TO 7:READ D:POKE 32768+CH*8+I,D:NEXT:GOTO370
390 POKE 53272,49:POKE 657,128:POKE 648,140:POKE 56576,PEEK(56576)AND253
400 POKE 198,0:PRINT"{SHIFT CLR}"TAB(13)T$:PRINT"{5CRSRDOWN}    ENTER SKILL LEVEL:{3CRSRDOWN}"
410 PRINT"{CRSRDOWN}          1) NORMAL OPERATION":PRINT"{CRSRDOWN}          2) FAULTY SYSTEMS"
420 PRINT"{CRSRDOWN}          3) EARTHQUAKE COUNTRY":PRINT"{CRSRDOWN}          4) ARMAGEDDON"
430 PRINT"{5CRSRDOWN}      CHOOSE ONE"
440 GET K$:IF K$<>"1" AND K$<>"2" AND K$<>"3" AND K$<>"4" THEN440
450 PRINT"{SHIFT CLR}":ON VAL(K$) GOTO470,520,560,630
460 REM START OF GAME MESSAGES
470 PRINT"YOU MAY ENCOUNTER MINOR PROBLEMS WITHIN    "
480 PRINT"THE REACTOR; HOWEVER, A COOL HEAD SHOULD  "
490 PRINT"BE ABLE TO KEEP THINGS UNDER CONTROL.    "
500 PRINT"WATCH FOR OVERHEATING IN THE REACTOR    "
510 PRINT"CORES, AND CHANGES IN THE LOAD.":SK=750:PR=1:GOTO680
520 PRINT"IT'S BEEN ONE OF THOSE DAYS!  ELECTRICAL"
530 PRINT"PROBLEMS IN THE COOLING SYSTEM AND    "
540 PRINT"CONTROL PANEL HAVE YOU IN A FRENZY.   "
550 PRINT"WATCH REACTOR TEMPERATURES!":SK=700:PR=2:GOTO680
560 PRINT"YOUR REACTOR IS IN THE HEART OF      "
570 PRINT"CALIFORNIA EARTHQUAKE COUNTRY!!   BE"
580 PRINT"PREPARED FOR DRASTIC AND SUDDEN CHANGES  "
590 PRINT"IN THE REACTOR.   IRREVERSABLE  DAMAGE MAY"
600 PRINT"RESULT.   WATCH THE STEAM PRESSURE!    "
610 PRINT"INTERNAL LEAKS MAY DEVELOP IF  AN     {SPACE}"
620 PRINT"EARTHQUAKE HITS.":SK=600:PR=3:GOTO680
630 PRINT"OUR WORST FEARS ARE NOW A REALITY.    "
640 PRINT"NUCLEAR WAR IS ON THE VERGE OF ERUPTING."
650 PRINT"THE LOAD ON YOUR REACTOR IS TERRIFIC AS  "
660 PRINT"THE WORLD PREPARES FOR WAR. RIOTS ARE   "
670 PRINT"INCREASING!!":SK=450:PR=4
680 X=8:Y=24:GOSUB2450:PRINT"PRESS RETURN TO CONTINUE";
690 GET K$:IF K$="" THEN690
700 IF ASC(K$)<>13 THEN690
710 REM DRAW CONTROL PANEL
720 PRINT"{SHIFT CLR}{CTRL RVSOFF} {CTRL BLK}{CMDR M}{8CMDR T}{CMDR G} {CMDR N}{5CMDR T}";
730 PRINT"{3CMDR T}{CMDR G}      {CMDR O}{SHIFT R}{CMDR O}     ";
740 PRINT" {CMDR M}      {CMDR G} {CMDR M}   ";
750 PRINT"   {CMDR G}      {CMDR K} {CMDR I}  ";
760 PRINT" {CTRL BLK}{CMDR M}{8CMDR @}{CMDR G}  {CMDR M}{5CMDR @}";
770 PRINT"{3CMDR @}{CMDR G}   S {CMDR K} {CMDR I}-1000  ";
780 PRINT"{CTRL BLU}{CMDR A}{3SHIFT C}{CMDR R}{3SHIFT C}{CMDR R}{3SHIFT C}{CTRL BLK}{SHIFT -}{CTRL BLU}{3SHIFT C}{CMDR R}{3SHIFT C}";
790 PRINT"{CMDR R}{3SHIFT C}{CMDR S}{CTRL BLK}T{SPACE}{CMDR K} {CMDR I}";
800 PRINT"{CTRL BLU}{SHIFT -}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT -}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT -}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLK}{SHIFT -}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT -}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF}";
810 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}   {CTRL BLK}E{SPACE}{CMDR K} {CMDR I}";
820 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}A {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}B{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}C {SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}D {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}E";
830 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}F {CTRL BLU}{SHIFT B}  {CTRL BLK}A{SPACE}{CMDR K} {CMDR I}←900";
840 PRINT"{CTRL BLU}{CMDR Q}{3SHIFT C}{SHIFT +}{3SHIFT C}{SHIFT +}{3SHIFT C}{CTRL BLK}{SHIFT B}{CTRL BLU}{3SHIFT C}{SHIFT +}{3SHIFT C}";
850 PRINT"{SHIFT +}{3SHIFT C}{CMDR W} {CTRL BLK}M {CMDR K} {CMDR I}";
860 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLK}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF}";
870 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}   {CTRL BLK} {CMDR K} {CMDR I}";
880 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}G {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}H{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}I {SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}J {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}K";
890 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}L {CTRL BLU}{SHIFT B}  {CTRL BLK}P {CMDR K} {CMDR I}←800";
900 PRINT"{12SHIFT C}{SHIFT +}{7SHIFT C}";
910 PRINT"{5SHIFT C}   R {CMDR K} {CMDR I}";
920 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLK}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF}";
930 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}   {CTRL BLK}E {CMDR K} {CMDR I}";
940 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}M {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}N{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}O {SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}P {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}Q";
950 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}R {CTRL BLU}{SHIFT B}  {CTRL BLK}S {CMDR K} {CMDR I}←700";
960 PRINT"{CTRL BLU}{CMDR Q}{3SHIFT C}{SHIFT +}{3SHIFT C}{SHIFT +}{3SHIFT C}{CTRL BLK}{SHIFT B}{CTRL BLU}{3SHIFT C}{SHIFT +}{3SHIFT C}";
970 PRINT"{SHIFT +}{3SHIFT C}{CMDR W} {CTRL BLK}S {CMDR K} {CMDR I}";
980 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLK}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF}";
990 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}0{CTRL RVSOFF} {CTRL BLU}{SHIFT B}   {CTRL BLK}U {CMDR K} {CMDR I}";
1000 PRINT"{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}S {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}T{CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}U {SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}V {CTRL BLU}{SHIFT B}{CTRL RVSON}{CTRL GRN}8{CTRL RVSOFF}{CTRL BLK}W";
```

Continued

# OVER-REACTION *Continued*

```
1010 PRINT"☛CTRL BLU☚☛SHIFT B☚☛CTRL RVSON☚☛CTRL GRN☚8☛CTRL RVSOFF☚☛CTRL BLK☚X ☛CTRL BLU☚☛SHIFT B☚ ☛CTRL BLK☚R ☛CMDR K☚ ☛CMDR I☚←  600  ";
1020 PRINT"☛CTRL BLU☚☛CMDR Z☚☛3SHIFT C☚☛CMDR E☚☛3SHIFT C☚☛CMDR E☚☛3SHIFT C☚☛CTRL BLK☚☛SHIFT -☚☛CTRL BLU☚☛3SHIFT C☚☛CMDR E☚☛3SHIFT C☚";
1030 PRINT"☛CMDR E☚☛3SHIFT C☚☛CMDR X☚☛CTRL BLK☚E ☛CMDR K☚ ☛CMDR . I☚ ";
1040 PRINT"  ☛CMDR M☚☛8CMDR T☚☛CMDR G☚☛CMDR M☚☛5CMDR T☚";
1050 PRINT"☛3CMDR T☚☛CMDR G☚     ☛CMDR K☚ ☛CMDR I☚ ";
1060 PRINT"  ☛CMDR M☚       ☛CMDR G☚  ☛CMDR M☚";
1070 PRINT"   ☛CMDR G☚      ☛3CMDR Y☚  ";
1080 PRINT"  ☛CMDR M☚☛8CMDR @☚☛CMDR G☚☛CMDR M☚☛5CMDR @☚";
1090 PRINT"☛3CMDR @☚☛CMDR G☚SLAM ☛SHIFT W☚ ☛SHIFT B☚ TRIP ☛SHIFT W☚ ";
1100 PRINT"                ";
1110 PRINT"    ☛15CMDR Y☚ ";
1120 PRINT"      FREQUENCY       ";
1130 PRINT"   ERROR DETECTION ☛SHIFT W☚   ";
1140 PRINT"☛CMDR M☚☛18CMDR T☚☛CMDR G☚";
1150 PRINT"☛CMDR A☚☛18SHIFT C☚☛CMDR S☚";
1160 PRINT"☛CMDR M☚                  ☛CMDR G☚";
1170 PRINT"☛SHIFT -☚                  ☛SHIFT -☚";
1180 PRINT"☛CMDR M☚☛18CMDR @☚☛CMDR G☚";
1190 PRINT"☛CMDR Z☚☛18SHIFT C☚☛CMDR X☚";
1200 PRINT"58        60        62";
1210 PRINT"                ☛HOME☚";
1220 REM INITIALIZE GAME VARIABLES
1230 FOR RY=1 TO 4:FOR RX=1 TO 6:S(RY,RX)=8:NEXT:CS(RY)=0:IN(RY)=0:TS(RY)=48
1240 BT(RY)=48:Z=RY:GOSUB1820:NEXT:SO=0:HO=0:BO=0:DO=0:LO=1:SP=64:TT=64:HZ=60
1250 SD=0:CT=.5:LD=1:LN=(RND(1)*.067)+.967:LP=.05:GOSUB1920:GOSUB1970
1260 REM MAIN CONTROL LOOP
1270 IF ED THEN GOSUB2050
1280 GOSUB1540:POKE V3,S3-1:SC=SC+(2-(ABS(60-HZ))):IF DO THENGOSUB1750:GOTO2090
1290 FOR Z=1 TO 4:F=20:IF CS(Z) THEN F=FL
1300 BT(Z)=BT(Z)+((TS(Z)+SIN((CT+Z*1.5))*F)-BT(Z))/3:GOSUB1820:NEXT
1310 TT=((BT(1)+BT(2)+BT(3)+BT(4))/3)-SD:IF TT<>SP THEN GOSUB1920
1320 GOSUB1970:IF LP<1.05 THEN1350
1330 LD=LN:LP=.05:IF PR=4 THEN LN=(RND(1)*.045)+.99:GOTO1350
1340 LN=(RND(1)*.067)+.967
1350 LO=(LD*(1-LP))+(LN*LP):LP=.05+LP:IF PR=1 THEN1380
1360 IF INT(RND(1)*15) THEN1380
1370 GOSUB1400
1380 CT=CT+.05:IF SO OR BO OR HO THEN2090
1390 GOTO1270
1400 IF PR=2 THEN GOTO1440
1410 P=INT(RND(1)*8)+1
1420 IF PR=4 THEN ON P GOTO1470,1470,1470,1470,1480,1480,1480,1520
1430 ON P GOTO1470,1470,1470,1480,1480,1480,1520,1520
1440 REM COOLING SYSTEM MALFUNCTION
1450 IF INT(RND(1)*5)>2 THEN1480
1460 C=INT(RND(1)*4)+1:IF CS(C)=1 THEN RETURN
1470 CS(C)=1:FL=INT(RND(1)*20)+20:ED=2:GOSUB2010:FOR I=1 TO 700:NEXT:RETURN
1480 REM GAUGE OUT
1490 I=INT(RND(1)*4)+1:IF IN(I)=1 THEN RETURN
1500 IN(I)=1:ED=3:GOSUB2010:X=BP(I,2):Y=BP(I,1):GOSUB2450:PRINT"☛CTRL RVSON☚☛CTRL RED☚☛8CMDR +☚☛CTRL BLK☚"
1510 FOR I=1 TO 700:NEXT:RETURN
1520 REM STEAM LEAK
1530 SD=SD+RND(1)*6+.2:ED=4:GOSUB2010:FOR I=1 TO 700:NEXT:RETURN
1540 REM KEY INPUT
1550 IF PEEK(198)>1 THEN POKE 198,1
1560 GET K$:IF K$="" THEN RETURN
1570 K=ASC(K$):IF K=133 THEN DO=1:RETURN
1580 K=K-64:CV=1:IF K>24 THEN K=K-128:CV=-1
1590 IF K<1 OR K>24 THEN RETURN
1600 POKE V3,S3:RY=INT(K/6.001)+1:RX=K-(RY-1)*6
1610 IF S(RY,RX)-CV>16 OR S(RY,RX)-CV<0 THEN RETURN
1620 S(RY,RX)=S(RY,RX)-CV
1630 REM POSITION THE RODS
1640 X=(RX-1)*4+1:Y=(RY-1)*3+4:GOSUB2450:IF S(RY,RX)>8 THEN1660
1650 PRINT"☛CTRL GRN☚☛CTRL RVSON☚0";:Y=Y+1:GOSUB2450:PRINT RIGHT$(STR$(S(RY,RX)),1)"☛CTRL BLK☚":GOTO1670
1660 PRINT"☛CTRL GRN☚☛CTRL RVSON☚"RIGHT$(STR$(S(RY,RX)-8),1);:Y=Y+1:GOSUB2450:PRINT"8☛CTRL BLK☚"
1670 REM UPDATE BLOCK TEMPERATURE
1680 IF RY>2 THEN1720
1690 IF RX>3 THEN1710
1700 TS(1)=TS(1)+CV:RETURN
1710 TS(2)=TS(2)+CV:RETURN
1720 IF RX>3 THEN1740
1730 TS(3)=TS(3)+CV:RETURN
1740 TS(4)=TS(4)+CV:RETURN
1750 REM TRIP AND SLAM ROUTINE
1760 GOSUB2050:POKEV2-4,223:POKE V2,S2:X=33:E$="TRIP":IF DO THEN E$="SLAM":X=24
1770 Y=18:GOSUB2450:PRINT"☛CTRL RVSON☚"E$"☛CTRL RVSOFF☚☛CTRL RED☚☛SHIFT Q☚☛CTRL BLK☚":IFDO=0 THEN FOR I=1 TO 1000:NEXT:GOTO1810
1780 FOR I=1 TO 4:GOSUB2450:Z=1-((I>2)*2)
1790 FOR RY=Z TO Z+1:Z=4-((IAND1)*3):FOR RX=Z TO Z+2:S(RY,RX)=16:GOSUB1630
1800 FOR Z=1 TO 200:NEXT:NEXT RX,RY:NEXT:Y=18:X=24
1810 POKEV2,S2-1:GOSUB2450:PRINT"☛CTRL BLK☚"E$" ☛SHIFT W☚":POKE V2-4,71:FOR I=1TO900:NEXT:RETURN
1820 REM UPDATE TEMPERATURE GAUGES
1830 BM=BT(Z):IF BM<2 THEN BM=1:BT(Z)=1:GOTO1860
1840 IF BM>90 THEN BO=1:RETURN
1850 IF BM>85 THEN ED=1:GOSUB2010
1860 IF IN(Z) THEN RETURN
1870 Y=BP(Z,1)
1880 IF BM>11 THEN X=BP(Z,2)+INT(BM/12)-1:GOSUB2450:IF X=>BP(Z,2) THENPRINT" "
1890 IF BM<84 THEN X=BP(Z,2)+INT((BM-1)/12)+1:GOSUB2450:PRINT" "
1900 X=BP(Z,2)+INT(BM/12):GOSUB2450:PRINTMID$(L$,INT((BM-INT(BM/12)*12)/3)+1,1)
1910 RETURN
1920 REM UPDATE STEAM PRESSURE METER
1930 SP=SP+(TT-SP)/15:SS=INT(SP*4-192):IF SS<1 THEN SS=1:GOTO1950
1940 IF SS>127 THEN SO=1:RETURN
1950 X=31:Y=1:GOSUB2450:PRINT" ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚"
1960 Y=16-INT(SS/8):GOSUB2450:PRINT MID$(U$,INT((SS-INT(SS/8)*8)/2)+1,1):RETURN
1970 REM UPDATE FREQUENCY METER
1980 HZ=(SP/4+48)*.9375*LO:HM=(HZ-58)*36:IF (HM<0) OR (HM>144) THEN HO=1:RETURN
1990 X=2:Y=22:GOSUB2450:PRINT"                    ":X=1+INT(HM/8):GOSUB2450
2000 PRINT MID$(L$,INT((HM-INT(HM/8)*8)/2)+1,1):RETURN
2010 REM ERROR DETECTION
2020 X=38:Y=20:GOSUB2450:PRINT"☛CTRL RED☚☛SHIFT Q☚☛CTRL BLK☚"
2030 X=21:Y=22:GOSUB2450:PRINT ED$(ED):IF ED=4 THEN POKE V1,S1:RETURN
2040 POKE V2,S2:RETURN
2050 REM CLEAR ERROR
2060 X=38:Y=20:GOSUB2450:PRINT"☛SHIFT W☚":X=21:Y=22:GOSUB2450
2070 PRINT"                  ":IF ED=4 THEN POKE V1,S1-1:ED=0:RETURN
2080 POKE V2,S2-1:ED=0:RETURN
2090 REM END OF GAME MESSAGES
2100 IF BO THEN2140
2110 IF SO THEN2210
2120 IF HO THEN2290
2130 IF DO THEN2360
2140 POKE 657,0:POKE V1,S1:POKE V2,S2:POKE V3,S3:FOR I=1 TO 101:POKE V2-4,I
2150 E$=CHR$(14+(IAND1)*128):PRINT"☛HOME☚"E$:NEXT:POKE 657,128:POKE V2-4,71
2160 POKE V3,S3-1:POKE V2,S2-1:POKE V1,S1-1
2170 PRINT"☛SHIFT CLR☚MELTDOWN!!!!☛CRSRDOWN☚":PRINT"THE REACTOR CORE TEMPERATURE HAS ☛SPACE☚"
2180 PRINT"EXCEEDED SAFE LEGAL LIMITS. MELTDOWN "
2190 PRINT"HAS BEGUN AND THE SURROUNDING AREA MUST "
2200 PRINT"BE EVACUATED":SC=SC*.2:GOTO2400
2210 POKE 657,0:POKE V1,S1:POKE V2,S2:POKE V3,S3:FOR I=1 TO 101:POKE V2-4,I
```

Continued

## ■ OVER-REACTION *Continued*

COMMODORE 64

```
2220 E$=CHR$(14+(IAND1)*128):PRINT"■HOME
     ■"E$:NEXT:POKE 657,128:POKE V2-4,71
2230 POKE V3,S3-1:POKE V2,S2-1:POKE V1,S
     1-1
2240 PRINT"■SHIFT CLR■THE REACTOR STEAM
     PRESSURE HAS CAUSED A "
2250 PRINT"MAJOR LEAK OUTSIDE THE SYSTEM
      AND A "
2260 PRINT"MELTDOWN HAS BEGUN.  THE SURR
     OUNDING "
2270 PRINT"AREA IS BECOMING CONTAMINATED
      AND MUST "
2280 PRINT"BE EVACUATED.":SC=SC*.2:GOTO2
     400
2290 GOSUB1750:PRINT"■SHIFT CLR■YOU WERE
      UNABLE TO MAINTAIN THE "
2300 PRINT"OPERATING FREQUENCY OF THE RE
     ACTOR'S "
2310 PRINT"POWER OUTPUT.  THE AREA'S POW
     ER GRID IS "
2320 PRINT"NOT CAPABLE OF HANDLING THIS
     BAD POWER. "
2330 PRINT"THE REACTOR'S MAIN BREAKER HA
     S BEEN "
2340 PRINT"TRIPPED.  YOU WILL BE SEVEREL
     Y "
2350 PRINT"REPRIMANDED.":SC=SC*.5:GOTO24
     00
2360 PRINT"■SHIFT CLR■THE REACTOR HAS SH
     UT DOWN SAFELY.  YOU "
2370 PRINT"MAY RESTART THE REACTOR AFTER
      ALL "
2380 PRINT"COMPONENTS HAVE BEEN INSPECTE
     D AND "
2390 PRINT"REPAIRED."
2400 X=2:Y=22:GOSUB2450:PRINT"YOUR SCORE
      IS"INT(SC)*100:Y=24:GOSUB2450
2410 POKE 198,0:PRINT"WOULD YOU LIKE TO
     PLAY AGAIN (Y/N)";
2420 GET K$:IF K$<>"Y" AND K$<>"N" THEN2
     420
2430 IF K$="Y" THEN400
2440 PRINT"■SHIFT CLR■BYE...":END
2450 REM PLACE CURSOR AT X,Y
2460 POKE 781,Y:POKE 782,X:POKE 783,0:SY
     S 65520:RETURN
2470 REM TEMPERATURE GAUGE POSITIONS
2480 DATA001,002,001,015,017,002,017,015
2490 REM DATA FOR ERROR DETECTION
2500 DATA "DANGEROUS TEMP","FLUCTUATIONS
         ","GAUGE BROKEN    ","STEAM LEAK
     
2510 REM SOUND DATA
2520 DATA030,025,000,112,128,000,171,071
     ,006
2530 DATA128,117,084,000,160,143,012,096
     ,123
2540 DATA064,000,240,000,090,177,047
2550 REM CUSTOM CHARACTER DATA
2560 DATA097,128,240,128,255,128,240,128
     ,255
2570 DATA098,001,015,001,255,001,015,001
     ,255
2580 DATA099,255,051,051,051,003,003,000
     ,000
2590 DATA100,000,000,003,003,051,051,051
     ,255
2600 DATA176,129,129,129,129,129,129,129
     ,129
2610 DATA177,129,129,129,129,129,129,129
     ,255
2620 DATA178,129,129,129,129,129,129,255
     ,255
2630 DATA179,129,129,129,129,129,255,255
     ,255
2640 DATA180,129,129,129,129,255,255,255
     ,255
2650 DATA181,129,129,129,255,255,255,255
     ,255
2660 DATA182,129,129,255,255,255,255,255
     ,255
2670 DATA183,129,255,255,255,255,255,255
     ,255
2680 DATA184,255,255,255,255,255,255,255
     ,255,-001
```

HCM

## ■ OVER-REACTION

IBM PC & IBM PCjr

```
100 '  ******************
110 '  * OVER-REACTION  *
120 '  ******************
130 '  COPYRIGHT 1985
140 '  EMERALD VALLEY PUBLISHING CO.
150 '  BY WILLIAM K. BALTHROP
160 '  HOME COMPUTER MAGAZINE
170 '  VERSION 5.3.1
180 '  IBM PCjr WITH CARTRIDGE BASIC &
    DOS 2.1 or
190 '  IBM PC WITH BASICA and
200 '  COLOR/GRAPHICS ADAPTER and
210 '  COLOR MONITOR
220 '
230 '  INITIALIZE
240 '
250 RTN$=CHR$(17)+CHR$(217):RANDOMIZE T
    IMER:PLAY "MF"
260 CLS:KEY OFF:SCREEN 1:COLOR 1,0:LOCA
    TE 12,13:PRINT "OVER-REACTION":LOCA
    TE 22,11:PRINT "PRESS ";RTN$;" TO S
    TART":GOSUB 1350:GOSUB 1050
270 DEFINT N:DEFDBL C
280 OPTION BASE 1:DIM N(44),S(4,6),TS(4
    ),BT(4),INST(4),CSYS(4),BP(4,2),RP(
    24,2)
290 RESTORE 1390:FOR Z=1 TO 24:READ RP(
    Z,1),RP(Z,2):NEXT:FOR Z=1 TO 4:READ
     BP(Z,1),BP(Z,2):NEXT
300 CLS:DRAW "BM10,10C2R6D2GD2GDL2UHU2H
    U2BF3P2,2U3C1D9":GET (6,10)-(20,19)
    ,N
310 FOR X=1 TO 4:FOR Y=1 TO 6:S(X,Y)=8:
    NEXT:BT(X)=48:TS(X)=48:INST(X)=0:CS
    YS(X)=0:NEXT:RESTORE 320:READ STOUT
    ,BTOUT,HZOUT,DONE,LODE,STP,TTS,HZ,S
    TDAM,CT,LDP,LP,HZO,SCORE
320 DATA 0,0,0,0,1,48,48,60,0,.05,1,.05
    ,3.14159,0
330 LDN=(RND*.045)+.999
340 GOSUB 1190
350 '
360 '  MAIN CONTROL LOOP
370 '
380 GOSUB 580:IF DONE=1 THEN 970 ELSE S
    CORE=SCORE+(2-(ABS(60-HZ))):GOSUB 6
    90:GOSUB 760:CTT=CTT+1:CT=CTT/20:IF
     CT<>INT(CT) THEN 400 ELSE LDP=LDN:
    LP=.05:IF PROBS<4 THEN LDN=(RND*.06
    7)+.967 ELSE LDN=(RND*.045)+.99
390 IF PROBS>1 THEN IF INT(RND*5)=0 THE
    N GOSUB 470
400 LODE=(LDP*(1-LP))+(LDN*LP):LP=LP+.0
    5
410 IF BT(1)>70 OR BT(2)>70 OR BT(3)>70
     OR BT(4)>70 THEN LINE (204,172)-(3
    07,187),0,BF
420 IF LP>1 THEN LP=1
430 IF STOUT=1 OR BTOUT=1 OR HZOUT=1 TH
    EN 890 ELSE 380
440 '
450 '  GENERATE PROBLEMS
460 '
470 ON PROBS-1 GOTO 480,530,530
480 IF INT(RND*5)<3 THEN 510
490 I=INT(RND*4)+1:IF INST(I)=1 THEN RE
    TURN
500 LOCATE 23,27:PRINT "INST. FAIL";:IN
    ST(I)=1:FOR FREQ=1000 TO 110 STEP -
    25:SOUND FREQ,1:NEXT:LINE (204,172)
    -(307,187),0,BF:RETURN
510 I=INT(RND*4)+1:IF CSYS(I)=1 THEN RE
    TURN
520 LOCATE 23,27:PRINT "COOLING SYS";:C
    SYS(I)=1:FLUX=INT(RND*20)+20:FOR RE
    P=1 TO 3:FOR FREQ=300 TO 2000 STEP
    176:SOUND FREQ,1:NEXT:NEXT:LINE (20
    4,172)-(307,187),0,BF:RETURN
530 P=INT(RND*8)+1:IF PROBS=3 THEN ON P
     GOTO 490,490,490,510,510,540,540,5
    40 ELSE ON P GOTO 490,490,490,490,5
    10,510,510,540
540 LOCATE 23,27:PRINT "STEAM LEAK";:ST
    DAM=STDAM+RND*6+.2:FOR REP=1 TO 50:
    FOR TM=1 TO REP*5:NEXT:SOUND 1000-R
    EP*15,1:NEXT:LINE (204,172)-(307,18
    7),0,BF:RETURN
550 '
560 '  GET INPUT AND ADJUST RODS
570 '
580 A$=INKEY$:IF A$="" THEN RETURN ELSE
     DEF SEG=0;POKE 1050,PEEK(1052):IF
    ((A$<"A" OR A$>"x") OR (A$>"X" AND
    A$<"a")) AND A$<>CHR$(27) AND A$<>"
    0" THEN RETURN ELSE SOUND 880,1:IF
    A$=CHR$(27) THEN DONE=1:RETURN
590 IF A$="0" THEN LOCATE 23,27:PRINT "
    ** PAUSE **";:GOSUB 1350:LINE (204,
    172)-(307,187),0,BF:RETURN
600 K=ASC(A$):IF K>95 THEN K=K-96:FLG=-
    1 ELSE K=K-64:FLG=1
610 X=INT(K/6.001)+1:Y=K-(X-1)*6:IF S(X
    ,Y)+FLG<0 OR S(X,Y)+FLG>16 THEN RET
    URN ELSE S(X,Y)=S(X,Y)+FLG
620 LINE(RP(K,1),RP(K,2)+(16-S(X,Y))-1)
    -(RP(K,1)+5,RP(K,2)+(16-S(X,Y))-1),
    0
630 IF S(X,Y)>0 THEN LINE(RP(K,1),RP(K,
    2)+(16-S(X,Y)))-(RP(K,1)+5,RP(K,2)+
    (16-S(X,Y))),2
640 IF X<3 THEN IF Y<4 THEN TS(1)=TS(1)
    -FLG ELSE TS(2)=TS(2)-FLG ELSE IF Y
    <4 THEN TS(3)=TS(3)-FLG ELSE TS(4)=
    TS(4)-FLG
650 RETURN
660 '
670 '  CONTROL TEMPERATURE METERS
680 '
690 FOR Z=1 TO 4:IF CSYS(Z)=0 THEN F=20
     ELSE F=FLUX
700 BT(Z)=BT(Z)+((TS(Z)+SIN(CT+Z*1.5)*F
    )-BT(Z))/3:IF INST(Z)=1 THEN LINE (
    BP(Z,1),BP(Z,2)-1)-(BP(Z,1)+56,BP(Z
    ,2)+16),2,BF ELSE PUT(BP(Z,1)+BT(Z)
    *.58,BP(Z,2)-1),N,PSET
```

Continued

## OVER-REACTION *Continued*

```
710  IF BT(Z)>78 THEN BTOUT=1:RETURN ELS
     E IF BT(Z)>68 THEN SOUND 330,1:LOCA
     TE 23,27:PRINT "HIGH TEMP";:FOR TD=
     1 TO 400:NEXT:LOCATE 23,27:PRINT SP
     ACE$(9);
720  NEXT:RETURN
730  '
740  '  CONTROL STEAM PRESSURE GAUGE
750  '
760  TTS=(BT(1)+BT(2)+BT(3)+BT(4))*.25-S
     TDAM:IF TTS<>STP THEN GOSUB 780
770  GOSUB 830:RETURN
780  STP=STP+(TTS-STP)/15:ST=INT(STP*4-1
     40):IF ST<1 THEN ST=1 ELSE IF ST>90
      THEN STOUT=1:RETURN
790  PUT (205+ST,10),N,PSET:RETURN
800  '
810  '  CONTROL FREQUENCY METER
820  '
830  HZ=(STP*.25+48)*LODE:HZM=3.14159+(6
     0-HZ)*.785:IF HZ<58 OR HZ>62 THEN H
     ZOUT=1
840  IF INT(SIN(HZO)*35)*INT(COS(HZO)*35
     )=INT(SIN(HZM)*35)*INT(COS(HZM)*35)
      THEN RETURN
850  LINE (255,114)-(SIN(HZO)*35+255,COS
     (HZO)*35+114),0:LINE (255,114)-(SIN
     (HZM)*35+255,COS(HZM)*35+114),2:HZO
     =HZM:RETURN
860  '
870  '  END OF GAME CONDITIONS
880  '
890  LOCATE 3,1:IF BTOUT=1 THEN 900 ELSE
      IF STOUT=1 THEN 920 ELSE IF HZOUT=
     1 THEN 940
900  LOCATE 23,27:PRINT "MELTDOWN";:K=0:
     LINE (256,144)-(300,151),2,BF:FOR Z
     =1 TO 24:LINE (RP(Z,1),RP(Z,2))-(RP
     (Z,1)+5,RP(Z,2)+15),0,BF:SOUND 3000
     /Z,3:FOR TD=1 TO 200:NEXT:NEXT:CLS:
     LOCATE 3,1
910  PRINT "!!! MELTDOWN !!!":PRINT:PRIN
     T"THE REACTOR CORE TEMPERATURE HAS
            EXCEEDED SAFE LIMITS. THE ME
     LTDOWN HAS  BEGUN. THE SURROUNDING
     AREA MUST BE     EVACUATED!":SCORE=
     SCORE*.2:GOTO 990
920  LOCATE 23,27:PRINT "HIGH PRESS.";:F
     OR Z=1 TO 7:LINE (256,144)-(300,151
     ),2,BF:SOUND 110,5:FOR TD=1 TO 200:
     NEXT:LINE (256,144)-(300,151),1,BF:
     SOUND 440,2:FOR TD=1 TO 200:NEXT:NE
     XT:CLS:LOCATE 3,1
930  PRINT "THE REACTOR STEAM PRESSURE H
     AS GONE     BEYOND SAFE LIMITS, CAU
     SING A MAJOR LEAKOUTSIDE THE SYSTEM
     . A MELTDOWN OF THE  CORE IS IMMIN
     ENT. THE SURROUNDING       AREA IS
     BECOMING CONTAMINATED, AND MUST BE
     EVACUATED.":SCORE=SCORE*.2:GOTO 990
940  LOCATE 23,27:PRINT "BAD FREQ";:FOR
     Z=1 TO 7:LINE (256,144)-(300,151),2
     ,BF:SOUND 220,5:FOR TD=1 TO 200:NEX
     T:LINE (256,144)-(300,151),1,BF:SOU
     ND 880,1:FOR TD=1 TO 200:NEXT:NEXT:
     CLS:LOCATE 3,1
950  PRINT "YOU WERE UNABLE TO MAINTAIN
     THE         OPERATING FREQUENCY OF
     THE REACTOR'S    POWER OUTPUT. THE
     AREA'S POWER GRID IS NOT CAPABLE O
     F HANDLING THIS BAD POWER. THE REAC
     TOR'S MAIN BREAKER HAS BEEN"
960  PRINT "TRIPPED. YOU WILL BE SEVEREL
     Y":PRINT "REPRIMANDED.":SCORE=SCORE
     *.5:GOTO 990
970  LOCATE 23,27:PRINT "START SLAM";:FO
     R Z=1 TO 24:LINE (256,160)-(300,167
     ),1,BF:LINE (RP(Z,1),RP(Z,2))-(RP(Z
     ,1)+5,RP(Z,2)+15),0,BF:SOUND 2600/Z
     ,2:LINE (256,160)-(300,167),2,BF:FO
     R TD=1 TO 50:NEXT:NEXT:CLS:LOCATE 3
     ,1
980  PRINT "THE REACTOR HAS SHUT DOWN SA
     FELY. YOU   MAY RESTART THE REACTOR
      AFTER ALL       COMPONENTS HAVE BE
     EN INSPECTED AND      REPAIRED."
990  LOCATE 17,1:PRINT "YOUR SCORE IS:";
     INT(SCORE*20):PRINT:PRINT "WOULD YO
     U LIKE TO PLAY AGAIN (Y/N)?"
1000 GOSUB 1350:IF A$<>"Y" AND A$<>"y" A
     ND A$<>"N" AND A$<>"n" THEN 1000 EL
     SE IF A$="N" OR A$="n" THEN CLS:PRI
     NT "BYE BYE...":END
1010 CLS:RUN
1020 '
1030 '  GET LEVEL OF DIFFICULTY
1040 '
1050 CLS:LOCATE 1,13:PRINT "OVER-REACTIO
     N":LOCATE 5,1:PRINT "1) NORMAL OPER
     ATION":PRINT:PRINT "2) FAULTY SYSTE
     MS":PRINT:PRINT "3) EARTHQUAKE COUN
     TRY":PRINT:PRINT "4) ARMAGEDDON"
1060 GOSUB 1350:IF A$<"1" OR A$>"4" THEN
      1060 ELSE CLS:ON VAL(A$) GOTO 1070
     ,1090,1110,1140
1070 LOCATE 3,1:PRINT "YOU MAY ENCOUNTER
      MINOR PROBLEMS WITHIN THE REACTOR;
      HOWEVER, A COOL HEAD SHOULDBE ABLE
      TO KEEP THINGS UNDER CONTROL.":PRI
     NT:PRINT "WATCH FOR OVERHEATING IN
     THE REACTOR    CORES, AND CHANGES I
     N THE LOAD."
1080 PROBS=1:LOCATE 22,1:PRINT "PRESS ";
     RTN$;" TO CONTINUE":GOSUB 1350:RETU
     RN
1090 LOCATE 3,1:PRINT "IT'S BEEN ONE OF
     THOSE DAYS! ELECTRICAL PROBLEMS IN
     THE COOLING SYSTEM AND     CONTROL
      PANEL HAVE YOU IN A FRENZY.":PRINT
     :PRINT "WATCH THE REACTOR TEMPERATU
     RES!"
1100 PROBS=2:LOCATE 22,1:PRINT "PRESS ";
     RTN$;" TO CONTINUE":GOSUB 1350:RETU
     RN
1110 LOCATE 3,1:PRINT "YOUR REACTOR IS I
     N THE HEART OF":PRINT "CALIFORNIA E
     ARTHQUAKE COUNTRY!!!!":PRINT:PRINT
     "BE PREPARED FOR DRASTIC AND SUDDEN
           CHANGES IN THE REACTOR. IRREV
     ERSABLE     DAMAGE MAY RESULT.":PRIN
     T
1120 PRINT "WATCH THE STEAM PRESSURE!":P
     RINT "INTERNAL LEAKS MAY DEVELOP IF
      AN          EARTHQUAKE HITS."
1130 PROBS=3:LOCATE 22,1:PRINT "PRESS ";
     RTN$;" TO CONTINUE":GOSUB 1350:RETU
     RN
1140 LOCATE 3,1:PRINT "OUR WORST FEARS A
     RE NOW A REALITY.      NUCLEAR WAR
     IS ON THE VERGE OF ERUPTING.THE LOA
     D ON YOUR REACTOR IS TERRIFIC AS TH
     E WORLD PREPARES FOR WAR.":PRINT:PR
     INT "RIOTS ARE INCREASING!!"
1150 PROBS=4:LOCATE 22,1:PRINT "PRESS ";
     RTN$;" TO CONTINUE":GOSUB 1350:RETU
     RN
1160 '
1170 '  DRAW REACTOR CONTROL PANEL
1180 '
1190 CLS:LOCATE 1,26:PRINT "STEAM PRESSU
     RE":LOCATE 7,3:PRINT "A   B   C   D
        E   F     FREQUENCY":LOCATE 11,3
     :PRINT "G   H   I   J   K   L":LOCA
     TE 15,3:PRINT "M   N   O   P   Q
     R":LOCATE 19,3:PRINT "S   T   U   V
        W   X    TRIP":PRINT
1200 PRINT TAB(28);"SLAM":LOCATE 5,27:PR
     INT "6     8    10";
1210 FOR Z=0 TO 2:LINE (16+Z,Z)-(87-Z,23
     -Z),3,B:LINE (104+Z,Z)-(175-Z,23-Z)
     ,3,B:LINE (16+Z,167+Z)-(87-Z,190-Z)
     ,3,B:LINE (104+Z,167+Z)-(175-Z,190-
     Z),3,B:NEXT
1220 FOR Z=31 TO 159 STEP 32:LINE (0,Z)-
     (192,Z),3:LINE (0,Z+1)-(192,Z+1),3:
     NEXT:FOR Z=0 TO 192 STEP 32:LINE (Z
     ,32)-(Z,160),3:LINE (Z+1,32)-(Z+1,1
     60),3:NEXT:LINE (0,94)-(193,97),1,B
     F:LINE (95,0)-(98,191),1,BF:LINE (0
     ,0)-(193,191),1,B
1230 FOR Z=0 TO 1:LINE (200+Z,8+Z)-(311-
     Z,31-Z),3,B:LINE (195+Z,56+Z)-(317-
     Z,123-Z),3,B:NEXT:LINE (256,160)-(3
     00,167),1,BF:LINE (256,144)-(300,15
     1),1,BF
1240 FOR Z=4.71 TO 1.57 STEP -.1:IF Z<2.
     3 OR Z>4.1 THEN C=2 ELSE C=1
1250 LINE (SIN(Z)*40+255,COS(Z)*40+114)-
     (SIN(Z)*45+255,COS(Z)*45+114),C:NEX
     T:DRAW "BM248,121C3E7C2NU35C3F7BM25
     5,117P3,3"
1260 DRAW "BM203,114C3UH2GD2L2U3BU3UEFDF
     REUHLGDGHUBM253,59G2D2FREHLBR4DFREU
     3HLGD2BM308,114U3RFDFREU2BU3H2L2GDF
     EU"
1270 FOR X=1 TO 4:FOR Z=0 TO 55 STEP 2:L
     INE (Z+BP(X,1),BP(X,2)+12)-(Z+BP(X,
     1),BP(X,2)+16),1:NEXT:NEXT:FOR Z=0
     TO 95 STEP 2:LINE (Z+208,24)-(Z+208
     ,29),1:NEXT
1280 RD$="C1D8RC2NR6LC1D8NR15FNR9DNR8FNR
     7DNR7FNR7DR12HRU4LUL7NU16BH4P2,1"
1290 FOR Z=1 TO 24:X=RP(Z,1)-1:Y=RP(Z,2)
     :DRAW "BM=X;,=Y;XRD$;":NEXT
1300 FOR Z=1 TO 3:LINE (200+Z,168+Z)-(31
     1-Z,191-Z),Z,B:NEXT
1310 RETURN
1320 '
1330 '  SCAN KEYBOARD
1340 '
1350 A$=INKEY$:IF A$="" THEN 1350 ELSE R
     ETURN
1360 '
1370 '  METER AND ROD POSITION DATA
1380 '
1390 DATA 8,40,40,40,72,40,104,40,136,40
     ,168,40,8,72,40,72,72,72,104,72,136
     ,72,168,72,8,104,40,104,72,104,104,
     104,136,104,168,104,8,136,40,136,72
     ,136,104,136,136,136,168,136
1400 DATA 24,4,112,4,24,171,112,171
```

HCM

# OVER-REACTION

```
100  REM  ******************
110  REM  * OVER-REACTION *
120  REM  ******************
130  REM  COPYRIGHT 1985
140  REM  EMERALD VALLEY PUBLISHING CO.
150  REM  BY WILLIAM K. BALTHROP
160  REM  HOME COMPUTER MAGAZINE
170  REM  VERSION 5.3.1
180  REM  TI BASIC OR
190  REM  TI EXTENDED BASIC
200  REM
210  CALL CLEAR
220  RANDOMIZE
230  PRINT TAB(8);"OVER-REACTION"::::::
     :"     PLACE ALPHA LOCK UP":::::"IN
     ITIALIZING":"PLEASE WAIT. . ."
240  OPTION BASE 1
250  DIM S(4,6),TS(4),BT(4),INST(4),CSYS
     (4),BP(4,2)
260  RESTORE 3250
270  FOR Z=1 TO 41
280  READ A,A$
290  CALL CHAR(A,A$)
300  NEXT Z
310  FOR Z=9 TO 14
320  READ F,B
330  CALL COLOR(Z,F,B)
340  NEXT Z
350  CALL CLEAR
360  FOR Z=1 TO 8
370  CALL COLOR(Z,2,15)
380  NEXT Z
390  FOR Z=1 TO 4
400  READ BP(Z,1),BP(Z,2)
410  NEXT Z
420  GOSUB 520
430  GOTO 900
440  CALL SCREEN(15)
450  PRINT TAB(8);"OVER-REACTION":::"ENT
     ER SKILL LEVEL:":::"1) NORMAL OPERA
     TION"::
460  PRINT "2) FAULTY SYSTEMS"::"3) EART
     HQUAKE COUNTRY"::"4) ARMAGEDDON":::
     ::::"CHOOSE ONE:"
470  GOSUB 3120
480  IF (K<49)+(K>52)THEN 470
490  CALL CLEAR
500  ON K-48 GOSUB 1500,1560,1610,1670
510  RETURN
520  RESTORE 3370
530  GOSUB 440
540  FOR Z=1 TO 24
550  READ A$
560  IF Z=24 THEN 590
570  PRINT A$
580  GOTO 600
590  PRINT A$;
600  NEXT Z
610  CALL VCHAR(1,31,32,96)
620  FOR RY=1 TO 4
630  FOR RX=1 TO 6
640  S(RY,RX)=8
650  GOSUB 3150
660  NEXT RX
670  CSYS(RY)=0
680  INST(RY)=0
690  TS(RY)=48
700  BT(RY)=48
710  Z=RY
720  GOSUB 2290
730  NEXT RY
740  STOUT=0
750  HZOUT=0
760  BTOUT=0
770  DONE=0
780  LOAD=1
790  STP=64
800  TTS=64
810  HZ=60
820  STDAM=0
830  CT=.05
840  LDP=1
850  LDN=(RND*.067)+.967
860  LP=.05
870  GOSUB 2530
880  GOSUB 2640
890  RETURN
900  GOSUB 2720
910  SCORE=SCORE+(2-(ABS(60-HZ)))
920  IF DONE=1 THEN 2050
930  FOR Z=1 TO 4
940  F=20
950  IF CSYS(Z)=0 THEN 970
960  F=FLUX
970  BT(Z)=BT(Z)+((TS(Z)+SIN((CT+Z*1.5))
     *F)-BT(Z))/3
980  IF INST(Z)=0 THEN 1000
990  CALL HCHAR(BP(Z,1),BP(Z,2),121,8)
1000 GOSUB 2290
1010 NEXT Z
1020 TTS=((BT(1)+BT(2)+BT(3)+BT(4))/3)-S
     TDAM
1030 IF TTS=STP THEN 1050
1040 GOSUB 2530
1050 GOSUB 2640
1060 IF INT(CT)<>CT THEN 1130
1070 LDP=LDN
1080 LP=.05
1090 IF PROBS<4 THEN 1120
1100 LDN=(RND*.045)+.99
1110 GOTO 1130
1120 LDN=(RND*.067)+.967
1130 LOAD=(LDP*(1-LP))+(LDN*LP)
1140 LP=.05+LP
1150 IF PROBS=1 THEN 1180
1160 IF INT(RND*15)THEN 1180
1170 GOSUB 1210
1180 CT=CT+.05
1190 IF (STOUT=1)+(BTOUT=1)+(HZOUT=1)THE
     N 1740
1200 GOTO 900
1210 ON PROBS-1 GOSUB 1230,1410,1410
1220 GOTO 900
1230 IF INT(RND*5)>2 THEN 1340
1240 C=INT(RND*4)+1
1250 IF CSYS(C)=1 THEN 1330
1260 CSYS(C)=1
1270 FLUX=INT(RND*20)+20
1280 FOR N=1 TO 3
1290 FOR M=300 TO 2000 STEP 176
1300 CALL SOUND(-100,M,0,M+10,0,M+50,0)
1310 NEXT M
1320 NEXT N
1330 RETURN
1340 I=INT(RND*4)+1
1350 IF INST(I)=1 THEN 1400
1360 INST(I)=1
1370 FOR N=0 TO 30 STEP 2
1380 CALL SOUND(-100,500-N*10,N,600-N*10
     ,N,700-N*10,N,-7,N)
1390 NEXT N
1400 RETURN
1410 P=INT(RND*8)+1
1420 IF PROBS<4 THEN 1440
1430 ON P GOTO 1270,1270,1270,1270,1340,
     1340,1340,1450
1440 ON P GOTO 1270,1270,1270,1340,1340,
     1450,1450,1450
1450 STDAM=STDAM+RND*6+.2
1460 FOR N=0 TO 30
1470 CALL SOUND(-100,660,30-N,880,30-N,1
     200,0,-7,N/2)
1480 NEXT N
1490 RETURN
1500 PRINT "YOU MAY ENCOUNTER MINOR":"PR
     OBLEMS WITHIN THE REACTOR;HOWEVER,
     A COOL HEAD SHOULD"
1510 PRINT "BE ABLE TO KEEP THINGS UNDER
     CONTROL.":"WATCH FOR OVERHEATING I
     N THEREACTOR CORES, AND CHANGES"
1520 PRINT "IN THE LOAD.":::::
1530 SKILL=750
1540 PROBS=1
1550 GOTO 1710
1560 PRINT "IT'S BEEN ONE OF THOSE DAYS!
     ELECTRICAL PROBLEMS IN THE  COOLING
      SYSTEM AND CONTROL"
1570 PRINT "PANEL HAVE YOU IN A FRENZY."
     ::"WATCH REACTOR TEMPERATURES!"::::
1580 SKILL=700
1590 PROBS=2
1600 GOTO 1710
1610 PRINT "YOUR REACTOR IS IN THE HEART
     OF CALIFORNIA EARTHQUAKE   COUNTRY
     !!!!"::
1620 PRINT "BE PREPARED FOR DRASTIC AND
     SUDDEN CHANGES IN THE":"REACTOR. IR
     REVERSIBLE DAMAGEMAY RESULT."::
1630 PRINT "WATCH THE STEAM PRESSURE!":"
     INTERNAL LEAKS MAY DEVELOP  IF AN E
     ARTHQUAKE HITS.":::::
1640 SKILL=600
1650 PROBS=3
1660 GOTO 1710
1670 PRINT "OUR WORST FEARS ARE NOW A
     REALITY. NUCLEAR WAR IS ON  THE VER
     GE OF ERUPTING. THE"
1680 PRINT "LOAD ON YOUR REACTOR IS":"TE
     RRIFIC AS THE WORLD":"PREPARES FOR
     WAR.":"RIOTS ARE INCREASING!!":::::
1690 SKILL=450
1700 PROBS=4
1710 PRINT ::::"PRESS ENTER TO CONTINUE"
     ::
1720 GOSUB 3120
1730 RETURN
1740 IF BTOUT=1 THEN 1780
1750 IF STOUT=1 THEN 1850
1760 IF HZOUT=1 THEN 1910
1770 GOTO 900
1780 GOSUB 2430
1790 CALL CLEAR
1800 PRINT "MELTDOWN!!!"
1810 PRINT "THE REACTOR CORE TEMPERATURE
     HAS EXCEEDED SAFE LIMITS.   MELTDOW
     N HAS BEGUN AND THE"
1820 PRINT "SURROUNDING AREA MUST BE
     EVACUATED.":::::
1830 SCORE=SCORE*.2
1840 GOTO 2090
1850 GOSUB 2430
1860 CALL CLEAR
1870 PRINT "THE REACTOR STEAM PRESSURE
     HAS CAUSED A MAJOR LEAK    OUTSIDE
      THE SYSTEM AND A"
```

Continued

~~~
1880 PRINT "MELTDOWN HAS BEGUN. THE AREA
     AROUND THE PLANT IS BECOMINGCONTAMI
     NATED AND MUST BE    EVACUATED.":::
1890 SCORE=SCORE*.2
1900 GOTO 2090
1910 FOR I=1 TO 8
1920 CALL HCHAR(22,26,121,1)
1930 CALL SOUND(100,110,1)
1940 FOR DELAY=1 TO 50
1950 NEXT DELAY
1960 CALL HCHAR(22,26,120,1)
1970 FOR DELAY=1 TO 50
1980 NEXT DELAY
1990 NEXT I
2000 CALL CLEAR
2010 PRINT "YOU WERE UNABLE TO MAINTAIN
     THE OPERATING FREQUENCY OF  THE REA
     CTOR'S POWER OUTPUT."
2020 PRINT :::"THE REACTOR'S MAIN BREAKE
     R   HAS BEEN TRIPPED. YOU WILL  BE S
     EVERELY REPRIMANDED.":::::
2030 SCORE=SCORE*.5
2040 GOTO 2090
2050 GOSUB 2180
2060 CALL CLEAR
2070 PRINT "THE REACTOR HAS SHUT DOWN":"
     SAFELY. YOU MAY RESTART THE REACTOR
      AFTER ALL COMPONENTS"
2080 PRINT "HAVE BEEN INSPECTED AND":"RE
     PAIRED.":::::
2090 PRINT "YOUR SCORE IS:";INT(SCORE)*1
     00::::"WOULD YOU LIKE TO PLAY AGAIN
     (Y/N)?"
2100 GOSUB 3120
2110 IF (K=89)+(K=121)THEN 2150
2120 IF (K<>78)*(K<>110)THEN 2100
2130 CALL CLEAR
2140 END
2150 PRINT ::::"REPAIRING REACTOR...":::
     :
2160 GOSUB 520
2170 GOTO 900
2180 FOR RY=1 TO 4
2190 FOR RX=1 TO 6
2200 CALL SOUND(200,330,RY*RX,-3,RX*RY-1
     )
2210 CALL HCHAR(22,24,120,1)
2220 CALL SOUND(200,110,30)
2230 S(RY,RX)=16
2240 GOSUB 3150
2250 CALL HCHAR(22,24,121,1)
2260 NEXT RX
2270 NEXT RY
2280 RETURN
2290 BM=BT(Z)
2300 IF BM>1 THEN 2340
2310 BM=1
2320 BT(Z)=1
2330 GOTO 2370
2340 IF BM>90 THEN 2510
2350 IF BM<86 THEN 2370
2360 CALL SOUND(500,-3,15)
2370 IF INST(Z)THEN 2420
2380 IF BM<12 THEN 2400
2390 CALL HCHAR(BP(Z,1),BP(Z,2)+INT(BM/1
     2)-1,32)
2400 CALL HCHAR(BP(Z,1),BP(Z,2)+INT((BM-
     1)/12)+1,32)
2410 CALL HCHAR(BP(Z,1),BP(Z,2)+INT(BM/1
     2),140+INT((BM-INT(BM/12)*12)/3))
2420 RETURN
2430 FOR RY=1 TO 4
2440 FOR RX=1 TO 6
2450 CALL SOUND(200,110+(RX*26),1,-3,1)
2460 GOSUB 3220
2470 NEXT RX
2480 NEXT RY
2490 CALL SOUND(500,110+(RX*40),1,-3,1)
2500 RETURN
2510 BTOUT=1
2520 RETURN
2530 STP=STP+(TTS-STP)/15
2540 ST=INT(STP*4-192)
2550 IF ST>0 THEN 2580
2560 ST=1
2570 GOTO 2610
2580 IF ST<128 THEN 2610
2590 STOUT=1
2600 RETURN
2610 CALL VCHAR(3,26,32,16)
2620 CALL VCHAR(18-INT(ST/8),26,139-INT(
     (ST-INT(ST/8)*8)/2))
2630 RETURN
2640 HZ=(STP/4+48)*.9375*LOAD
2650 HZM=(HZ-58)*36
2660 IF (HZM>=0)*(HZM<=144)THEN 2690
2670 HZOUT=1
2680 RETURN
2690 CALL HCHAR(22,4,32,18)
2700 CALL HCHAR(22,4+INT(HZM/8),140+INT(
     (HZM-INT(HZM/8)*8)/2))
2710 RETURN
2720 CALL KEY(0,K,A)
2730 IF A<>0 THEN 2750
2740 RETURN
2750 IF ((K>64)*(K<89))+((K>96)*(K<121))
     THEN 2790
2760 IF K<>3 THEN 2780
2770 DONE=1
2780 RETURN
2790 CALL SOUND(-1,880,0)
2800 IF K>88 THEN 2910
2810 K=K-64
2820 RY=INT(K/6.001)+1
2830 RX=K-(RY-1)*6
2840 IF S(RY,RX)<16 THEN 2860
2850 RETURN
2860 S(RY,RX)=S(RY,RX)+1
2870 GOSUB 3150
2880 CV=-1
2890 GOSUB 3010
2900 RETURN
2910 K=K-96
2920 RY=INT(K/6.001)+1
2930 RX=K-(RY-1)*6
2940 IF S(RY,RX)>0 THEN 2960
2950 RETURN
2960 S(RY,RX)=S(RY,RX)-1
2970 GOSUB 3150
2980 CV=1
2990 GOSUB 3010
3000 RETURN
3010 IF RY>2 THEN 3070
3020 IF RX>3 THEN 3050
3030 TS(1)=TS(1)+CV
3040 RETURN
3050 TS(2)=TS(2)+CV
3060 RETURN
3070 IF RX>3 THEN 3100
3080 TS(3)=TS(3)+CV
3090 RETURN
3100 TS(4)=TS(4)+CV
3110 RETURN
3120 CALL KEY(0,K,A)
3130 IF A=0 THEN 3120
3140 RETURN
3150 IF S(RY,RX)>8 THEN 3190
3160 CALL HCHAR((RY-1)*3+5,(RX-1)*3+5,12
     7)
3170 CALL HCHAR((RY-1)*3+6,(RX-1)*3+5,S(
     RY,RX)+127)
3180 RETURN
3190 CALL HCHAR((RY-1)*3+5,(RX-1)*3+5,S(
     RY,RX)+119)
3200 CALL HCHAR((RY-1)*3+6,(RX-1)*3+5,13
     5)
3210 RETURN
3220 CALL HCHAR((RY-1)*3+5,(RX-1)*3+5,12
     1)
3230 CALL HCHAR((RY-1)*3+6,(RX-1)*3+5,12
     1)
3240 RETURN
3250 DATA 127,8181818181818181
3260 DATA 128,81818181818181FF,129,81818
     18181FFFF,130,818181818181FFFFFF,13
     1,818181FFFFFFFF
3270 DATA 132,818181FFFFFFFFFF,133,8181F
     FFFFFFFFFFF,134,81FFFFFFFFFFFFFF,13
     5,FFFFFFFFFFFFFFFF
3280 DATA 104,000000FF,105,8080808080808
     080,106,808080FF80808080,107,000000
     FF80808080,108,808080FF
3290 DATA 109,80808080,110,0000008080808
     080,112,0101010101010101,113,010101
     FFFF010101,114,808080FFFF808080
3300 DATA 115,000000FFFF000000,116,00000
     0FF01010101,117,01010101FF,118,010101
     FF01010101
3310 DATA 96,FFFFAAAAAAAA404040,97,404040A
     AAAAAFFFF,99,031F031F031F03FF,98,C0
     F8C0F8C0F8C0FF,100,FFFF
3320 DATA 101,0303030303030303,102,00000
     000000000FFFF,103,C0C0C0C0C0C0C0C0,13
     6,FFFF,137,0000FFFF
3330 DATA 138,000000000000FFFF,139,00000000
     000FFFF,140,C0C0C0C0C0C0C0C0,141,30
     30303030303030
3340 DATA 142,0C0C0C0C0C0C0C0C,143,03030
     30303030303,120,FF000000000000FF,12
     1,FFFFFFFFFFFFFFFF
3350 DATA 2,15,5,15,5,15,7,15,3,15,7,15
3360 DATA 2,4,2,14,18,4,18,14
3370 DATA "e````````ge`````` g X100","e
              ge          g    fff10","efffff
     fffgeffffffffg Sb c"
3380 DATA "  khhkhhkhvjhhkhhkhhn" Tb c","
     i  i  i pi  i  i  i Eb c", "i Ai Bi
      C i D i E i F i Ab c9"
3390 DATA "  jhhjhhjhvjhhjhhjhhi" Mb c","
     i  i  i pi  i  i  i  b c", "i Gi Hi
      I i J i K i L i Pb c"
3400 DATA " srssrssrsqrssrssrssr" Rb c8","
      i  i  i pi  i  i  i Eb c", "i Mi N
     i O i P i Q i R i Sb c"
3410 DATA "  jhhjhhjhvjhhjhhjhhi" Sb c","
     i  i  i pi  i  i  i Ub c7", "i Si T
     i U i V i W i X i Rb c"
3420 DATA "  lhhlhhlhvjhhlhhlhhm" Eb c","e
     ````````ge````````g b c", "e
      ge        g    b c6
3430 DATA "efffffffffgefffffffgddd","
        FREQUENCY", "edddddddddddddddd
     dg S T", "e        g  x x
     "
3440 DATA "eaaaaaaaaaaaaaaaaaaag","58
        60         62"
~~~

# TORPEDO ALLEY

PROGRAM LISTING

```
100 REM  *****************
110 REM  * TORPEDO ALLEY *
120 REM  *****************
130 REM  COPYRIGHT 1985
140 REM  EMERALD VALLEY PUBLISHING CO
150 REM  BY WILLIAM K. BALTHROP
160 REM  AND THE HCM STAFF
170 REM  HOME COMPUTER MAGAZINE
180 REM  VERSION 5.3.1
190 REM  APPLE II FAMILY APPLESOFT
200 REM
210 REM
220 IF  PEEK (103) = 1 AND  PEEK (104)
    = 64 THEN 250
230 POKE 103,1: POKE 104,64: POKE 16384
    ,0: POKE 16385,0: POKE 16386,0
240 PRINT "RUN TORPEDO"
250 REM  INTRO
260 TEXT : POKE 34,0: HOME : SPEED= 255
    : NOTRACE
270 VTAB 12: HTAB 12: PRINT "TORPEDO AL
    LEY": VTAB 22: HTAB 8: PRINT "PRESS
     RETURN TO START": VTAB 24: HTAB 5:
     PRINT "BE SURE CAPS LOCK IS DOWN"
280 GET DUMMY$: IF DUMMY$ <  > CHR$ (1
    3) THEN  GOTO 280
290 CLEAR : HOME : VTAB 5: HTAB 5: PRIN
    T "ARE YOU USING JOYSTICKS? (Y/N) "
    ;
300 GET YN$: IF YN$ <  > "N" AND YN$ <
     > "Y" THEN 300
310 IF YN$ = "Y" THEN JOYSTICK = 1
320 PRINT YN$: VTAB 7: PRINT "PLEASE WA
    IT WHILE I PREPARE YOUR SUB."
330 DIM QW(21),CX(21),CY(21),P(2)
340 THESUB = 2:SHIP = 1:BOMB = 3:FIRE =
     4:BANG = 5:P(0) = 6:P(1) = 7:P(2)
    = 8:WAVE = 9
350 WHITE = 7:BLUE = 6:RED = 5
360 SCALE= 1: ROT= 0
370 SP = 1:SUP = 30:SS = 0:SC = 0:DC =
    0:TT = 0:DB = 1
380 SOUND = 768
390 REM  SET UP SHAPE TABLES
400 FOR A = 24576 TO 25053: READ B: POK
    E A,B: NEXT : POKE 232,0: POKE 233,
    96
410 FOR A = SOUND TO SOUND + 19: READ B
    : POKE A,B: NEXT
420 REM  DISPLAY SCREEN
430 HGR : HCOLOR= WHITE: HPLOT 20,10 TO
     260,10 TO 260,159 TO 20,159 TO 20,
    10
440 HCOLOR= BLUE: FOR Y = 40 TO 158: HP
    LOT 21,Y TO 258,Y: NEXT
450 HCOLOR= WHITE: HPLOT 140,10 TO 140,
    159
460 FOR X = 10 TO 158 STEP 10: HPLOT 13
    6,X TO 144,X: HPLOT 138,X + 5 TO 14
    2,X + 5: NEXT
470 VTAB 22: HTAB 1: PRINT "     SCORE:
                    SUPPLY:"
480 VTAB 23: HTAB 1: PRINT "
                              ";: VTA
    B 23: HTAB 5: PRINT SC;: HTAB 30: P
    RINT SUP;
490 REM  POKE 34,22
500 REM  ****** MAIN LOOP
510 IF SS = 0 THEN  GOSUB 550
520 IF ((B < 123 AND SP =  - 1) OR (B >
     157 AND SP = 1)) THEN  GOSUB 1570
530 IF SUP = 0 THEN 1050
540 GOSUB 590: GOSUB 730: GOTO 510
550 REM  PLACE A SHIP IN THE     PER
    ISCOPE
560 A =  INT ( RND (1) * 50) + 60:SY =
    ( INT ( RND (1) * 3) + 1) / 4:S =
    INT ( RND (1) * 2): IF S = 1 THEN S
    P =  SGN ( - SP)
570 B = 279 + SP * 25: IF B > 279 THEN
    B = B - 279
580 SS = 1: RETURN
590 REM  ********** SCAN KB
600 IF JOYSTICK THEN 680
610 K =  PEEK ( - 16384)
620 IF K = 203 THEN B = B - .75
630 IF K = 202 THEN B = B + .75
640 IF K = 160 THEN  GOSUB 850
650 IF K = 205 THEN  POKE  - 16368,0: G
    OTO 1150
660 POKE  - 16368,0
670 RETURN
680 REM  ******** SCAN JOYSTICK
690 B = B - ( PDL (0) - 128) / 160
700 IF  PDL (1) > 192 THEN  GOTO 1150
710 IF  PEEK ( - 16287) > 127 THEN  GOS
    UB 850
720 RETURN
730 REM  *** MOVE SHIP & ZIG ZAG
740 HCOLOR= RED
750 IF B < 25 OR B > 243 THEN SS = 0
760 IF SS = 0 THEN  RETURN
770 IF A > 120 OR A < 50 THEN SY =  - S
    Y
780 A = A + SY:B = B + SP
790 REM  DRAW SHIP
800 IF  NOT FR AND XO THEN  XDRAW SHIP
    AT XO,YO
810 XDRAW SHIP AT B,A
820 XO = B:YO = A:FR = 0
830 IF ((B < 123 AND SP =  - 1) OR (B >
     157 AND SP = 1)) THEN  GOSUB 1570:
     IF SUP < 0 THEN SUP = 0
840 RETURN
850 REM  FIRE TORPEDO
860 SUP = SUP - 1: HCOLOR= WHITE
870 NS = NS + 1
880 VTAB 23: HTAB 1: PRINT "
                              ";: VTA
    B 23: HTAB 5: PRINT SC;: HTAB 30: P
    RINT SUP;
890 FOR TA = 0 TO 2:TD = 159 - TA * 39
900 FOR I = TD TO TD - 38 STEP  - 3 + T
    A
910 IF I <  > TD THEN  XDRAW P(TA) AT 1
    45,I + 3 - TA
920 XDRAW P(TA) AT 145,I
930 IF  ABS (B - 145) < 7 AND  ABS (I -
     A) < 4 THEN  GOTO 1000
940 IF A - I > 3 THEN  XDRAW P(TA) AT 1
    45,I: RETURN
950 GOSUB 730: FOR PAUSE = 1 TO 50: NEX
    T : GOSUB 730
960 NEXT I
970 IF X > 50 THEN  XDRAW P(TA) AT 145,
    I + 3 - TA
980 NEXT TA
990 RETURN
1000 REM  HIT
1010 XDRAW P(TA) AT 145,I
1020 HCOLOR= RED: XDRAW BANG AT B,A:SS =
     0:SC =  INT (SC + 107 - A):TT = TT
     + 1: GOSUB 1660: XDRAW BANG AT B,A
1030 VTAB 23: HTAB 1: PRINT "
                              ";: VTA
    B 23: HTAB 5: PRINT SC;: HTAB 30: P
    RINT SUP;
1040 RETURN
1050 REM  GAME OVER
1060 POKE 34,0: HOME : TEXT
1070 VTAB 5: HTAB 1: PRINT "YOU HAVE SUN
    K ";TT;" OF THE ENEMY SHIPS."
1080 VTAB 8: PRINT "YOUR ACCURACY IS ";
1090 IF NS = 0 THEN  PRINT "0%": GOTO 11
    10
1100 IF SUP <  > 30 THEN  PRINT  INT ((T
    T / NS) * 100 + .5);"%"
1110 VTAB 12: PRINT "YOUR FINAL SCORE IS
     ";SC;"."
1120 VTAB 24: HTAB 1: PRINT "PLAY AGAIN?
     (Y/N)";: GET YN$: IF YN$ = "Y" TH
    EN 290
1130 IF YN$ <  > "N" THEN 1120
1140 HOME : END
1150 REM  ********* DIVE UNDER WATER
1160 LD = 300: HGR : HCOLOR= WHITE: HPLO
    T 0,0 TO 279,0 TO 279,159 TO 0,159
    TO 0,0
1170 HCOLOR= BLUE:SX = 140:SV = 35:SL =
    260
1180 FOR WY = 158 TO 31 STEP  - 1: HPLOT
     1,WY TO 278,WY: NEXT
1190 HCOLOR= WHITE: FOR SV = 35 TO 100:
    XDRAW THESUB AT SX,SV: FOR PAUSE =
    1 TO 100: NEXT : XDRAW THESUB AT SX
    ,SV: NEXT :OT = SX:OU = SV: XDRAW T
    HESUB AT SX,SV
1200 XDRAW SHIP AT SL,30:OS = SL
1210 IF  NOT JOYSTICK GOTO 1270
1220 IF  PDL (0) > 192 THEN SX = SX + 1:
     IF SX > 266 THEN SX = 266
1230 IF  PDL (0) < 63 THEN SX = SX - 1:
    IF SX < 12 THEN SX = 12
1240 IF  PDL (1) > 192 THEN SV = SV + 1:
     IF SV > 156 THEN SV = 156
1250 IF  PDL (1) < 63 THEN SV = SV - 1:
    IF SV < 35 THEN SV = 35
1260 GOTO 1330
1270 K =  PEEK ( - 16384)
1280 IF K = 201 THEN SV = SV - 1: IF SV
    < 35 THEN SV = 35
1290 IF K = 205 THEN SV = SV + 1: IF SV
    > 156 THEN SV = 156
1300 IF K = 202 THEN SX = SX - 1: IF SX
    < 12 THEN SX = 12
1310 IF K = 203 THEN SX = SX + 1: IF SX
    > 266 THEN SX = 266
1320 POKE  - 16368,0
1330 IF  NOT (DC - DB + 1) THEN 1400
1340 FOR CHARGE = DB TO DC: IF CY(CHARGE
    ) < SV - 4 THEN 1390
1350 HCOLOR= RED: XDRAW BANG AT CX(CHARG
    E),CY(CHARGE): GOSUB 1710: XDRAW BA
    NG AT CX(CHARGE),CY(CHARGE): HCOLOR
    = WHITE: XDRAW BOMB AT CX(CHARGE),C
    Y(CHARGE)
1360 DB = DB + 1: IF DB = 21 THEN 1560
1370 IF CX(CHARGE) < = SX THEN  IF SX -
     CX(CHARGE) < 15 THEN  GOSUB 1740:
    GOTO 1050
```

Continued

## TORPEDO ALLEY *Continued*

### APPLE // Family

```
1380 IF CX(CHARGE) > = SX THEN IF CX(CHARGE) - SX < 23 THEN GOSUB 1740: GOTO 1050
1390 CY(CHARGE) = CY(CHARGE) + 1.5: NEXT CHARGE
1400 REM UPDATE SCREEN2
1410 HCOLOR= RED
1420 IF OS THEN XDRAW SHIP AT OS,30:OS = SL
1430 XDRAW SHIP AT SL,30
1440 SL = SL - 3: IF SL < 1 THEN SL = 279:LD = 300
1450 HCOLOR= WHITE
1460 IF OT THEN XDRAW THESUB AT OT,OU:OT = SX:OU = SV
1470 XDRAW THESUB AT SX,SV
1480 IF NOT (DC - DB + 1) THEN 1530
1490 FOR CHARGE = DB TO DC
1500 IF CY(CHARGE) > 25 THEN XDRAW BOMB AT CX(CHARGE),CY(CHARGE) - 1.5
1510 XDRAW BOMB AT CX(CHARGE),CY(CHARGE)
1520 NEXT CHARGE
1530 IF DC = 20 THEN 1210
1540 IF ABS (SL - SX) < 52 AND INT ( RND (1) * 20) > 16 AND LD - SL > 10 THEN DC = DC + 1:CX(DC) = SL:CY(DC) = 23:LD = SL
1550 GOTO 1210
1560 FR = 1:DB = 1:DC = 0: GOTO 420
1570 REM SHIP FIRES AT YOU
1580 IF NOT ( INT ( RND (1) * 75) = 1) THEN RETURN
1590 HCOLOR= RED: XDRAW FIRE AT B,A
1600 POKE SOUND + 1,85: CALL SOUND
1610 XDRAW FIRE AT B,A: FOR PAUSE = 1 TO 100: NEXT PAUSE
1620 GOSUB 1680: REM TUNE WHEN HIT
1630 SUP = SUP - INT ( RND (1) * 9 + 1): IF SUP < 0 THEN SUP = 0
1640 VTAB 23: HTAB 1: PRINT "                    ";: VTAB 23: HTAB 5: PRINT SC;: HTAB 30: PRINT SUP;
1650 RETURN
1660 REM NOISE WHEN HIT SHIP
1670 FOR QW = 1 TO 4: POKE SOUND + 1,QW: CALL SOUND: NEXT : RETURN
1680 REM TUNE WHEN HIT
1690 POKE SOUND + 1,5: CALL SOUND: CALL SOUND
1700 RETURN
1710 REM SOUND OF DEPTH CHARGE EXPLODING
1720 POKE SOUND + 1,5: CALL SOUND
1730 RETURN
1740 REM MUSIC FOR DEPTH CHARGE HIT
1750 FOR Z = 1 TO 8: POKE SOUND + 1,Z: CALL SOUND: NEXT : RETURN
1760 DATA 10,0,22,0,72,0,101,0,114,0,211,0,175,1,191,1,204,1,213,1,220,1,45,45,45,45,45,37,61,63,63,63,63,63,63,60,45,45,45,45,45,45,45,37,61,63,63,63,63,63,63,63,63
1770 DATA 28,8,24,8,53,53,45,42,45,44,46,44,46,45,12,37,37,45,0,45,45,45,45,45,37,37,60,60,63,63,63,63,63,63,36,63,60,62,54,63,62,54,53,45,45,45,45,0,45
1780 DATA 44,36,39,63,63,63,55,55,46,46,45,0,36,37,36,60,36,54,54,55,63,60,60,63,36,54,54,46,46,45,53,55,63,63,63,63,60,57,9,9,9,9,9,9,9,44,44,45,37,37,36
1790 DATA 45,37,54,55,55,21,45,37,37,45,37,18,18,19,27,27,27,27,27,27,45,37,45,45,45,45,37,45,53,45,53,63,63,63,62,60,63,63,63,23,63,63,63,63,55,63,56,27,27,27,36
1800 DATA 63,39,54,46,46,45,0,36,44,44,36,39,39,34,18,62,54,54,55,36,39,39,39,36,60,60,62,55,53,53,54,60,63,39,63,54,61,45,45,45,54,63,60,63,60,63,63,55,63,9,53
1810 DATA 37,46,45,45,55,55,63,62,63,55,62,45,45,37,45,44,44,44,46,46,47,63,53,53,55,62,45,45,36,44,36,62,60,37,45,54,54,46,46,55,53,45,54,44,44,39,39,39,36,60,45
1820 DATA 45,53,53,46,46,54,47,53,44,36,60,39,39,37,45,53,45,53,44,39,60,60,55,36,63,55,63,55,63,44,44,44,45,37,39,37,45,46,46,14,63,62,45,45,45,45,53,45,44,44,63
1830 DATA 63,60,37,44,63,62,55,62,54,53,53,45,53,53,45,46,55,63,39,63,39,45,39,63,60,39,63,36,36,36,36,44,44,36,60,60,62,55,53,55,55,62,45,37,36,42,18,23,63,63,36
1840 DATA 39,39,54,54,54,62,62,52,44,36,36,36,56,36,55,55,55,46,44,37,45,0,45,37,39,39,60,60,60,62,62,54,55,55,45,45,45,0,45,60,60,36,39,55,55,62,62,46,45,45,0,45
1850 DATA 60,60,60,62,62,46,45,0,45,46,45,46,45,44,45,44,45,0
1860 DATA 162,0,234,134,6,169,0,168,136,208,253,44,48,192,56,229,6,208,244,96
```

HCM

## TORPEDO ALLEY

### COMMODORE 64

```
100 REM ****************
110 REM * TORPEDO ALLEY *
120 REM ****************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM BY WILLIAM K. BALTHROP
160 REM AND THE HCM STAFF
170 REM HOME COMPUTER MAGAZINE
180 REM VERSION 5.3.1
190 REM C-64 BASIC
200 REM
210 REM RECONFIGURE THE VIC CHIP
220 POKE 56576,PEEK(56576)AND253:POKE 53272,4:POKE 648,128:POKE 56,128:CLR
230 REM DISPLAY TITLE SCREEN
240 POKE 53281,6:POKE 53280,5
250 T$="TORPEDO ALLEY"
260 PRINT"{SHIFT CLR}{CTRL WHT}{10CRSRDOWN}        **  "T$"  **"
270 PRINT"{13CRSRDOWN}               PRESS{SPACE}RETURN{SPACE}TO{SPACE}CONTINUE";:X=13:Y=10:S=1
280 GET K$:GOSUB2370:S=ABS(S-1):POKE 199,S
290 FOR I=1 TO 50:NEXT:PRINTT$:IF K$="" THEN280
300 IF ASC(K$)<>13 THEN280
310 PRINT"{SHIFT CLR}{10CRSRDOWN}PLEASE WAIT WHILE I PREPARE"
320 PRINT"                YOUR SUBMARINE"
330 REM INITIALIZE SPRITES AND VARIABLES
340 V=53248:SP=33784:V1=54276:V2=V1+7:V3=V2+7:S1=129:S2=17:S3=129:I=RND(-TI)
350 DIMDC(7,1):FORI=33792 TO 34367:READ D:POKE I,D:NEXT:POKE SP,16:POKE SP+1,17
360 POKE SP+5,20:POKE SP+6,20:POKE SP+7,20:POKE V+39,0:POKE V+40,3:POKE V+42,1
370 POKE V+44,1:POKE V+45,1:POKE V+46,1:POKE V+37,1:POKE V+38,1
380 FOR I=54272 TO 54296:READ D:POKE I,D:NEXT:SU=30:SC=0:TL=SC
390 REM ENTER OPTION TO USE JOYSTICK
400 POKE 198,0:PRINT"{SHIFT CLR}{11CRSRDOWN} JOYSTICK - Y OR N:";
410 GET K$:IF K$<>"Y" AND K$<>"N" THEN410
420 PRINT K$:J=0:IF K$="Y" THEN J=-1:PRINT"{CRSRDOWN} PLUG JOYSTICK INTO CONTROL PORT 2"
430 IF J THEN FOR I=1TO 1500:NEXT
440 REM PRINT UP PERISCOPE
450 GOSUB2270:POKE SP+2,18:POKE SP+3,19:POKE V+28,8:POKE V+43,13:POKE V+9,75
460 POKE V+41,1:POKE V+6,172:POKE V+7,186:POKE V+4,172:POKE V,0:POKE V+27,13
470 IL=-1:SX=2:SH=0:IX=172:SN=4:PO=IX
480 POKE V+16,0:GOSUB2230:GOSUB490:POKE V+21,25:GOTO990
490 PRINT"{HOME}{CTRL RVSON}{CTRL GRN}{CTRL BLK}                   ";
500 PRINT"                {CTRL GRN}";
510 PRINT"     {CTRL BLK}{CMDR BLU}        ";
520 PRINT"                {CTRL BLK}{CTRL GRN}        ";
530 PRINT"     {CTRL BLK}{CMDR BLU}        ";
540 PRINT"                {CTRL BLK}{CTRL GRN}        ";
550 PRINT"     {CTRL BLK}{CMDR BLU}        ";
560 PRINT"                {CTRL BLK}{CTRL GRN}        ";
570 PRINT"     {CTRL BLK}{CTRL RVSOFF}{CTRL WHT}{CMDR A}{CTRL BLK}     ";
580 PRINT"                {CTRL RVSON}{CTRL GRN}        ";
590 PRINT"     {CTRL BLK}{CTRL RVSOFF}{CTRL WHT}{CMDR Q}{CTRL BLK}     ";
600 PRINT"                {CTRL RVSON}{CTRL GRN}        ";
```

Continued

## TORPEDO ALLEY *Continued*

```
610 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
620 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
630 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
640 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
650 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
660 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
670 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
680 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
690 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
700 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
710 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
720 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
730 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
740 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
750 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
760 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
770 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
780 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
790 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Q☚☛CTRL
     BLK☚ ";
800 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
810 PRINT"   ☛CTRL BLK☚ ☛CTRL RVSOFF☚
              ☛CTRL WHT☚☛CMDR Z☚☛CTRL
     BLK☚ ";
820 PRINT"                ☛CTRL RVSON☚ ☛
    CTRL GRN☚      ";
830 PRINT"   ☛CTRL BLK☚
    ";
840 PRINT"                 ☛CTRL GRN☚
    ";
850 PRINT"  ☛13 CMDR P☚ ☛CTRL BLK☚      ";
860 PRINT"   ☛CTRL GRN☚ ☛13 CMDR P☚      ";
870 PRINT"  ☛CMDR L☚☛CTRL WHT☚SCORE:
      ☛CTRL GRN☚☛CMDR J☚ ☛CTRL BLK☚
    ";
880 PRINT"  ☛CTRL GRN☚ ☛CMDR L☚☛CTRL W
    HT☚SUPPLIES:  ☛CTRL GRN☚☛CMDR J☚
    ";
890 PRINT"  ☛CMDR L☚☛CTRL WHT☚
      ☛CTRL GRN☚☛CMDR J☚  ☛CTRL BLK☚
    ";
900 PRINT" ☛CTRL GRN☚  ☛CMDR L☚☛CTRL W
    HT☚        ☛CTRL GRN☚☛CMDR J☚
    ";
910 PRINT"  ☛13 CMDR Y☚  ☛CTRL BLK☚   ";
920 PRINT" ☛CTRL GRN☚   ☛13 CMDR Y☚      ";
930 PRINT" ☛CMDR CYN☚                     ☛
    CTRL BLK☚ ";
940 PRINT" ☛CMDR CYN☚                     ☛
    CTRL GRN☚ ";
950 PRINT" ☛CMDR CYN☚                     ☛
    CTRL BLK☚ ";
960 PRINT" ☛CMDR CYN☚                     ☛
    CTRL GRN☚ ";
970 PRINT"                  ☛CTRL BLK☚
    ";
980 PRINT" ☛CTRL GRN☚
    ☛HOME☚☛CTRL WHT☚";:POKE 33767,160:P
    OKE 56295,5:RETURN
990 REM MAIN CONTROL LOOP
1000 IF NOT SH THEN GOSUB1040
1010 IF ((B<85 AND SX=-2) OR (B>260 AND
     SX=2)) AND (FR=0) THEN GOSUB1390
1020 X=3:Y=20:GOSUB2370:PRINTSC;:X=25:GO
     SUB2370:PRINTSU"☛SHIFT CRSRLEFT☚";
     :IF SU=0 THEN2290
1030 GOSUB1120:GOTO990
1040 REM PLACE SHIP IN PERISCOPE VIEW
1050 SH=-1:FR=0
1060 A=INT(RND(1)*93)+83:SY=INT(RND(1)*3
     )-1:I=INT(RND(1)*2):IFI=1 THEN SX=-
     1*SX
1070 B=25:IF SX=-2 THEN B=310
```

```
1080 REM MOVE SHIP
1090 IF A<82 THEN SY=1
1100 IF A>177 THEN SY=-1
1110 A=A+SY:B=B+SX*2+SD:PO=B:SN=0:GOSUB2
     230:POKE V+1,A:RETURN
1120 REM SCAN KEYBOARD
1130 IF J THEN1200
1140 K=PEEK(197):IF K=64 THEN SD=0:GOTO1
     340
1150 IF K=13 THEN SD=3:Y=5:GOTO1260
1160 IF K=18 THEN SD=-3:Y=-5:GOTO1260
1170 SD=0:IF K=60 THEN GOTO1470
1180 IF K=23 THEN1670
1190 GOTO1340
1200 REM SCAN JOYSTICK
1210 IF (PEEK(56320)AND16)/16=0 THEN GOT
     O1470
1220 JS=NOT PEEK(56320)AND15:IF JS=4 THE
     N SD=3:Y=5:GOTO1260
1230 IF JS=8 THEN SD=-3:Y=-5:GOTO1260
1240 IF JS=2 THEN1670
1250 SD=0:GOTO1340
1260 REM MOVE ISLAND AND BOAT
1270 IF SH THEN GOSUB1090
1280 IF NOT IL THEN DE=DE-Y
1290 IF DE<>105 THEN1310
1300 IX=278:PO=IX:SN=4:GOSUB2230:GOSUB22
     70:POKE V+21,PEEK(V+21)OR16:DE=0:IL
     =-1
1310 IF DE<>5 THEN1330
1320 IX=56:PO=IX:SN=4:GOSUB2230:GOSUB227
     0:POKE V+21,PEEK(V+21)OR16:DE=0:IL=
     -1
1330 IF IL THEN IX=IX+SD:SN=4:PO=IX:GOSU
     B2230
1340 IF (B>304 AND SX=2) OR (B<32 AND SX
     =-2) THEN SH=0
1350 IF SH THEN GOSUB1090
1360 IF IL AND IX>276 THEN DE=100:IL=0:P
     OKE V+21,PEEK(V+21)AND239:POKE V+8,
     0
1370 IF IL AND IX<59 THEN DE=10:IL=0:POK
     E V+21,PEEK(V+21)AND239:POKE V+8,0
1380 RETURN
1390 REM THE SHIP SHOOTS BACK
1400 FR=-1
1410 POKE V+5,A:SN=2:PO=B:GOSUB2230:POKE
      V+21,PEEK(V+21)OR4:FOR I=1 TO 200
1420 NEXT:POKE V+21,PEEK(V+21)AND251:POK
     E V3,S3:FOR I=1 TO 50:NEXT:POKE 532
     80,2
1430 FOR I=1 TO 10:NEXT:POKE 53265,PEEK(
     53265)AND239:FOR I=1 TO 30:NEXT
1440 POKE 53265,PEEK(53265)OR16:FOR I=1
     TO 10:NEXT:POKE 53280,5:POKE V3,S3-
     1
1450 SU=SU-INT(RND(1)*10+3):IF SU<0 THEN
      SU=0
1460 H=PEEK(V+30):RETURN
1470 REM FIRE TORPEDO AT SHIP
1480 IF NOT SH THEN RETURN
1490 H=PEEK(V+30):H=0:SU=SU-1:SD=0:POKE
     V1,S1:FOR I=1 TO 50:NEXT
1500 POKE V1,S1-1:GOSUB1090
1510 D=160:FOR TY=186 TO D STEP-3:GOSUB1
     550:NEXT:POKE V+38,6:IF H THEN1620
1520 D=140:FOR TY=TY TO D STEP-3:GOSUB15
     50:NEXT:POKE V+42,6:IF H THEN1620
1530 D=83:FOR TY=TY TO D STEP-3:GOSUB155
     0:NEXT:POKE V+7,186:IF H THEN1620
1540 POKE V+38,1:POKE V+42,1:RETURN
1550 REM CHECK FOR HIT - MOVE SHIP AND T
     ORPEDO
1560 H=PEEK(V+30)AND9:IF (H=9) AND (A-3<
     TY) THEN H=-1:TY=D:RETURN
1570 IF TY<A-5 THEN TY=D:RETURN
1580 IF ((B<85 AND SX=-2) OR (B>260 AND
     SX=2)) AND (FR=0) THEN GOSUB1390
1590 FOR I=1 TO 130:NEXT
1600 POKE V+7,TY:IF (B>304 AND SX=2) OR
     (B<32 AND SX=-2) THEN SH=0:RETURN
1610 GOSUB1090:RETURN
1620 REM SHIP IS HIT BY TORPEDO
1630 SH=0:TL=TL+1:POKE V+7,186:POKE V+38
     ,1:POKE V+42,1:POKE V3,S3
1640 FOR I=1 TO 80:POKE V+39,I:NEXT:POKE
      V+5,A:SN=2:PO=B:GOSUB2230
1650 POKE V3,S3-1:POKE V+21,PEEK(V+21)OR
     4:SN=0:PO=0:GOSUB2230:FOR I=1 TO80:
     NEXT
1660 POKE V+21,PEEK(V+21)AND219:SC=SC+AB
     S(245-A)-50:H=PEEK(V+30):RETURN
1670 REM DIVE
1680 FOR I=2 TO 7:DC(I,0)=0:NEXT
1690 H=PEEK(V+30):BN=2:NB=21:ND=NB:POKE
     V+21,0:POKE V+27,0
1700 POKE SP+2,20:POKE SP+3,20:POKE SP+4
     ,20:POKE V+43,1:POKE V+28,0:POKE V+
     16,0
1710 A=115:B=345:C=124:D=172:SN=0:PO=B:G
     OSUB2230:SN=1:PO=D:GOSUB2230
```

*Continued*

## TORPEDO ALLEY *Continued*

```
1720 POKE V+1,A:POKE V+3,C:POKE 53281,14
     :X=3:Y=19:GOSUB2370:PRINT"☛CTRL RVS
     ON☚☛CTRL WHT☚**DIVING**"
1730 PRINT TAB(3)"☛CTRL RVSON☚
      ":PRINT"☛HOME☚":X=25:GOSUB2370:PRI
     NT"☛CTRL RVSON☚**DIVING**"
1740 PRINT TAB(25)"☛CTRL RVSON☚
      ":X=18:Y=4:GOSUB2370
1750 PRINT" ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛
     CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN
     ☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT
     CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT
     ☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CRSRD
     OWN☚☛SHIFT CRSRLEFT☚ ☛CRSRDOWN☚☛SHI
     FT CRSRLEFT☚ ☛CRSRDOWN☚☛SHIFT CRSRL
     EFT☚ ☛CRSRDOWN☚☛SHIFT CRSRLEFT☚ ☛CR
     SRDOWN☚☛SHIFT CRSRLEFT☚ ":X=4
1760 FOR Y=16 TO 1 STEP-1:GOSUB2370:PRIN
     T "☛CTRL RVSOFF☚☛CTRL WHT☚££££££££
     ££££££££££££££££££££££"
1770 GOSUB2370
1780 FOR I=1 TO 100:NEXT:PRINT TAB(4)"☛C
     TRL RVSON☚☛CTRL BLU☚
                   ":NEXT
1790 FOR I=1 TO 500:NEXT:PRINT"☛SHIFT CL
     R☚☛CMDR BLU☚";:POKE 53281,6:FOR I=1
      TO 9
1800 PRINT"☛CTRL RVSON☚
                          ";:NEXT:POKE
      V+21,3
1810 REM MAIN CONTROL LOOP FOR SCREEN 2
1820 POKE V2-4,C:POKE V2,S2:POKE V2,S2-1
1830 K=NOT PEEK(56320)AND15:IF K=0 THEN1
     880
1840 IF K=1 THEN K=14:GOTO1890
1850 IF K=2 THEN K=23:GOTO1890
1860 IF K=4 THEN K=13:GOTO1890
1870 IF K=8 THEN K=18:GOTO1890
1880 K=PEEK(197):IF K=64 THEN1950
1890 IF K=14 AND C>122 THEN C=C-2:GOTO19
     40
1900 IF K=23 AND C<240 THEN C=C+2:GOTO19
     40
1910 IF K=13 AND D>24 THEN D=D-2:GOTO194
     0
1920 IF K=18 AND D<320 THEN D=D+2:GOTO19
     40
1930 GOTO1950
1940 SN=1:PO=D:GOSUB2230:POKE V+3,C
1950 B=B-5:SN=0:PO=B:GOSUB2230:IF B=0 TH
     EN B=345
1960 FOR I=2 TO 7:IF DC(I,0)=0 THEN1990
1970 DC(I,1)=DC(I,1)+3:POKE V+I*2+1,DC(I
     ,1):IF DC(I,1)=>C THEN GOSUB2070
1980 H=PEEK(V+30)AND2:IF H THEN2100
1990 NEXT:IF NB=0 THEN2130
2000 IF ND=0 THEN1810
2010 IF B<D-48 OR B>D+48 OR B<12 OR B>31
     0 THEN1810
2020 IF INT(RND(1)*3.5)<>0 THEN1810
2030 IF DC(BN,0) THEN1810
2040 DC(BN,0)=-1:DC(BN,1)=A+8:POKE V+BN*
     2+1,DC(BN,1):SN=BN:PO=B:GOSUB2230
2050 POKE V+21,PEEK(V+21)OR2↑BN:ND=ND-1:
     BN=BN+1:IF BN=8 THEN BN=2
2060 GOTO1810
2070 REM DEPTH CHARGE EXPLODES
2080 POKE V3,S3:POKE SP+I,18:DC(I,0)=0:N
     B=NB-1:FOR X=1 TO 50:NEXT:H=PEEK(V+
     30)
2090 POKE V+21,(PEEK(V+21))AND(255-2↑I):
     POKE V3,S3-1:POKE SP+I,20:RETURN
2100 REM HIT BY A DEPTH CHARGE
2110 POKE V3,S3:FOR I=1 TO 83:POKE V+40,
     I:NEXT:POKE SP+1,18:FOR I=1 TO 300:
     NEXT
2120 POKE SP+1,17:POKE V3,S3-1:GOTO2290
2130 REM SURFACING ROUTINE
2140 FOR I=C TO 124 STEP-1:POKE V2,S2:PO
     KE V2-4,I:POKE V+3,I:POKE V2,S2-1:N
     EXT
2150 POKE V+21,0:POKE 53281,14:POKE 5326
     5,PEEK(53265)AND239:GOSUB490
2160 X=4:FOR Y=1 TO 16:GOSUB2370:PRINT"☛
     CTRL RVSON☚☛CTRL BLU☚
                   ":NEXT
2170 X=3:Y=19:GOSUB2370
2180 PRINT"☛CTRL WHT☚☛CTRL RVSON☚*SURFAC
     ING*":X=25:GOSUB2370:PRINT"☛CTRL RV
     SON☚*SURFACING*"
2190 POKE 53265,PEEK(53265)OR16
2200 X=4:FOR Y=1 TO 16:GOSUB2370:PRINT"☛
     CTRL RVSOFF☚☛CTRL WHT☚££££££££££££
     ££££££££££££££££££"
2210 FOR I=1 TO 150:NEXT:GOSUB2370:PRINT
     "☛CTRL RVSON☚☛CMDR BLU☚
                   "
2220 NEXT:FORI=1 TO 500:NEXT:POKE 53281,
     6:GOTO440
2230 REM POKE LB/HB SPRITE POSITION
2240 IF PO>255 THEN PO=PO-256:POKE V+16,
     PEEK(V+16)OR2↑SN:GOTO2260
2250 POKE V+16,PEEK(V+16)AND(255-2↑SN)
2260 POKE V+SN*2,PO:RETURN
2270 REM CHOOSE ONE OF FOUR ISLANDS
2280 POKE SP+4,INT(RND(1)*3+21):RETURN
2290 REM END OF GAME
2300 POKE 198,0:POKE V+21,0:PRINT"☛CTRL
     WHT☚☛CTRL RVSOFF☚☛SHIFT CLR☚☛4CRSRD
     OWN☚YOU HAVE SUNK"TL"OF THE ENEMY S
     HIPS"
2310 PRINT"☛CRSRDOWN☚YOUR ACCURACY IS";:
     IF SU=30 THEN PRINT" 0%":GOTO2330
2320 PRINTINT((TL/(30-SU))*100+.5)"☛SHIF
     T CRSRLEFT☚%"
2330 PRINT"☛CRSRDOWN☚YOUR FINAL SCORE IS
     "SC:PRINT"☛2CRSRDOWN☚      DO YOU WANT
      TO PLAY AGAIN (Y/N)?"
2340 GET K$:IF K$<>"N" AND K$<>"Y" THEN2
     340
2350 IF K$="Y" THEN SU=30:SC=0:TL=SC:GOT
     O440
2360 PRINT"☛SHIFT CLR☚BYE...":END
2370 REM PLACE CURSOR AT X,Y
2380 POKE 781,Y:POKE 782,X:POKE 783,0:SY
     S 65520:RETURN
2390 REM SPRITE DATA
2400 DATA000,000,000,000,024,000,008,024
2410 DATA016,007,060,224,255,255,255,063
2420 DATA255,252,015,255,240,000,000,000
2430 DATA000,000,000,000,000,000,000,000
2440 DATA000,000,000,000,000,000,000,000
2450 DATA000,000,000,000,000,000,000,000
2460 DATA000,000,000,000,000,000,000,000
2470 DATA000,000,000,000,000,000,000,000
2480 DATA000,004,000,000,004,000,000,028
2490 DATA000,000,028,000,127,255,240,255
2500 DATA255,248,127,255,240,000,000,000
2510 DATA000,000,000,000,000,000,000,000
2520 DATA000,000,000,000,000,000,000,000
2530 DATA000,000,000,000,000,000,000,000
2540 DATA000,000,000,000,000,000,000,000
2550 DATA000,000,000,000,000,000,000,000
2560 DATA000,016,000,008,145,000,036,146
2570 DATA064,018,084,128,053,069,096,002
2580 DATA170,128,004,148,064,001,018,000
2590 DATA000,000,000,000,000,000,000,000
2600 DATA000,000,000,000,000,000,000,000
2610 DATA000,000,000,000,000,000,000,000
2620 DATA000,000,000,000,000,000,000,000
2630 DATA000,000,000,000,000,000,000,000
2640 DATA000,016,000,000,016,000,000,016
2650 DATA000,000,016,000,000,236,000,000
2660 DATA136,000,003,051,000,000,204,000
2670 DATA000,000,000,000,000,000,000,000
2680 DATA000,000,000,000,000,000,000,000
2690 DATA000,000,000,000,000,000,000,000
2700 DATA000,000,000,000,000,000,000,000
2710 DATA000,000,000,000,000,000,000,000
2720 DATA000,000,000,000,000,000,000,000
2730 DATA000,000,120,000,000,252,000,000
2740 DATA120,000,000,000,000,000,000,000
2750 DATA000,000,000,000,000,000,000,000
2760 DATA000,000,000,000,000,000,000,000
2770 DATA000,000,000,000,000,000,000,000
2780 DATA000,000,000,000,000,000,000,000
2790 DATA000,000,000,000,000,000,000,000
2800 DATA000,192,000,001,224,000,003,240
2810 DATA000,015,255,128,063,255,248,127
2820 DATA255,255,255,255,255,000,000,000
2830 DATA000,000,000,000,000,000,000,000
2840 DATA000,000,000,000,000,000,000,000
2850 DATA000,000,000,000,000,000,000,000
2860 DATA000,000,000,000,000,000,000,000
2870 DATA000,000,000,000,000,000,000,000
2880 DATA000,000,000,000,000,000,000,000
2890 DATA016,015,130,056,063,199,252,127
2900 DATA255,255,255,255,255,000,000,000
2910 DATA000,000,000,000,000,000,000,000
2920 DATA000,000,000,000,000,000,000,000
2930 DATA000,000,000,000,000,000,000,000
2940 DATA000,000,000,000,000,000,000,000
2950 DATA000,000,000,000,000,000,000,000
2960 DATA000,000,000,000,000,000,000,006
2970 DATA000,003,015,000,015,159,192,127
2980 DATA255,248,255,255,248,000,000,000
2990 DATA000,000,000,000,000,000,000,000
3000 DATA000,000,000,000,000,000,000,000
3010 DATA000,000,000,000,000,000,000,000
3020 DATA000,000,000,000,000,000,000,000
3030 DATA000,000,000,000,000,000,000,000
3040 DATA000,000,000,000,064,096,000,224
3050 DATA240,001,245,248,015,255,252,063
3060 DATA255,254,255,255,255,000,000,000
3070 DATA000,000,000,000,000,000,000,000
3080 DATA000,000,000,000,000,000,000,000
3090 DATA000,000,000,000,000,000,000,000
3100 DATA000,000,000,000,000,000,000,000
3110 DATA000,000,000,000,000,000,000,000
3120 REM SOUND DATA
3130 DATA163,002,096,064,128,003,138,143
3140 DATA012,224,112,016,003,250,099,056
3150 DATA160,144,128,010,170,000,064,001
3160 DATA063
```

HCM

# TORPEDO ALLEY

```
100 ' *****************
110 ' * TORPEDO ALLEY *
120 ' *****************
130 ' COPYRIGHT 1985
140 ' EMERALD VALLEY PUBLISHING CO.
150 ' BY WILLIAM K. BALTHROP
160 ' HOME COMPUTER MAGAZINE
170 ' VERSION 5.3.1
180 ' IBM PCjr WITH CARTRIDGE BASIC AND DOS 2.1 or
190 ' IBM PC WITH BASICA and
200 ' COLOR/GRAPHICS ADAPTER and
210 ' COLOR MONITOR
220 '
230 ' INITIALIZE & DISPLAY TITLE SCREEN
240 '
250 CLS:KEY OFF:SCREEN 0:LOCATE 12,13:PRINT "TORPEDO ALLEY":RTN$=CHR$(17)+CHR$(217):LOCATE 22,11:PRINT "PRESS ";RTN$;" TO START":GOSUB 1320:RANDOMIZE TIMER
260 DEFINT B-Z:SCREEN 1:CLS:LOCATE 12,1:PRINT "PRESS:":PRINT:PRINT "J FOR JOYSTICKS":PRINT:PRINT "K FOR KEYBOARD":GOSUB 1320:IF A$="K" OR A$="k" THEN JK=0 ELSE IF A$="J" OR A$="j" THEN JK=1:STRIG ON:JXC=STICK(0):JYC=STICK(1) ELSE 260
270 CLS:PRINT "INITIALIZING...":DIM T1(5),T2(4),T3(2),SHP1(16),SHP(16),I1(15),I2(15),I3(15),SUB(16),DC(6),EX(16),D(21,2):D(0,1)=400
280 DRAW "BM101,100C3D2GD2BR3U2HU2L":GET (100,100)-(103,105),T1:PUT (100,100),T1:DRAW "BM101,100DGNDEFND":GET (100,100)-(102,103),T2:PUT (100,100),T2:DRAW "BM101,100DU":GET (100,100)-(101,101),T3:PUT (100,100),T3
290 LINE(100,100)-(115,106),1,BF:S$="BM101,100C2F2NR7ENR5RERBR5G3NR4NL9DNR3NL8DNR2NL7":DRAW "XS$;":GET (100,100)-(115,105),SHP1:LINE (100,100)-(115,106),0,BF:DRAW "XS$;":GET (100,100)-(115,105),SHP:PUT (100,100),SHP
300 DRAW "BM100,104C2NR18RENR13RENR5RERD":GET (100,100)-(118,104),I1:PUT (100,100),I1
310 DRAW "BM100,104NR18ENR5ER2BR7R2FRNL6":GET (100,100)-(118,104),I2:PUT (100,100),I2:DRAW "BM100,104NR18ENR17ENR5RENR2ERBR10DNL1FNL3":GET (100,100)-(118,104),I3:PUT (100,100),I3
320 DRAW "BM104,100RFNL2RDNL6NR7DNL7NR8DNL6NR7DNL5NR5":GET (100,100)-(115,105),SUB:PUT (100,100),SUB:DRAW "BM101,100R5FDGL5HUEFD2":GET (100,100)-(107,103),DC:PUT (100,100),DC
330 DRAW "BM101,100F2G2NLRFBR2U2HUEBR3FGLGBR2FNGR2FBR2UEH2NUGL2":GET (100,100)-(115,105),EX:PUT (100,100),EX
340 '
350 ' SET UP FOR START OF A GAME
360 '
370 SCORE=0:SUPPLY=30:TOT=0:SH=0:TPO=0:DC=0:DONE=0:HIT=0:LDC=0:DEG=0:FOR Z=1 TO 21:D(Z,1)=0:D(Z,2)=0:NEXT:DEF SEG=0:POKE 1050,PEEK(1052)
380 GOSUB 420:GOTO 470
390 '
400 ' DRAW PERISCOPE
410 '
420 CLS:COLOR 1,1:LINE (50,5)-(269,120),3,B:DRAW "BM150,121C3D20L40G2D15F2R100E2U15H2L40U20L20BF5P1,3BM160,199P3,3":FOR Z=120 TO 200 STEP 3:LINE (Z,142)-(Z,159),2:NEXT
430 LINE (51,6)-(268,40),1,BF:LINE (159,6)-(159,119),3,BF:FOR Z=6 TO 114 STEP 8:LINE (157,Z)-(161,Z),3:LINE (155,Z+4)-(163,Z+4),3:NEXT:GOSUB 1330:RETURN
440 '
450 ' MAIN CONTROL LOOP FOR PERISCOPE
460 '
470 GOSUB 1240:GOSUB 680:IF HIT=1 THEN GOSUB 530
480 IF DONE=1 THEN 620 ELSE IF TPO=1 THEN GOSUB 1100
490 IF (SHX<145 AND MVSH=-2 AND INT(RND*30)=5) OR (SHX>175 AND MVSH=2 AND INT(RND*30)=5) THEN 570 ELSE 470
500 '
510 ' SHIP HIT BY TORPEDO
520 '
530 FOR Z=1 TO 40:PUT (SHX,SHY),EX:SOUND 110+RND*100,.5:NEXT:SCORE=SCORE+(160-SHY):TOT=TOT+1:SH=0:PUT (SHX,SHY),SHP:HIT=0:GOSUB 1330:RETURN
540 '
550 ' SHIP FIRES BACK AT YOU
560 '
570 FOR Z=1 TO 6:PUT (SHX,SHY),EX:SOUND 110+Z*100,.5:NEXT:FOR Z=1 TO 10:COLOR INT(RND*8),INT(RND*2):SOUND 400-Z*25,1:NEXT:COLOR 1,1:SUPPLY=SUPPLY-RND*9-1:IF SUPPLY<0 THEN SUPPLY=0
580 IF SUPPLY>0 THEN GOSUB 1330:GOTO 470
590 '
600 ' END OF GAME ROUTINE
610 '
620 CLS:LOCATE 8,1:PRINT "YOU HAVE SUNK";TOT;"SHIPS.":PRINT:PRINT "YOUR ACCURACY IS";:IF SUPPLY=30 THEN PRINT " 0%" ELSE PRINT TOT/(30-SUPPLY)*100;"%":PRINT:PRINT "YOUR FINAL SCORE IS";SCORE
630 LOCATE 16,1:PRINT "PLAY AGAIN (Y/N)?";:GOSUB 1320:IF A$<>"Y" AND A$<>"y" AND A$<>"N" AND A$<>"n" THEN 630 ELSE IF A$="Y" OR A$="y" THEN 370
640 CLS:END
650 '
660 ' SCAN KEYBOARD
670 '
680 CM=0:IF JK=0 THEN A$=INKEY$:DEF SEG=0:POKE 1050,PEEK(1052):IF A$="" THEN RETURN ELSE 710 ELSE JX=STICK(0):JY=STICK(1):JB=STRIG(0):IF JX<JXC-30 AND TPO=0 THEN A$=CHR$(75):GOSUB 1010:RETURN ELSE IF JX>JXC+30 AND TPO=0 THEN A$=CHR$(77):GOSUB 1010:RETURN
690 IF JY>JYC+30 AND TPO=0 THEN GOSUB 760:SH=0:DEG=0:DC=0:TPO=0:IF DONE=1 THEN RETURN ELSE GOSUB 420:RETURN
700 IF JB<>0 AND TPO=0 THEN GOSUB 1100 RETURN ELSE CM=0:RETURN
710 IF A$=CHR$(32) AND TPO=0 THEN GOSUB 1100 ELSE RETURN ELSE A$=RIGHT$(A$,1):IF (A$=CHR$(77) OR A$=CHR$(75)) AND TPO=0 THEN GOSUB 1010:RETURN
720 IF A$<>CHR$(13) OR TPO=1 THEN CM=0:RETURN ELSE GOSUB 760:SH=0:DEG=0:DC=0:TPO=0:IF DONE=1 THEN RETURN ELSE GOSUB 420:RETURN
730 '
740 ' DIVE ROUTINE AND SUBMERGED GAME ROUTINE
750 '
760 LINE (51,6)-(268,119),1,BF:PAINT (55,118),0,3:CLS:LINE (0,0)-(319,40),1,BF:FOR Z=40 TO 101 STEP 2:PUT (155,Z),SUB:FOR TD=1 TO 300:NEXT:PUT (155,Z),SUB:NEXT:PUT (155,101),SUB
770 SHX=300:ASHX=300:SHY=35:OSBX=155:SBX=155:OSBY=101:SBY=101
780 GOSUB 790:GOSUB 870:GOSUB 890:IF DONE=1 THEN RETURN ELSE IF LDC=1 THEN 970 ELSE 780
790 IF JK=1 THEN 800 ELSE A$=INKEY$:DEF SEG=0:POKE 1050,PEEK(1052):IF A$="" THEN RETURN ELSE A$=RIGHT$(A$,1):IF A$=CHR$(75) THEN 830 ELSE IF A$=CHR$(77) THEN 840 ELSE IF A$=CHR$(72) THEN 850 ELSE IF A$=CHR$(80) THEN 860 ELSE RETURN
800 JX=STICK(0):JY=STICK(1):IF JX<JXC-30 THEN GOSUB 830 ELSE IF JX>JXC+30 THEN GOSUB 840
810 IF JY<JYC-30 THEN GOSUB 850 ELSE IF JY>JYC+30 THEN GOSUB 860
820 RETURN
830 IF SBX=40 THEN RETURN ELSE SBX=SBX-1:PUT (OSBX,SBY),SUB:PUT (SBX,SBY),SUB:OSBX=SBX:RETURN
840 IF SBX=264 THEN RETURN ELSE SBX=SBX+1:PUT (OSBX,SBY),SUB:PUT (SBX,SBY),SUB:OSBX=SBX:RETURN
850 IF SBY=40 THEN RETURN ELSE SBY=SBY-1:PUT (SBX,OSBY),SUB:PUT (SBX,SBY),SUB:OSBY=SBY:RETURN
860 IF SBY=190 THEN RETURN ELSE SBY=SBY+1:PUT (SBX,OSBY),SUB:PUT (SBX,SBY),SUB:OSBY=SBY:RETURN
870 ASHX=ASHX-2:SHX=ASHX:IF SHX<7 THEN LD=400:ASHX=300:LINE (0,35)-(30,40),1,BF:RETURN
880 PUT (SHX,SHY),SHP1,PSET:RETURN
890 IF DC=0 THEN 910 ELSE FOR Z=1 TO DC:IF D(Z,1)=0 THEN 900 ELSE PUT (D(Z,1),D(Z,2)),DC:D(Z,2)=D(Z,2)+2:PUT (D(Z,1),D(Z,2)),DC:IF D(Z,2)>=SBY THEN GOSUB 930:IF DONE=1 THEN RETURN
900 NEXT
910 IF SHX-50>SBX OR SHX+50<SBX OR SHX<40 OR SHX>264 OR INT(RND*7)<>2 THEN RETURN ELSE IF DC>0 THEN IF SHX+10>LD THEN RETURN
920 IF DC<21 THEN LD=SHX:DC=DC+1:D(DC,1)=SHX:D(DC,2)=40:PUT (D(DC,1),D(DC,2)),DC:SOUND 440,3:RETURN ELSE RETURN
930 PUT (D(Z,1),D(Z,2)),DC:FOR W=1 TO 9:SOUND 110+RND*200,1:CIRCLE (D(Z,1)+3,D(Z,2)+1),W,2:NEXT
940 FOR W=1 TO 9:CIRCLE (D(Z,1)+3,D(Z,2)+1),W,0:NEXT
950 IF ABS((D(Z,1)-4)-SBX)<18 THEN DONE=1:FOR Z=1 TO 20:PUT (SBX,SBY),EX:SOUND 110+RND*600,1:FOR TD=1 TO 20:NEXT:NEXT
```

***Continued***

## TORPEDO ALLEY *Continued*

IBM PC & IBM PCjr

```
 960  D(Z,1)=0:D(Z,2)=0:IF Z=21 THEN LDC=
      1:RETURN ELSE LDC=0:RETURN
 970  PUT (SBX,SBY),SUB:FOR SBY=SBY TO 40
       STEP -2:PUT (SBX,SBY),SUB:FOR TD=1
       TO 300:NEXT:PUT (SBX,SBY),SUB:NEXT
      :LDC=0:CLS:RETURN
 980  '
 990  ' MOVE ISLANDS & ADJUST SHIPS SPEED
1000  '
1010  IF A$=CHR$(75) THEN DEG=DEG+1:CM=.7
      5 ELSE DEG=DEG-1:CM=-.75
1020  IF DEG=360 THEN DEG=0 ELSE IF DEG<0
       THEN DEG=359
1030  IN=INT(DEG/120)+1:ON IN GOSUB 1040,
      1050,1060:RETURN
1040  IF DEG=0 THEN RETURN ELSE IP=DEG*1.
      65:IF DEG=1 OR DEG=119 THEN OIP=IP:
      LINE(51,36)-(76,40),1,BF:LINE(240,3
      6)-(268,40),1,BF:PUT(51+IP,36),I1:R
      ETURN ELSE PUT (51+OIP,36),I1:PUT (
      51+IP,36),I1:OIP=IP:RETURN
1050  IP=(DEG-120)*1.65:IF DEG=120 OR DEG
      =239 THEN OIP=IP:LINE(51,36)-(76,40
      ),1,BF:LINE(240,36)-(268,40),1,BF:P
      UT(51+IP,36),I2:RETURN ELSE PUT (51
      +OIP,36),I2:PUT (51+IP,36),I2:OIP=I
      P:RETURN
1060  IP=(DEG-240)*1.65:IF DEG=240 OR DEG
      =359 THEN OIP=IP:LINE(51,36)-(76,40
      ),1,BF:LINE(240,36)-(268,40),1,BF:P
      UT(51+IP,36),I3:RETURN ELSE PUT (51
      +OIP,36),I3:PUT (51+IP,36),I3:OIP=I
      P:RETURN
1070  '
1080  ' FIRE AND MOVE TORPEDO
1090  '
1100  IF TPO=0 THEN TPO=1:SUPPLY=SUPPLY-1
      :GOSUB 1330:FOR Z=1 TO 10:SOUND 400
      -(Z*25),.5:NEXT:TP=113:OTP=113:TPS=
      1:PUT (168,TP),T1:RETURN
1110  IF TP<SHY+4 THEN TPO=0:ON TPS GOSUB
       1160,1180,1200:TPS=1:IF SUPPLY=0 T
      HEN DONE=1:RETURN ELSE RETURN
1120  IF SHX>156 AND SHX<168 AND TP<SHY+7
       THEN HIT=1:ON TPS GOSUB 1160,1180,
      1200:TPO=0:IF SUPPLY=0 THEN DONE=1:
      RETURN ELSE RETURN
1130  IF TP=95 THEN GOSUB 1180:TPS=3:GOSU
      B 1200 ELSE IF TP=105 THEN GOSUB 11
      60:TPS=2:GOSUB 1180
1140  TP=TP-2:ON TPS GOSUB 1150,1170,1190
      :OTP=TP:RETURN
1150  PUT (168,OTP),T1
1160  PUT (168,TP),T1:RETURN
1170  PUT (168,OTP),T2
1180  PUT (168,TP),T2:RETURN
1190  PUT (168,OTP),T3
1200  PUT (168,TP),T3:RETURN
1210  '
1220  ' MOVE ENEMY SHIP THROUGH PERISCOPE
1230  '
1240  IF SH=1 THEN 1260
1250  SH=1:YP=RND*40+45:MVSH=INT(RND*2):M
      VSH=(MVSH+(SGN(MVSH-1)))*2:MVY=INT(
      RND*3)-1:IF MVSH=2 THEN ASHY=YP:SHX
      =53:OSHX=53:ASHX=53:SHY=YP::OSHY=YP
      :PUT (SHX,SHY),SHP:RETURN ELSE SHX=
      252:OSHX=252:ASHX=252:ASHY=YP:SHY=Y
      P:OSHY=YP:PUT (SHX,SHY),SHP:RETURN
1260  IF SHX+MVSH<52 OR SHX+MVSH>253 THEN
       SH=0:PUT (SHX,SHY),SHP:RETURN
1270  IF SHY+MVY>90 THEN MVY=-1 ELSE IF S
      HY+MVY<42 THEN MVY=1
1280  ASHY=ASHY+MVY*.5:SHY=ASHY:ASHX=ASHX
      +MVSH+CM:SHX=ASHX:PUT (OSHX,OSHY),S
      HP:PUT (SHX,SHY),SHP:OSHX=SHX:OSHY=
      SHY:RETURN
1290  '
1300  ' MISC. ROUTINES
1310  '
1320  A$=INKEY$:IF A$="" THEN 1320 ELSE R
      ETURN
1330  LOCATE 22,3:PRINT " SCORE:";SCORE:L
      OCATE 22,26:PRINT " SUPPLY:";SUPPLY
      ;:RETURN
```

HCM

## TORPEDO ALLEY

TI-99/4A

```
100  ! ******************
110  ! * TORPEDO ALLEY *
120  ! ******************
130  ! COPYRIGHT 1985
140  ! EMERALD VALLEY PUBLISHING CO.
150  ! BY WILLIAM K. BALTHROP
160  ! HOME COMPUTER MAGAZINE
170  ! VERSION 5.3.1
180  ! TI EXTENDED BASIC
190  !
200  DISPLAY AT(12,7)ERASE ALL:"TORPEDO
     ALLEY" :: DISPLAY AT(22,4):"PRESS E
     NTER TO START"
210  CALL KEY(0,K,S):: IF S=0 THEN 210 E
     LSE CALL CLEAR
220  CALL SCREEN(15):: DISPLAY AT(5,1):"
     JOYSTICKS Y or N?" :: ACCEPT AT(5
     ,20)VALIDATE("YNyn")SIZE(1):J$
230  IF J$="Y" OR J$="y" THEN J=1 ELSE I
     F J$="N" OR J$="n" THEN J=0 ELSE GO
     TO 220
240  IF J=0 THEN DISPLAY AT(20,1)BEEP:"P
     USH ALPHA LOCK DOWN" :: GOTO 260
250  DISPLAY AT(20,1)BEEP:"PLACE ALPHA L
     OCK UP"
260  FOR TD=1 TO 500 :: NEXT TD
270  RANDOMIZE :: CALL CLEAR :: CALL SCR
     EEN(13):: CALL COLOR(1,2,4):: CALL
     MAGNIFY(3):: SP=1
280  CALL CHARPAT(110,CN$,121,CY$)
290  CALL CHAR(121,"00",122,"00",123,"00
     "):: SUP=30 :: SH,SC,DC,TOT=0 :: DB
     =1
300  FOR X=2 TO 8 :: CALL COLOR(X,2,4)::
      NEXT X
310  RESTORE 1310 :: FOR X=1 TO 31 :: RE
     AD A,A$ :: CALL CHAR(A,A$):: NEXT X
320  RESTORE 1410 :: FOR X=1 TO 4 :: REA
     D A,B,C :: CALL COLOR(A,B,C):: NEXT
      X
330  RESTORE 1390 :: M=1 :: GOSUB 340 ::
      GOTO 360
340  FOR X=M TO 16 :: READ A,B,C,D :: IF
      X=3 THEN CALL VCHAR(1,27,35,288)::
      CALL HCHAR(18,1,32,224)
350  CALL HCHAR(A,B,C,D):: NEXT X :: RET
     URN
360  I=1
370  DISPLAY AT(20,1)SIZE(6):"SCORE:" ::
      DISPLAY AT(20,22)SIZE(7):"SUPPLY:"
380  FOR X=2 TO 9 :: CALL SPRITE(#X,96,1
     6,(X-2)*16+1,117):: NEXT X :: CALL
     SPRITE(#1,96,13,249,117)
390  IF SH=0 THEN GOSUB 420
400  CALL POSITION(#21,A,B):: IF (B<110
     AND B>48 AND SP=-1)OR(B>145 AND B<2
     00 AND SP=1)THEN GOSUB 1220
410  DISPLAY AT(21,1)SIZE(6):SC :: DISPL
     AY AT(21,22)SIZE(7):SUP :: IF SUP=0
      THEN 740 ELSE GOSUB 450 :: GOTO 39
     0
420  A=INT(RND*40)+33 :: SY=INT(RND*3)-1
      :: S=INT(RND*2):: IF S=1 THEN SP=S
     GN(-SP)
430  B=255+SP*40 :: IF B>255 THEN B=B-25
     5
440  CALL SPRITE(#21,134+2*SP,13,A,B,SY,
     SP*2):: SH=1 :: RETURN
450  IF J=1 THEN 480 ELSE CALL KEY(0,K,S
     ):: IF S=0 THEN SPD=0 :: GOTO 540
460  IF K=83 THEN SPD=1 :: Y=5 :: GOTO 5
     00 ELSE IF K=68 THEN SPD=-1 :: Y=-5
      :: GOTO 500 ELSE SPD=0 :: IF K=32
     THEN GOTO 620
470  IF K=13 THEN GOSUB 870 :: GOTO 470
     ELSE GOTO 550
480  CALL JOYST(1,X,Y):: CALL KEY(1,K,S)
     :: IF X=-4 THEN SPD=1 :: GOTO 500 E
     LSE IF X=4 THEN SPD=-1 :: GOTO 500
     ELSE SPD=0
490  IF Y=-4 THEN GOSUB 870 :: GOTO 470
     ELSE IF K=18 THEN GOTO 620 ELSE GOT
     O 540
500  IF SH=1 THEN CALL MOTION(#21,SY,SP*
     2+SPD)
510  IF I=0 THEN DEG=DEG-Y
520  IF DEG=105 THEN GOSUB 810 :: CALL S
     PRITE(#11,104,13,17,215):: DEG=0 ::
      I=1
530  IF DEG=5 THEN GOSUB 810 :: CALL SPR
     ITE(#11,104,13,17,25):: DEG=0 :: I=
     1
540  IF I=1 THEN CALL MOTION(#11,0,SPD)
550  CALL POSITION(#21,X,Y):: IF Y<24 OR
      Y>216 THEN CALL DELSPRITE(#21):: S
     H=0
560  IF X<29 THEN SY=1 ELSE IF X>66 THEN
      SY=-1
570  CALL MOTION(#21,SY,SP*2+SPD)
580  IF I=1 THEN CALL POSITION(#11,A,M):
     : IF M<24 AND M<>0 THEN DEG=10 :: C
     ALL DELSPRITE(#11):: I=0
590  IF I=1 AND M>216 THEN DEG=100 :: CA
     LL DELSPRITE(#11):: I=0
600  RETURN
610  !
620  IF SH=1 THEN CALL MOTION(#21,SY,SP*
     2)ELSE RETURN
```

Continued

## TORPEDO ALLEY *Continued*

```
 630 FOR X=1 TO 30 STEP 6 :: CALL SOUND(
     -600,-6,X):: CALL CK(SP):: NEXT X
 640 CALL SPRITE(#25,120,1,144,129,-6,0)
     :: RESTORE 1300
 650 CALL POSITION(#25,X,Y):: IF X<130 T
     HEN CALL COLOR(#25,16)ELSE GOTO 650
 660 FOR X=1 TO 6 :: READ A$ :: CALL CHA
     R(120,A$):: CALL MOTION(#25,X-9,0):
     : NEXT X :: SUP=SUP-1
 670 CALL CK(SP):: CALL POSITION(#25,X,Y
     ,#21,A,B):: IF X<A+10 THEN CALL DEL
     SPRITE(#25):: CALL CHAR(120,"00")::
      RETURN
 680 IF A<29 OR A>72 THEN CALL MOTION(#2
     1,0,SP*2)
 690 IF X<A+16 THEN CALL POSITION(#21,B,
     A):: IF A>119 AND A<134 THEN GOTO 7
     10 ELSE GOTO 670
 700 GOTO 670
 710 CALL DELSPRITE(#25):: CALL SOUND(-1
     00,110,10,220,10,7000,30,-8,0):: CA
     LL EX(2):: CALL COLOR(#21,7):: CALL
      EX(2)
 720 CALL CHAR(120,"00"):: CALL PATTERN(
     #21,140):: CALL MOTION(#21,0,0):: C
     ALL EX(10)
 730 CALL DELSPRITE(#21):: SH=0 :: TOT=T
     OT+1 :: SC=SC+80-B :: RETURN
 740 CALL CLEAR :: CALL SCREEN(4):: CALL
      DELSPRITE(ALL)
 750 DISPLAY AT(5,1):"YOU HAVE SUNK";TOT
     ;" OF THE":"ENEMY SHIPS."
 760 CALL CHARSET :: DISPLAY AT(8,1):"YO
     UR ACCURACY IS"
 770 IF SUP=30 THEN DISPLAY AT(8,17):" 0
     %" ELSE DISPLAY AT(8,17):INT((TOT/(
     30-SUP))*100+.5);"%"
 780 DISPLAY AT(12,1):"YOUR FINAL SCORE
     IS";SC;"." :: CALL COLOR(10,2,1,12,
     2,1):: CALL CHAR(121,CY$,110,CN$)
 790 DISPLAY AT(24,1):"PLAY AGAIN (Y/N)?
     " :: ACCEPT AT(24,19)VALIDATE("YNyn
     ")SIZE(1):A$ :: IF A$="Y" OR A$="y"
      THEN GOTO 270
 800 CALL CLEAR :: STOP
 810 C=INT(RND*4)+1 :: ON C GOSUB 830,84
     0,850,860
 820 READ A$ :: CALL CHAR(104,A$):: RETU
     RN
 830 RESTORE 1460 :: RETURN
 840 RESTORE 1470 :: RETURN
 850 RESTORE 1480 :: RETURN
 860 RESTORE 1490 :: RETURN
 870 CALL DELSPRITE(#25,#21,#11):: FOR X
     =5 TO 15 :: CALL HCHAR(X,7,96,20)::
      NEXT X
 880 FOR X=16 TO 3 STEP -1 :: CALL HCHAR
     (X,7,104,20):: CALL HCHAR(X-1,7,100
     ,20):: FOR D=1 TO 40 :: NEXT D
 890 NEXT X :: CALL HCHAR(2,7,104,20)
 900 CALL DELSPRITE(ALL):: CALL CLEAR ::
      CALL SCREEN(8):: CALL HCHAR(1,1,96
     ,96):: CALL HCHAR(4,1,104,672)
 910 CALL SPRITE(#27,128,15,25,128,1,0):
     : FOR D=1 TO 1000 :: NEXT D :: CALL
      SPRITE(#28,132,13,9,240,0,-2)
 920 CALL MOTION(#27,0,0)
 930 CALL DISTANCE(#27,#28,D):: CALL SON
     (SQR(D)/15)
 940 IF J=0 THEN GOTO 970 ELSE CALL JOYS
     T(1,X,Y):: IF X=-4 THEN K=83 ELSE I
     F X=4 THEN K=68 ELSE IF Y=-4 THEN K
     =88
 950 IF Y=4 THEN K=69 ELSE IF X=0 AND Y=
     0 THEN GOTO 1030
 960 GOTO 980
 970 CALL KEY(0,K,S):: IF S=0 THEN 1030
 980 CALL POSITION(#27,X,Y):: IF (K=69)*
     (X<20)OR(K=88)*(X>167)OR(K=83)*(Y<3
     0)OR(K=68)*(Y>225)THEN 1030
 990 IF K=69 THEN CALL MOTION(#27,-1,0):
     : GOTO 1040
1000 IF K=83 THEN CALL MOTION(#27,0,-1):
     : GOTO 1040
1010 IF K=68 THEN CALL MOTION(#27,0,1)::
      GOTO 1040
1020 IF K=88 THEN CALL MOTION(#27,1,0)::
      GOTO 1040
1030 CALL MOTION(#27,0,0)
1040 FOR X=DB TO DC :: CALL POSITION(#X,
     A,B,#27,C,D)
1050 IF A<C-8 THEN 1100
1060 CALL COINC(#X,#27,10,B):: CALL PATT
     ERN(#X,140):: CALL COLOR(#X,7):: CA
     LL SOUND(500,110,7,220,7,1000,30,-8
     ,0):: CALL DELSPRITE(#X)
1070 IF B<>-1 THEN GOTO 1090
1080 CALL COLOR(#27,7):: CALL PATTERN(#2
     7,140):: FOR A=1 TO 5 :: CALL SOUND
     (200,110,30,220,10,RND*2000+110,30,
     -8,0):: NEXT A
1090 IF B THEN GOTO 740 ELSE DB=DB+1 ::
      IF DB=21 THEN GOTO 1150
1100 NEXT X
1110 IF DC=20 THEN GOTO 930
1120 CALL POSITION(#27,A,B,#28,C,D):: IF
      D<B-48 OR D>B+48 THEN 930
1130 IF INT(RND*1.5)=0 THEN DC=DC+1 :: C
     ALL POSITION(#28,X,Y):: CALL SPRITE
     (#DC,112,4,X,Y,3,0):: GOTO 930
1140 GOTO 930
1150 CALL DELSPRITE(#28):: CALL MOTION(#
     27,-2,0)
1160 CALL POSITION(#27,X,Y):: IF X>17 TH
     EN 1160
1170 CALL DELSPRITE(ALL):: CALL CLEAR ::
      CALL SCREEN(13):: M=3 :: RESTORE 1
     390 :: READ A,A,A,A,A,A,A,A :: GOSU
     B 340
1180 FOR X=2 TO 16 :: CALL HCHAR(X,7,104
     ,20):: NEXT X
1190 FOR X=2 TO 14 :: CALL HCHAR(X,7,96,
     20):: CALL HCHAR(X+1,7,100,20):: CA
     LL HCHAR(X+2,7,104,20)
1200 FOR Y=1 TO 50 :: NEXT Y :: NEXT X :
     : DB=1 :: DC=0
1210 FOR X=5 TO 16 :: CALL HCHAR(X,7,104
     ,20):: NEXT X :: FOR X=2 TO 4 :: CA
     LL HCHAR(X,7,96,20):: NEXT X :: GOT
     O 360
1220 IF INT(RND*7)<>1 THEN RETURN
1230 CALL POSITION(#21,A,B):: CALL SPRIT
     E(#28,140,7,A,B,SY,SP*3+SPD):: CALL
      SOUND(300,110,3,220,3,600,30,-8,0)
     :: CALL CK(SP)
1240 CALL SOUND(1,110,30):: CALL DELSPRI
     TE(#28):: FOR D=1 TO 20 :: CALL CK(
     SP):: NEXT D
1250 CALL MOTION(#21,0,0,#28,0,0,#11,0,0
     )
1260 FOR X=1 TO 5 :: CALL SCREEN(7):: CA
     LL SOUND(-990,110,5,220,4,RND*1000+
     500,30,-8,0):: CALL SCREEN(13)
1270 CALL CK(SP):: CALL SOUND(-990,110,5
     ,220,4,RND*500+1000,30,-8,0):: CALL
      SCREEN(7):: CALL SCREEN(13):: NEXT
      X
1280 SUP=SUP-INT(RND*10+3):: IF SUP<0 TH
     EN SUP=0
1290 RETURN
1300 DATA 081C2A142A552A55,08081408142A1
     42A,080808140814082A,080814001414,0
     8081414,080808
1310 DATA 132,00,133,000845231BFF7F3F00,
     135,0000C1C2ECFFFFE00,137,00008343
     37FFFF7F00,139,0010A2C4D8FFFEFC
1320 DATA 140,00,141,1295492A22D4191800,
     143,1295492A22D41918
1330 DATA 35,FFFFFFFFFFFFFFFF808080C0808
     080F0FF7F3F1F0F070301FFFEFCF8F0E0C0
     80
1340 DATA 96,00,97,00,98,808080C0808080F
     0808080C0808080F0,104,00
1350 DATA 105,00010303071F3FFF,106,00,10
     7,80C0C0C0F8FCFEFF
1360 DATA 108,00,109,0000000000033FFF,11
     0,00,111,00000000C0F0FCFF,100,24029
     1AA556DDBFF
1370 DATA 113,00000103071F3FFF000080C0F0
     FFFFFF000000000080F8FF,120,00
1380 DATA 112,00,113,0000070F0F07,114,00
     ,115,0000E0F0F0E0,128,00,129,000007
     077FFF7F,130,00,131,00000080FEFFFE
1390 DATA 2,1,96,92,5,1,104,384,1,1,35,3
     2,17,1,35,32,18,12,37,1,18,13,35,8,
     18,21,38,1,19,13,37,1
1400 DATA 19,14,35,6,19,20,38,1,20,14,37
     ,1,20,15,35,4,20,19,38,1,21,16,35,2
     ,22,10,35,14,23,10,35,14
1410 DATA 1,13,4,9,16,8,10,16,5,11,13,8
1420 DATA 00000000000000000000000000033F
     FF000000000000000000000000C0F0FCFF
1430 DATA 000000000000000000010303071F3F
     FF000000000000000080C0C0C0F8FCFEFF
1440 DATA 00000000000000000000000001337F
     FF00000000000000000000008080CCFEFF
1450 DATA 00000000000000000000001C3E7FFF
     FF00000000000000000000001038FCFCFF
1460 DATA 00000000000000000000000000033F
     FF000000000000000000000000C0F0FCFF
1470 DATA 000000000000000000010303071F3F
     FF000000000000000080C0C0C0F8FCFEFF
1480 DATA 00000000000000000000000001337F
     FF00000000000000000000008080CCFEFF
1490 DATA 00000000000000000000001C3E7FFF
     FF00000000000000000000001038FCFCFF
1500 SUB EX(X)
1510 FOR Z=1 TO X :: CALL SOUND(-1000,11
     0,10,220,10,RND*5000+750,30,-8,0)::
      NEXT Z
1520 SUBEND
1530 SUB CK(SP)
1540 CALL POSITION(#21,X,Y):: IF X<29 TH
     EN SY=1 :: CALL MOTION(#21,SY,SP*2)
1550 IF X>72 THEN SY=-1 :: CALL MOTION(#
     21,SY,SP*2)
1560 SUBEND
1570 SUB SON(X)
1580 CALL SOUND(100,440,X):: FOR Y=1 TO
     3 :: CALL SOUND(1,440,14+X):: NEXT
     Y :: SUBEND
```

HCM

# GEOMETRIX

```
100 REM  *************
110 REM  * GEOMETRIX *
120 REM  *************
130 REM  COPYRIGHT 1985
140 REM  EMERALD VALLEY PUBLISHING
150 REM  BY WAYNE KOBERSTEIN
160 REM  AND THE HCM STAFF
170 REM  HOME COMPUTER MAGAZINE
180 REM  VERSION 5.3.1
190 REM  APPLE II FAMILY APPLESOFT
200 REM
210 IF PEEK (103) = 1 AND PEEK (104) = 64 THEN 240
220 POKE 103,1: POKE 104,64: POKE 16384,0: POKE 16385,0: POKE 16386,0
230 PRINT CHR$ (4);"RUN GEOMETRY"
240 TEXT : HOME : VTAB 12: PRINT TAB( 7)"GEOMETRIX": VTAB 24: PRINT "INITIALIZING...";
250 FOR I = 2048 TO 2161: READ V: POKE I,V: NEXT : POKE 232,0: POKE 233,8
260 READ R$,W$,I$
270 HOME : VTAB 5: PRINT TAB( 9)"GEOMETRIX": PRINT : PRINT : PRINT "ENTER YOUR CHOICE:": PRINT : PRINT "1) RECTANGLES": PRINT "2) RIGHT TRIANGLES": PRINT "3) PARALLELOGRAMS": PRINT "4) TRAPEZOIDS": PRINT "5) EXIT"
280 GOSUB 2000: IF KB < 177 OR KB > 181 THEN 280
290 IF KB < 181 THEN 310
300 HOME : END
310 SH = KB - 176: HOME : PRINT "SELECT A LEVEL:": PRINT : PRINT : PRINT "1) EASY": PRINT "2) AVERAGE": PRINT "3) HARD"
320 GOSUB 2000: IF KB < 177 OR KB > 179 THEN 320
330 LE = KB - 176: ON SH GOSUB 350,730,1040,1370
340 GOTO 270
350 HOME : ON LE GOTO 360,360,510
360 V(1) = INT (10 * RND (1)) + 1:V(2) = V(1):T = 0: IF LE = 2 THEN V(2) = INT (V(1) * RND (1)) + 1
370 GOSUB 1710: VTAB 21: PRINT "AB=";V(1);" AC=";V(2)
380 GOSUB 2020: IF A > 0 THEN 400
390 GOTO 420
400 AN = V(1) * V(2): IF A = AN THEN 440
410 PRINT W$: GOSUB 2100:T = T + 1: IF T = 3 THEN 430
420 HOME : GOTO 370
430 HOME : PRINT I$;" ";AN: GOTO 450
440 PRINT R$
450 GOSUB 1970: HOME : ON KB GOTO 360,470,460
460 TEXT : RETURN
470 HOME :T = 2: INPUT "AB=";A$: GOSUB 2040: IF A = 0 THEN 470
480 V(1) = A
490 INPUT "AC=";A$: GOSUB 2040: IF A = 0 THEN 490
500 V(2) = A: GOTO 370
510 V(1) = INT (20 * RND (1)) + 5:V(2) = V(1) - ABS ( INT ((V(1) - 3) * RND (1)) + 1)
520 V(3) = INT (20 * RND (1)) + 5:V(4) = V(3) - INT ((V(3) - 3) * RND (1)) + 1: IF V(4) > V(3) THEN 520
530 T = 0
540 GOSUB 1710: VTAB 21: PRINT "AC=";V(1);" DE=";V(2);" AF=";V(3);" DG=";V(4)
550 GOSUB 2020: IF A = 0 THEN 580
560 AN = (V(3) * (V(1) - V(2))) + (V(2) * (V(3) - V(4))): IF A = AN THEN 600
570 PRINT W$: GOSUB 2100:T = T + 1: IF T = 3 THEN 590
580 HOME : GOTO 540
590 PRINT I$;" ";AN: GOTO 610
600 PRINT R$
610 GOSUB 1970: HOME : ON KB GOTO 510,630,620
620 TEXT : RETURN
630 HOME :T = 2
640 INPUT "AC=";A$: GOSUB 2040: IF A = 0 THEN 640
650 V(1) = A
660 INPUT "DE=";A$: GOSUB 2040: IF A = 0 THEN 660
670 V(2) = A: IF V(2) > V(1) THEN 660
680 INPUT "AF=";A$: GOSUB 2040: IF A = 0 THEN 680
690 V(3) = A
700 INPUT "DG=";A$: GOSUB 2040: IF A = 0 THEN 700
710 V(4) = A: IF V(4) > V(3) THEN 700
720 HOME : GOTO 540
730 HOME : ON LE GOTO 740,740,890
740 T = 0:V(1) = INT (10 * RND (1)) + 1:V(2) = INT (10 * RND (1)) + 1
750 GOSUB 1710: VTAB 21: PRINT "AB=";V(1);" BC=";V(2): IF LE = 2 THEN 770
760 GOSUB 2020:AN = V(1) * V(2) / 2: GOTO 780
770 SG$ = "AC": GOSUB 2030:AN = INT (( SQR ((V(1) * V(1)) + (V(2) * V(2)))) * 100 + .50001) / 100
780 IF A = AN THEN 820
790 PRINT W$;: GOSUB 2100:T = T + 1: IF T = 3 THEN 810
800 HOME : GOTO 750
810 HOME : PRINT I$;" ";AN: GOTO 830
820 PRINT R$
830 GOSUB 1970: HOME : ON KB GOTO 740,850,840
840 TEXT : RETURN
850 INPUT "AB=";A$: GOSUB 2040: IF A = 0 THEN 850
860 V(1) = A
870 INPUT "BC=";A$: GOSUB 2040: IF A = 0 THEN 870
880 V(2) = A:T = 2: HOME : GOTO 750
890 T = 0:V(1) = INT (10 * RND (1)) + 1:V(2) = INT (10 * RND (1)) + 1:V(3) = V(1) * 2
900 GOSUB 1710: VTAB 21: PRINT "AB=";V(1);" BD=";V(2);" AC=";V(3):SG$ = "DC": GOSUB 2030:AN = INT (( SQR ((V(1) * V(1)) + (V(2) * V(2)))) * 100 + .50001) / 100: IF A = AN THEN 940
910 PRINT W$;: GOSUB 2100:T = T + 1: IF T = 3 THEN 930
920 HOME : GOTO 900
930 HOME : PRINT I$;" ";AN: GOTO 950
940 PRINT R$
950 GOSUB 1970: HOME : ON KB GOTO 890,970,960
960 TEXT : RETURN
970 INPUT "AB=";A$: GOSUB 2040: IF A = 0 THEN 970
980 V(1) = A
990 INPUT "BD=";A$: GOSUB 2040: IF A = 0 THEN 990
1000 V(2) = A
1010 INPUT "AC=";A$: GOSUB 2040: IF A = 0 THEN 1010
1020 V(3) = A: IF V(3) < = V(1) THEN 1010
1030 T = 2: HOME : GOTO 900
1040 T = 0: FOR Z = 1 TO 3
1050 V(Z) = INT (10 * RND (1)) + 4 - Z: IF Z > 1 THEN IF V(Z) > = V(Z - 1) THEN 1050
1060 IF Z = 1 THEN 1080
1070 IF V(Z) > = V(Z - 1) THEN 1050
1080 NEXT Z
1090 GOSUB 1710: ON LE GOTO 1100,1110,1120
1100 AN = V(1) * V(2): VTAB 21: PRINT "AC=";V(1);" AE=";V(2): GOTO 1130
1110 AN = INT ((V(2) * SQR ((V(1) * V(1)) - ((V(2) - V(3)) * (V(2) - V(3)))))) * 100 + .50001) / 100: VTAB 21: PRINT "AD=";V(1);" DF=";V(2);" AB=";V(3): GOTO 1130
1120 AN = INT ((V(2) * ( SQR ((V(1) * V(1)) - (V(2) * V(2)))) + V(3))) * 100 + .50001) / 100: VTAB 21: PRINT "EB=";V(1);" BF=";V(2);" DE=";V(3)
1130 GOSUB 2020: IF A = AN THEN 1170
1140 PRINT W$: GOSUB 2100:T = T + 1: IF T = 3 THEN 1160
1150 HOME : GOTO 1090
1160 PRINT I$;" ";AN: GOTO 1180
1170 PRINT R$
1180 GOSUB 1970: HOME : ON KB GOTO 1040,1200,1190
1190 TEXT : RETURN
1200 ON LE GOTO 1210,1250,1310
1210 INPUT "AC=";A$: GOSUB 2040: IF A = 0 THEN 1210
1220 V(1) = A
1230 INPUT "AE=";A$: GOSUB 2040: IF A = 0 THEN 1230
1240 V(2) = A:T = 2: HOME : GOTO 1090
1250 INPUT "AD=";A$: GOSUB 2040: IF A = 0 OR A < 3 THEN 1250
1260 V(1) = A
1270 INPUT "DF=";A$: GOSUB 2040: IF A = 0 OR A > = V(1) THEN 1270
1280 V(2) = A
1290 INPUT "AB=";A$: GOSUB 2040: IF A = 0 OR A > = V(2) THEN 1290
1300 V(3) = A:T = 2: HOME : GOTO 1090
1310 INPUT "EB=";A$: GOSUB 2040: IF A = 0 OR A < 3 THEN 1310
1320 V(1) = A
1330 INPUT "BF=";A$: GOSUB 2040: IF A = 0 OR A > = V(1) THEN 1330
1340 V(2) = A
1350 INPUT "DE=";A$: GOSUB 2040: IF A = 0 OR A > = V(2) THEN 1350
1360 V(3) = A:T = 2: HOME : GOTO 1090
1370 T = 0: ON LE GOTO 1380,1400,1420
1380 V(1) = INT (10 * RND (1)) + 1:V(2) = INT (10 * RND (1)) + 1:V(3) = V(1) + (V(2) * 2)
```

Continued

## GEOMETRIX *Continued* — APPLE // Family

```
1390 GOSUB 1710: VTAB 21: PRINT "AB=";V(
     1);" AD=";V(2);" CF=";V(3):AN = (
     (V(1) + V(3)) * V(2)) / 2: GOTO 144
     0
1400 V(1) = INT (10 * RND (1)) + 1:V(2
     ) = V(1) + ( INT (10 * RND (1)) +
     1)
1410 GOSUB 1710: VTAB 21: PRINT "AB=";V(
     1);" CF=";V(2): PRINT "ANGLES C AN
     D F ARE BOTH 45 DEGREES":SG$ = "AD"
     :AN = (V(2) - V(1)) / 2: GOTO 1440
1420 V(1) = INT (10 * RND (1)) + 1:V(2
     ) = INT (10 * RND (1)) + 3:V(3) =
      V(1) + INT (10 * RND (1)) + 1: I
     F V(2) < = V(1) OR V(3) < = V(1)
     THEN 1420
1430 GOSUB 1710: VTAB 21: PRINT "AB=";V(
     1);" DB=";V(2);" CF=";V(3): PRINT
      "ANGLE C IS CONGRUENT TO ANGLE F":
     SG$ = "AC":AN = INT ( SQR (((V(3)
     - V(1)) / 2) ^ 2 + (V(2) ^ 2 - V(1)
      ^ 2)) * 100 + .50001) / 100
1440 IF LE > 1 THEN GOSUB 2030: GOTO 14
     60
1450 GOSUB 2020
1460 IF A = AN THEN 1500
1470 PRINT W$;: GOSUB 2100:T = T + 1: IF
      T = 3 THEN 1490
1480 HOME : ON LE GOTO 1390,1410,1430
1490 HOME : PRINT I$;" ";AN: GOTO 1510
1500 PRINT R$
1510 GOSUB 1970: HOME : ON KB GOTO 1370,
     1530,1520
1520 TEXT : RETURN
1530 ON LE GOTO 1540,1610,1650
1540 INPUT "AB=";A$: GOSUB 2040: IF A =
     0 THEN 1540
1550 V(1) = A
1560 INPUT "AD=";A$: GOSUB 2040: IF A =
     0 THEN 1560
1570 V(2) = A
1580 INPUT "CF=";A$: GOSUB 2040: IF A =
     0 THEN 1580
1590 V(3) = A: IF V(3) < V(1) THEN 1580
1600 T = 2: HOME : GOTO 1390
1610 INPUT "AB=";A$: GOSUB 2040: IF A =
     0 THEN 1610
1620 V(1) = A
1630 INPUT "CF=";A$: GOSUB 2040: IF A <
     V(1) THEN 1630
1640 V(2) = A:T = 2: HOME : GOTO 1410
1650 INPUT "AB=";A$: GOSUB 2040: IF A =
     0 THEN 1650
1660 V(1) = A
1670 INPUT "DB=";A$: GOSUB 2040: IF A =
     0 THEN 1670
1680 V(2) = A
1690 INPUT "CF=";A$: GOSUB 2040: IF A <
     V(1) THEN 1690
1700 V(3) = A:T = 2: HOME : GOTO 1430
1710 HGR : POKE 34,20: HCOLOR= 3: ROT= 0
     : SCALE= 1: ON SH GOTO 1720,1800,18
     80,1930
1720 ON LE GOTO 1730,1730,1760
1730 X = 139:Y = 79:HL = 40 * LE:VL = 40
     : HPLOT X - (HL / 2),Y - (VL / 2) T
     O X + (HL / 2),Y - (VL / 2) TO X +
     (HL / 2),Y + (VL / 2) TO X - (HL /
     2),Y + (VL / 2) TO X - (HL / 2),Y -
      (VL / 2)
1740 DRAW 1 AT X - (HL / 2) - 10,Y - (VL
      / 2): DRAW 2 AT X + (HL / 2) + 6,Y
      - (VL / 2): DRAW 3 AT X - (HL / 2)
      - 10,Y + (VL / 2) + 6: DRAW 4 AT X
      + (HL / 2) + 6,Y + (VL / 2) + 6
1750 RETURN
1760 HPLOT 59,59 TO 179,59 TO 179,79 TO
     139,79 TO 139,99 TO 59,99 TO 59,59
1770 FOR I = 59 TO 79 STEP 2: HPLOT 139,
     I: NEXT
1780 DRAW 1 AT 49,59: DRAW 2 AT 137,56:
     DRAW 3 AT 185,59: DRAW 4 AT 129,85:
      DRAW 5 AT 185,85: DRAW 6 AT 49,105
     : DRAW 7 AT 145,105
1790 RETURN
1800 ON LE GOTO 1810,1810,1840
1810 HPLOT 99,119 TO 179,119 TO 179,39 T
     O 99,119: SCALE= 4: DRAW 8 AT 171,1
     19: SCALE= 1
1820 DRAW 1 AT 93,129: DRAW 2 AT 185,129
     : DRAW 3 AT 179,35
1830 RETURN
1840 HPLOT 59,119 TO 219,119 TO 139,39 T
     O 59,119
1850 FOR I = 39 TO 119 STEP 2: HPLOT 139
     ,I: NEXT : SCALE= 4: DRAW 8 AT 131,
     119: SCALE= 1
1860 DRAW 1 AT 49,130: DRAW 2 AT 136,130
     : DRAW 3 AT 225,130: DRAW 4 AT 136,
     35
1870 RETURN
1880 HPLOT 99,59 TO 219,59 TO 179,99 TO
     59,99 TO 99,59: POKE 228,130: HPLOT
      179,60 TO 99,98: HCOLOR= 3
1890 FOR I = 59 TO 99 STEP 2: HPLOT 99,I
     : HPLOT 179,I: NEXT
1900 SCALE= 3: DRAW 8 AT 93,99: DRAW 8 A
     T 179,65: SCALE= 1
1910 DRAW 1 AT 97,55: DRAW 2 AT 177,55:
     DRAW 3 AT 221,55: DRAW 4 AT 49,109:
      DRAW 5 AT 97,109: DRAW 6 AT 177,10
     9
1920 RETURN
1930 HPLOT 99,59 TO 179,59 TO 219,99 TO
     59,99 TO 99,59: POKE 228,130: HPLOT
      179,60 TO 99,98: HCOLOR= 3
1940 FOR I = 59 TO 99 STEP 2: HPLOT 99,I
     : HPLOT 179,I: NEXT : SCALE= 3: DRA
     W 8 AT 93,99: DRAW 8 AT 99,65: DRAW
      8 AT 173,99: DRAW 8 AT 179,99: SCA
     LE= 1
1950 DRAW 1 AT 97,55: DRAW 2 AT 177,55:
     DRAW 3 AT 53,109: DRAW 4 AT 97,109:
      DRAW 5 AT 177,109: DRAW 6 AT 221,1
     09
1960 RETURN
1970 PRINT "PRESS: 1) ANSWER ANOTHER": P
     RINT TAB( 8)"2) ENTER VALUES": PRI
     NT TAB( 8)"3) EXIT";
1980 GOSUB 2000: IF KB < 177 OR KB > 179
      THEN 1980
1990 KB = KB - 176: RETURN
2000 KB = PEEK ( - 16384): IF KB > 128
     THEN POKE - 16368,0: RETURN
2010 GOTO 2000
2020 INPUT "WHAT IS THE AREA?";A$: GOSUB
      2040: RETURN
2030 PRINT "HOW LONG IS SEGMENT ";SG$;"?
     ": INPUT "LENGTH: ";A$: GOSUB 2040
     : RETURN
2040 IF A$ = "" THEN A = 0: RETURN
2050 FOR Z = 1 TO LEN (A$):B$ = MID$ (
     A$,Z,1): IF ((B$ < "0") + (B$ > "9"
     )) * (B$ < > ".") THEN 2090
2060 NEXT Z
2070 N = VAL (A$): IF ABS (N) < > IN
     T (N) THEN N = (N + .005) * 100:N =
      INT (N):N = N / 100
2080 A = N: RETURN
2090 CALL - 198: PRINT "BAD NUMBER -- R
     EDO INPUT!!":A = 0: RETURN
2100 CALL - 198: FOR D = 1 TO 1000: NEX
     T : RETURN
2110 DATA 8,0,18,0,31,0,44,0,57,0,68,0,
     83,0,95,0,106,0,36,36,12,12,14,14,5
     4,54,7,200,56,63,0,36,36,36,45,117,
     246,63,78,49,30,63,7,0,41,45,248,27
     ,36,36,12,45,21,6,0,41,45,36,36,36,
     45,173,54,54,30,63,7,0
2120 DATA 36,36,36,45,45,222,147,45,22
     2,18,45,45,0,36,36,36,36,36,45,45,2
     22,147,45,5,0,36,36,32,36,100,45,18
     1,154,53,62,63,4,0,36,45,54,63,0,36
     ,45,54
2130 DATA THAT IS RIGHT!,THAT IS WRON
     G!,THE ANSWER IS
```

HCM

## GEOMETRIX — COMMODORE 64

```
100 REM *************
110 REM * GEOMETRIX *
120 REM *************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM BY WAYNE KOBERSTEIN
160 REM AND THE HCM STAFF
170 REM HOME COMPUTER MAGAZINE
180 REM
190 REM C-64 BASIC
200 REM
210 PRINT"{SHIFT CLR}{CTRL WHT}":POKE53
    281,14:POKE53280,14:BL$="
    "
220 SL$="{HOME}{15CRSRDOWN}"
230 PRINT"{SHIFT CLR}"TAB(12)"{4CRSRDOW
    N}GEOMETRIX":PRINT"{3CRSRDOWN}{2CRS
    RRIGHT}ENTER YOUR CHOICE BY NUMBER:
    "
240 PRINTTAB(6)"{CRSRDOWN}1) RECTANGLES
    ":PRINTTAB(6)"{CRSRDOWN}2) RIGHT TR
    IANGLES"
250 PRINTTAB(6)"{CRSRDOWN}3) PARALLELOG
    RAMS":PRINTTAB(6)"{CRSRDOWN}4) TRAP
    EZOIDS"
260 PRINTTAB(6)"{CRSRDOWN}5) EXIT"
270 GETK$:IFK$=""THEN270
280 IFK$<"1"ORK$>"5"THEN270
290 IFK$="5"THENEND
300 SHAPE=VAL(K$)
310 PRINT"{SHIFT CLR}{5CRSRDOWN}{5CRSRR
    IGHT}SELECT A LEVEL BY NUMBER{4CRSR
    DOWN}"
320 PRINTTAB(7)"1) EASY":PRINTTAB(7)"{C
    RSRDOWN}2) AVERAGE":PRINTTAB(7)"{CR
    SRDOWN}3) HARD"
330 GETK$:IFK$=""THEN330
340 IFK$<"1"ORK$>"3"THEN330
```

Continued

```
350 LEV=VAL(K$):ON SHAPE GOSUB370,770,1140,1520
360 GOTO230
370 IF LEV=3THEN530
380 V(1)=INT(10*RND(0))+1:V(2)=V(1):T=0:IFLEV=1THENGOSUB2160:GOTO400
390 V(2)=INT(5*RND(0))+1:GOSUB2210
400 PRINTSL$"AB="V(1):PRINT"AC="V(2)
410 GOSUB2040:GOSUB2650:A=VAL(A$):IFA<=0THEN430
420 AN=V(1)*V(2):IFABS(AN-A)<.00001THENGOSUB2020:GOSUB2650:GOTO460
430 GOSUB1970:GOSUB2650
440 T=T+1:IFT=3THENGOSUB1950:GOSUB2650:GOTO460
450 GOSUB2000:GOTO400
460 GOSUB2000:GOSUB2090:ONKGOTO380,480,470
470 RETURN
480 T=2:ONLEVGOSUB2160,2210,2260
490 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO490
500 V(1)=A
510 INPUT"AC=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO510
520 V(2)=A:ONLEVGOSUB2160,2210:GOTO400
530 T=0:V(1)=INT(20*RND(0))+5:V(2)=V(1)-ABS(INT((V(1)-3)*RND(0))+1)
540 V(3)=INT(20*RND(0))+5:V(4)=V(3)-INT((V(3)-3)*RND(0))+1
550 IFV(4)>=V(3)THEN540
560 GOSUB2260
570 PRINTSL$"AC="V(1):PRINT"DE="V(2)
580 PRINT"AF="V(3):PRINT"DG="V(4)
590 GOSUB2040:A=VAL(A$):IFA<=0GOTO620
600 AN=(V(3)*(V(1)-V(2)))+(V(2)*(V(3)-V(4)))
610 IFABS(AN-A)<.00001THENGOSUB2020:GOSUB2650:GOTO640
620 GOSUB1970:GOSUB2650:T=T+1:IFT=3THENGOSUB1950:GOSUB2650:GOTO640
630 GOSUB2000:GOTO570
640 GOSUB2000:GOSUB2090:ONK GOTO530,660,650
650 RETURN
660 T=2:PRINT"{SHIFT CLR}":GOSUB2260:PRINTSL$
670 INPUT"AC=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO670
680 V(1)=A
690 INPUT"DE=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO690
700 V(2)=A
710 INPUT"AF=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO710
720 V(3)=A
730 INPUT"DG=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO730
740 V(4)=A:IFV(2)>=V(1)THENMG$="AC MUST BE LARGER THAN DE":GOSUB2670:GOTO660
750 IF V(4)>=V(3)THENMG$="AF MUST BE LARGER THAN DG":GOSUB2670:GOTO660
760 GOSUB2000:PRINT"{SHIFT CLR}":GOTO560
770 REM       RIGHT TRIANGLES
780 IF LEV=3THEN960
790 T=0:V(1)=INT(10*RND(0))+1:V(2)=INT(10*RND(0))+1
800 GOSUB2330
810 PRINTSL$"AB="V(1):PRINT"BC="V(2)
820 IF LEV=2 GOTO840
830 GOSUB2040:AN=V(1)*V(2)/2:GOTO850
840 SG$="AC":GOSUB2060:AN=INT((SQR(V(1)↑2+V(2)↑2))*100+.50001)/100
850 A=VAL(A$):IFA<=0THEN870
860 IFABS(AN-A)<.00001THENGOSUB2020:GOSUB2650:GOTO890
870 GOSUB1970:GOSUB2650:T=T+1:IFT=3THENGOSUB1950:GOSUB2650:GOTO890
880 GOSUB2000:GOTO810
890 GOSUB2000:GOSUB2090:ONKGOTO790,910,900
900 RETURN
910 GOSUB2330:PRINTSL$
920 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO920
930 V(1)=A
940 INPUT"BC=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO940
950 V(2)=A:T=2:PRINT"{SHIFT CLR}":GOTO800
960 T=0:V(1)=INT(10*RND(0))+1:V(2)=INT(10*RND(0))+1:V(3)=V(1)*2
970 GOSUB2380
980 PRINTSL$"AB="V(1):PRINT"BD="V(2):PRINT"AC="V(3)
990 GOSUB2050:A=VAL(A$)
1000 AN=INT((((V(1)*V(2)/2)+((V(3)-V(1))*V(2)/2))*100+.50001)/100
1010 IFABS(AN-A)<.00001THENGOSUB2020:GOSUB2650:GOTO1040
1020 GOSUB1970:GOSUB2650:T=T+1:IFT=3THENGOSUB1950:GOSUB2650:GOTO1040
1030 GOSUB2000:GOTO980
1040 GOSUB2000:GOSUB2090:ONKGOTO960,1060,1050
1050 RETURN
1060 GOSUB2380:PRINTSL$
1070 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1070
1080 V(1)=A
1090 INPUT"BD=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1090
1100 V(2)=A
1110 INPUT"AC=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1110
1120 V(3)=A:IFV(1)>=V(3)THENMG$="AC MUST BE LARGER THAN AB":GOSUB2670:GOTO1060
1130 T=2:PRINT"{SHIFT CLR}":GOTO970
1140 REM      PARALLELOGRAMS
1150 T=0:FORZ=1TO3
1160 V(Z)=INT(10*RND(0))+4-Z
1170 IF Z>1 THEN IF V(Z)>=V(Z-1) THEN1160
1180 NEXT
1190 GOSUB2450
1200 PRINTSL$:ONLEVGOTO1210,1220,1240
1210 AN=V(1)*V(2):PRINT"AC="V(1):PRINT"AE="V(2):GOTO1260
1220 AN=INT((V(2)*SQR(V(1)↑2-(V(2)-V(3))↑2))*100+.50001)/100
1230 PRINT"AD="V(1):PRINT"DF="V(2):PRINT"AB="V(3):GOTO1260
1240 AN=INT((V(2)*(SQR(V(1)↑2-V(2)↑2)+V(3)))*100+.50001)/100
1250 PRINT"EB="V(1):PRINT"BF="V(2):PRINT"DE="V(3)
1260 GOSUB2040:A=VAL(A$):IFA<=0THEN1280
1270 IFABS(AN-A)<.00001THENGOSUB2020:GOSUB2650:GOTO1300
1280 GOSUB1970:GOSUB2650:T=T+1:IFT=3THENGOSUB1950:GOSUB2650:GOTO1300
1290 GOSUB2000:GOTO1200
1300 GOSUB2000:GOSUB2090:ONKGOTO1150,1320,1310
1310 RETURN
1320 GOSUB2000:T=2:PRINTSL$:ON LEV GOTO1330,1370,1450
1330 INPUT"AC=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1330
1340 V(1)=A
1350 INPUT"AE=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1350
1360 V(2)=A:PRINT"{SHIFT CLR}":GOTO1190
1370 INPUT"AD=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1370
1380 V(1)=A
1390 INPUT"DF=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1390
1400 V(2)=A
1410 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1410
1420 V(3)=A:IFV(3)>=V(2)THENMG$="DF MUST BE LARGER THAN AB":GOSUB2670:GOTO1370
1430 IF(V(2)-V(3))>=V(1)THENMG$="DF MUST BE A LARGER NUMBER":GOSUB2670:GOTO1370
1440 PRINT"{SHIFT CLR}":GOTO1190
1450 INPUT"EB=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1450
1460 V(1)=A
1470 INPUT"BF=";A$:A=VAL(A$):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1470
1480 V(2)=A
1490 INPUT"DE=";A$:A=VAL(LEFT$(A$,10)):IFA<1THENPRINT"{2SHIFT CRSRUP}":GOTO1490
1500 V(3)=A:IFV(2)>=V(1)THENMG$="EB MUST BE LARGER THAN BF":GOSUB2670:GOTO1450
1510 GOTO1190
1520 REM      TRAPEZOIDS
1530 T=0:GOSUB2550:ONLEV GOTO1540,1570,1610
1540 V(1)=INT(10*RND(0))+1:V(2)=INT(10*RND(0))+1:V(3)=V(1)+(V(2)*2)
1550 PRINTSL$"AB="V(1):PRINT"AD="V(2):PRINT"CF="V(3)
1560 AN=((V(1)+V(3))*V(2))/2:GOTO1660
1570 V(1)=INT(10*RND(0))+1:V(2)=V(1)+(INT(10*RND(0))+1)
1580 PRINTSL$"AB="V(1):PRINT"CF="V(2)
1590 PRINT"THE ANGLES C AND F ARE":PRINT"BOTH 45 DEGREE ANGLES"
1600 SG$="AD":AN=(V(2)-V(1))/2:GOTO1660
```

Continued

## GEOMETRIX *Continued* COMMODORE 64

```
1610 V(1)=INT(10*RND(0))+1:V(3)=V(1)+(IN
     T(10*RND(0))+1)
1620 V(2)=INT(RND(0)*10)+3:IF V(2)<=V(1)
      THEN 1610
1630 PRINTSL$"AB="V(1):PRINT"DB="V(2):PR
     INT"CF="V(3)
1640 PRINT"ANGLE C IS EQUAL TO ANGLE F":
     SG$="AC"
1650 AN=INT(SQR(((V(3)-V(1))/2)↑2+(V(2)↑
     2-V(1)↑2))*100+.50001)/100
1660 IF LEV>1THEN1680
1670 GOSUB2040:GOTO1690
1680 GOSUB2060
1690 A=VAL(A$):IFA<=0THEN1710
1700 IFABS(AN-A)<.00001THENGOSUB2020:GOS
     UB2650:GOTO1730
1710 GOSUB1970:GOSUB2650:T=T+1:IFT=3THEN
     GOSUB1950:GOSUB2650:GOTO1730
1720 GOSUB2000:ON LEV GOTO1550,1580,1630
1730 GOSUB2000:GOSUB2090:ONKGOTO1530,175
     0,1740
1740 RETURN
1750 GOSUB2550:PRINTSL$:T=2:ONLEV GOTO17
     60,1820,1870
1760 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1760
1770 V(1)=A
1780 INPUT"AD=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1780
1790 V(2)=A
1800 INPUT"CF=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1800
1810 V(3)=A:GOSUB2000:GOTO1550
1820 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1820
1830 V(1)=A
1840 INPUT"CF=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1840
1850 V(2)=A:IFV(1)>=V(2)THENMG$="CF MUST
      BE LARGER THAN AB":GOSUB2670:GOTO1
     750
1860 GOSUB2000:GOTO1580
1870 INPUT"AB=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1870
1880 V(1)=A
1890 INPUT"DB=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1890
1900 V(2)=A
1910 INPUT"CF=";A$:A=VAL(LEFT$(A$,10)):I
     FA<1THENPRINT"{2SHIFT CRSRUP}":GOTO
     1910
1920 V(3)=A:IFV(1)>=V(3)THENMG$="CF MUST
      BE LARGER THAN AB":GOSUB2670:GOTO1
     750
1930 IFV(1)>=V(2)THENMG$="DB MUST BE LAR
     GER THAN AB":GOSUB2670:GOTO1750
1940 V(4)=SQR(V(2)↑2-V(1)↑2):GOSUB2000:G
     OTO1630
1950 REM        PRINT ANSWER
1960 PRINT"  THE ANSWER IS:"AN:RETURN
1970 REM        WRONG ANSWER
1980 PRINT"        THAT IS WRONG":FORMA=1TO2
     0:POKE54296,15:FORMT=1TO3:NEXT
1990 POKE54296,0:FORMT=1TO3:NEXT:NEXT:RE
     TURN
2000 REM        CLEAR WORD AREA
2010 PRINT"{HOME}{14CRSRDOWN}":FORX=1TO9
     :PRINTBL$:NEXT:RETURN
2020 REM        RIGHT ANSWER
2030 PRINT"        THAT IS RIGHT":RETURN
2040 REM        AREA?
2050 INPUT"{CRSRDOWN}WHAT IS THE TOTAL A
     REA";A$:A$=LEFT$(A$,20):RETURN
2060 REM        SEGMENT?
2070 PRINT"{CRSRDOWN}{4CRSRRIGHT}HOW LON
     G IS SEGMENT "SG$
2080 INPUT"{6CRSRRIGHT}LENGTH=";A$:A$=LE
     FT$(A$,20):RETURN
2090 REM        REPLAY
2100 PRINT"{HOME}{16CRSRDOWN}"
2110 PRINT"PRESS:":PRINT"       1) ANSWER AN
     OTHER":PRINT"       2) ENTER VALUES"
2120 PRINT"       3) EXIT"
2130 GETK$:IFK$=""THEN2130
2140 K=VAL(K$):IFK<1 OR K>3THEN2130
2150 RETURN
2160 REM RECTANGLES - LEVEL 1
2170 PRINT"{SHIFT CLR}"
2180 PRINTTAB(15)"A ";:FORX=1TO10:PRINT"
     {CMDR @}";:NEXT:PRINT" B"
2190 FORX=1TO7:PRINTTAB(17)"{CMDR G}
           {CMDR M}":NEXT
2200 PRINTTAB(15)"C ";:FORX=1TO10:PRINT"
     {CMDR T}";:NEXT:PRINT" D":RETURN
2210 REM RECTANGLES - LEVEL 2
2220 PRINT"{SHIFT CLR}"
2230 PRINTTAB(15)"A ";:FORX=1TO15:PRINT"
     {CMDR @}";:NEXT:PRINT" B"
2240 FORX=1TO7:PRINTTAB(17)"{CMDR G}
                {CMDR M}":NEXT
2250 PRINTTAB(15)"C ";:FORX=1TO15:PRINT"
     {CMDR T}";:NEXT:PRINT" D":RETURN
2260 PRINT"{SHIFT CLR}":REM RECTANGLES
      - LEVEL 3
2270 PRINTTAB(16)"B"
2280 PRINTTAB(5)"A ";:FORX=1TO20:PRINT"{
     CMDR P}";:NEXT:PRINT" C"
2290 FORX=1TO4:PRINTTAB(7)"{CMDR G}
                {CMDR M}":NEXT
2300 PRINTTAB(7)              "{CMDR G}
      D {9CMDR P}{SHIFT @} E"
2310 FORX=1TO5:PRINTTAB(7)"{CMDR H}
        {CMDR M}":NEXT
2320 PRINTTAB(5)"F {SHIFT L}";:FORX=1TO8
     :PRINT"{CMDR P}";:NEXT:PRINT"{SHIFT
     @} G":RETURN
2330 REM RIGHT TRIANGLES LEVEL 1&2
2340 PRINT"{SHIFT CLR}"
2350 PRINTTAB(21)"C":PRINTTAB(20)"{SHIFT
     N}{CMDR H}"
2360 FORX=1TO10:PRINTTAB(20-X)"{SHIFT N}
     ";SPC(X);"{CMDR G}":NEXTX
2370 PRINTTAB(8)"A {11CMDR Y} B":RETURN
2380 REM RIGHT TRIANGLES LEVEL 3
2390 PRINT"{SHIFT CLR}"
2400 PRINTTAB(21)"D":PRINTTAB(20)"{SHIFT
     N}{SHIFT M}"
2410 FORX=1TO8:PRINTTAB(20-X)"{SHIFT N}"
     ;SPC(X);"{CMDR G}";SPC(X-1)"{SHIFT
     M}":NEXT
2420 PRINTTAB(11)"{SHIFT N}        {SHIF
     T O}{CMDR H}      {SHIFT M}"
2430 PRINTTAB(11)"{20CMDR Y}"
2440 PRINTTAB(9)"A             B
     C":RETURN
2450 REM PARALLELOGRAMS LEVEL 1,2,& 3
2460 PRINT"{SHIFT CLR}"
2470 PRINTTAB(15)"A";SPC(10);"B";SPC(9)"
     C"
2480 PRINTTAB(15);:FORX=1TO22:PRINT"{CMD
     R P}";:NEXT:PRINT
2490 :PRINTTAB(14)"{SHIFT N}:";SPC(9);"/
     {CMDR Z}{CMDR X}";SPC(8);"{SHIFT N}
     "
2500 FORX=2TO9:Y=10-X:PRINTTAB(15-X)"{SH
     IFT N}";SPC(X-1);":";SPC(Y);"/";SPC
     (X-1);":";
2510 PRINTSPC(10-X);"{SHIFT N}":NEXT
2520 PRINTTAB(5)"{SHIFT N}          {CMDR
     A}{CMDR S}/         :{SHIFT N}"
2530 PRINTTAB(5)"{22CMDR Y}"
2540 PRINTTAB(5)"D           E           F
     ":RETURN
2550 REM TRAPEZOID LEVEL 1,2,& 3
2560 PRINT"{SHIFT CLR}"
2570 PRINTTAB(10)"A"SPC(10);"B"
2580 PRINTTAB(10);:FORX=1TO12:PRINT"{CMD
     R P}";:NEXT:PRINT
2590 :PRINTTAB(9)"{SHIFT N}{CMDR Z}{CMDR
      X}";SPC(8);"/:{SHIFT M}"
2600 FORX=2TO9:Y=10-X:PRINTTAB(10-X)"{SH
     IFT N}";SPC(X-1);":";SPC(Y);"/";SPC
     (X-1);":";
2610 PRINTSPC(X-1);"{SHIFT M}":NEXT
2620 PRINT"{SHIFT N}          {CMDR A}{CMD
     R S}/       {CMDR A}{CMDR S}
        {SHIFT M}"
2630 FORX=1TO32:PRINT"{CMDR Y}";:NEXT
2640 PRINT:PRINT"C"SPC(9);"D"SPC(11);"E"
     SPC(10)"F":RETURN
2650 REM        TIMER
2660 FORX9=1TO2000:NEXTX9:RETURN
2670 PRINTMG$:PRINT"HIT ANY KEY TO CONTI
     NUE":
2680 GETK$:IFK$=""THEN2680
2690 GOSUB2000:PRINTSL$:RETURN
```

HCM

## GEOMETRIX IBM PC & IBM PCjr

```
100 ' *************
110 ' *  GEOMETRIX  *
120 ' *************
130 ' COPYRIGHT 1985
140 ' EMERALD VALLEY PUBLISHING CO.
150 ' BY WAYNE KOBERSTEIN
160 ' AND THE HCM STAFF
170 ' HOME COMPUTER MAGAZINE
180 ' VERSION 5.3.1
190 ' IBM PCjr WITH CARTRIDGE BASIC and
      DOS 2.1 or
200 ' IBM PC WITH BASICA and
210 ' COLOR/GRAPHICS ADAPTER and
220 ' COLOR MONITOR
230 '
240 CLS:KEY OFF:SCREEN 1:COLOR 1,0:RAND
    OMIZE TIMER:PLAY "MF"
250 DEF FNR(X)=INT(X*100+.5001001)*.01
```

Continued

## GEOMETRIX *Continued*

```
260 ENTDR$="E3 D1 G2 D1 E2 D1 G1 D1 F1 U2 E1 R6 U3 L1 D2 L1 U2":LOCATE 12,15:PRINT "GEOMETRIX":LOCATE 22,10,0:PRINT "PRESS ";:PSET (120,171):DRAW ENTDR$:LOCATE 22,19:PRINT "TO CONTINUE";:GOSUB 1140
270 CLS:LOCATE 1,15:PRINT "GEOMETRIX":LOCATE 5,1:PRINT "SELECT ONE:":PRINT:PRINT "1) RECTANGLES":PRINT:PRINT"2) RIGHT TRIANGLES":PRINT:PRINT "3) PARALLELOGRAMS":PRINT:PRINT "4) TRAPEZOIDS":PRINT:PRINT "5) EXIT"
280 GOSUB 1140:IF A$<"1" OR A$>"5" THEN 280 ELSE SHAPE=VAL(A$):IF SHAPE=5 THEN CLS:END
290 LOCATE 5,13:IF SHAPE=1 THEN PRINT "RECTANGLES" ELSE IF SHAPE=2 THEN PRINT "RIGHT TRIANGLES" ELSE IF SHAPE=3 THEN PRINT "PARALLELOGRAMS" ELSE PRINT "TRAPEZOIDS"
300 LOCATE 17,1:PRINT "SELECT A LEVEL:":PRINT:PRINT "1) EASY":PRINT:PRINT"2) AVERAGE":PRINT:PRINT "3) HARD"
310 GOSUB 1140:IF A$=CHR$(13) THEN 270 ELSE IF A$<"1" OR A$>"3" THEN 310
320 LEV=VAL(A$):ON SHAPE GOSUB 360,520,660,840:GOTO 270
330 '
340 '   RECTANGLES
350 '
360 T=0:V1=INT(RND*10)+1:IF LEV=1 THEN V2=V1:S=1:GOTO 390 ELSE IF LEV=2 THEN V2=INT(RND*5)+1:S=2:GOTO 390 ELSE V3=INT(RND*20)+5:S=3:V4=V3-INT(RND*V3-3)+1:IF V4>=V3 THEN 370
380 V2=V1-ABS(INT(RND*(V1-3)+1)):IF V2>=V1 THEN 380
390 CLS:ON S GOSUB 1050,1060,1070:LOCATE 16,1:IF LEV<3 THEN PRINT "AB=";V1:PRINT "AC=";V2:AN=FNR(V1*V2) ELSE PRINT "AC=";V1:PRINT "DE=";V2:PRINT "AF=";V3:PRINT "DG=";V4:AN=FNR((V3*(V1-V2))+(V2*(V3-V4)))
400 GOSUB 990:IF AN=ANS THEN GOSUB 1020 ELSE GOSUB 1010:IF T<3 THEN 390 ELSE PRINT "THE ANSWER IS:";AN:T=0:FOR TD=1 TO 2000:NEXT
410 GOSUB 1030:IF NXT=1 THEN 360 ELSE IF NXT=3 THEN RETURN
420 LINE (0,116)-(319,199),0,BF:LOCATE 16,1:IF LEV=3 THEN 450
430 INPUT "AB=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 430
440 INPUT "AC=",V$:V2=VAL(V$):IF V2<.01 OR FNR(V1*V2)<.01 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 440 ELSE GOTO 390
450 INPUT "AC=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 450
460 INPUT "DE=",V$:V2=VAL(V$):IF V2>=V1 OR V2<.01 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 460
470 INPUT "AF=",V$:V3=VAL(V$):IF V3<.01 THEN LOCATE 18,4:PRINT SPACE$(30);:LOCATE 18,1:GOTO 470
480 INPUT "DG=",V$:V4=VAL(V$):IF V4>=V3 OR V4<.01 THEN LOCATE 19,4:PRINT SPACE$(30);:LOCATE 19,1:GOTO 480 ELSE T=2:GOTO 390
490 '
500 '   RIGHT TRIANGLES
510 '
520 T=0:V1=INT(RND*10)+1:V2=INT(RND*10)+1:S=4:IF LEV=3 THEN TV=INT(RND*5)+1:V1=SQR(V2^2+TV^2):V3=INT(RND*10)+1:S=5
530 CLS:ON S-3 GOSUB 1080,1090:LOCATE 16,1:IF LEV<3 THEN PRINT "AB=";V1:PRINT "BC=";V2:IF LEV=1 THEN AN=FNR(V1*V2/2) ELSE AN=FNR(SQR(V1^2+V2^2)) ELSE PRINT "AD=";V1:PRINT "DB=";V2:PRINT "BC=";V3:AN=FNR((SQR(V1^2-V2^2)*V2/2)+(V2*V3/2))
540 IF LEV=1 OR LEV=3 THEN GOSUB 990 ELSE SG$="AC":GOSUB 1000
550 IF AN=ANS THEN GOSUB 1020 ELSE GOSUB 1010:IF T<3 THEN 530 ELSE PRINT "THE ANSWER IS:";AN:T=0:FOR TD=1 TO 2000:NEXT
560 GOSUB 1030:IF NXT=1 THEN 520 ELSE IF NXT=3 THEN RETURN
570 T=2:LINE (0,99)-(319,199),0,BF:LOCATE 16,1:IF LEV=3 THEN 600
580 INPUT "AB=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 580
590 INPUT "AC=",V$:V2=VAL(V$):IF V2<.01 OR (LEV=1 AND FNR(V1*V2/2)<.01) OR (LEV=2 AND FNR(SQR(V1^2+V2^2))<.01) THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 590 ELSE GOTO 530
600 INPUT "AB=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 600
610 INPUT "BD=",V$:V2=VAL(V$):IF V2>=V1 OR V2<.01 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 610
620 INPUT "AC=",V$:V3=VAL(V$):IF V3<.01 THEN LOCATE 18,4:PRINT SPACE$(30);:LOCATE 18,1:GOTO 620 ELSE GOTO 530
630 '
640 '   PARALLELOGRAMS
650 '
660 T=0:S=6:V1=INT(RND*10)+4:V2=INT(RND*10)+1:IF V2>=V1 THEN 660
670 V3=INT(RND*10)+1:IF V3>=V2 THEN 670
680 CLS:GOSUB 1100:LOCATE 16,1:IF LEV=1 THEN PRINT "AC=";V1:PRINT "AE=";V2:AN=FNR(V1*V2):GOTO 700 ELSE IF LEV=2 THEN PRINT "AD=";V1:PRINT "DF=";V2:PRINT "AB=";V3:AN=FNR(V2*SQR(V1^2-(V2-V3)^2)):GOTO 700
690 PRINT "EB=";V1:PRINT "BF=";V2:PRINT "DE=";V3:AN=FNR(V2*(SQR(V1^2-V2^2)+V3))
700 GOSUB 990:IF AN=ANS THEN GOSUB 1020 ELSE GOSUB 1010:IF T<3 THEN 680 ELSE PRINT "THE ANSWER IS:";AN:T=0:FOR TD=1 TO 2000:NEXT
710 GOSUB 1030:IF NXT=1 THEN 660 ELSE IF NXT=3 THEN RETURN
720 LINE (0,116)-(319,199),0,BF
730 LOCATE 16,1:T=2:IF LEV>1 THEN 750 ELSE INPUT "AC=",V1:INPUT "AE=",V2:GOTO 680
740 INPUT "AE=",V$:V2=VAL(V$):IF V2<.01 OR FNR(V1*V2)<.01 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 740 ELSE GOTO 680
750 IF LEV>2 THEN 780 ELSE INPUT "DF=",V$:V2=VAL(V$):IF V2<.1 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 750
760 INPUT "AB=",V$:V3=VAL(V$):IF V3<.01 OR V3>=V2 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 760
770 INPUT "AD=",V$:V1=VAL(V$):IF V1<=V3 OR V1<.01 THEN LOCATE 18,4:PRINT SPACE$(30);:LOCATE 18,1:GOTO 770 ELSE GOTO 680
780 INPUT "BF=",V$:V2=VAL(V$):IF V2<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 780
790 INPUT "EB=",V$:V1=VAL(V$):IF V1<.01 OR V1<=V2 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 790
800 INPUT "DE=",V$:V3=VAL(V$):IF V3<.01 THEN LOCATE 18,4:PRINT SPACE$(30);:LOCATE 18,1:GOTO 800 ELSE GOTO 680
810 '
820 '   TRAPEZOIDS
830 '
840 T=0:S=7:V1=INT(RND*10)+1:IF LEV=1 THEN V2=INT(RND*10)+1:V3=V2*2+V1 ELSE IF LEV=2 THEN V2=V1+INT(RND*10)+1 ELSE V2=INT(RND*10)+V1:V3=V1+INT(RND*10)+1
850 IF LEV=1 THEN AN=FNR((V1+V3)*V2/2) ELSE IF LEV=2 THEN AN=FNR(V2-V1)/2:SG$="AD" ELSE AN=FNR(SQR(((V3-V1)/2)^2+((V2^2)-V1^2))):SG$="AC"
860 CLS:GOSUB 1120:LOCATE 16,1:IF LEV=1 THEN PRINT "AB=";V1:PRINT "AD=";V2:PRINT "CF=";V3:GOSUB 990:GOTO 880 ELSE IF LEV=2 THEN PRINT "AB=";V1:PRINT "CF=";V2:PRINT "THE ANGLES C AND F ARE BOTH 45 DEGREES":GOSUB 1000:GOTO 880
870 PRINT "AB=";V1:PRINT "DB=";V2:PRINT "CF=";V3:PRINT "ANGLE C IS EQUAL TO ANGLE F":GOSUB 1000
880 IF AN=ANS THEN GOSUB 1020 ELSE GOSUB 1010:IF T<3 THEN 860 ELSE PRINT "THE ANSWER IS:";AN:T=0:FOR TD=1 TO 2000:NEXT
890 GOSUB 1030:IF NXT=1 THEN 840 ELSE IF NXT=3 THEN RETURN
900 LINE (0,116)-(319,199),0,BF:T=2
910 LOCATE 16,1:IF LEV>1 THEN 940 ELSE INPUT "AB=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 910
920 INPUT "AD=",V$:V2=VAL(V$):IF V2<.01 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 920
930 INPUT "CF=",V$:V3=VAL(V$):IF V3<=V1 OR V3<.01 THEN LOCATE 18,4:PRINT SPACE$(30);:LOCATE 18,1:GOTO 930 ELSE GOTO 850
940 IF LEV>2 THEN 960 INPUT "AB=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 940
950 INPUT "CF=",V$:V2=VAL(V$):IF V2<=V1 THEN LOCATE 17,4:PRINT SPACE$(30);:LOCATE 17,1:GOTO 950 ELSE GOTO 850
960 INPUT "AB=",V$:V1=VAL(V$):IF V1<.01 THEN LOCATE 16,4:PRINT SPACE$(30);:LOCATE 16,1:GOTO 960
```

*Continued*

## GEOMETRIX *Continued*

IBM PC & IBM PCjr

```
 970 INPUT "DB=",V$:V2=VAL(V$):IF V2<=V1
      THEN LOCATE 17,4:PRINT SPACE$(30);
     :LOCATE 17,1:GOTO 970
 980 INPUT "CF=",V$:V3=VAL(V$):IF V3<=V1
      THEN LOCATE 18,4:PRINT SPACE$(30);
     :LOCATE 18,1:GOTO 980 ELSE GOTO 850
 990 INPUT "WHAT IS THE TOTAL AREA:",ANS
     :RETURN
1000 PRINT "LENGTH OF SEGMENT "+SG$+" IS
     :";:INPUT "";ANS:RETURN
1010 PRINT:PRINT "THAT IS NOT RIGHT!!":S
     OUND 110,9:FOR TD=1 TO 2000:NEXT:T=
     T+1:RETURN
1020 PRINT:PRINT "*** THAT IS RIGHT ***"
     :FOR TD=1 TO 30:SOUND TD*110,1:NEXT
     :FOR TD=1 TO 2000:NEXT:RETURN
1030 LINE (0,116)-(319,199),0,BF:LOCATE
     16,1:PRINT "SELECT ONE:":PRINT:PRIN
     T "1) ANSWER ANOTHER":PRINT:PRINT "
     2) ENTER YOUR OWN VALUES":PRINT:PRI
     NT "3) EXIT"
1040 GOSUB 1140:IF A$<"1" OR A$>"3" THEN
      1040 ELSE NXT=VAL(A$):RETURN
1050 LOCATE 2,16:PRINT "A      B":LOCATE
      7,16:PRINT "C      D":DRAW "BM132,
     12C2R40D40L40U40":RETURN
1060 LOCATE 2,13:PRINT "A         B":L
     OCATE 7,13:PRINT "C           D":DR
     AW "BM108,12C2R80D40L80U40":RETURN
1070 LOCATE 1,22:PRINT "B":LOCATE 2,13:P
     RINT "A              C":LOCATE 5,21
     :PRINT "D      E":LOCATE 10,13:PRIN
     T "F         G":DRAW "BM108,12C2R10
     4D24L40D40L64U64":RETURN
1080 LOCATE 2,22:PRINT "C":LOCATE 10,12:
     PRINT "A         B":DRAW "BM160,12C
     2D64L8U8R8D8L64E64":RETURN
1090 LOCATE 1,21:PRINT "D":LOCATE 13,11:
     PRINT "A         B         C":DRAW
     "BM164,12C2F80L160E80D4BD4D4BD4D4BD
     4D4BD4D4BD4D4BD4D4BD4D4BD4D4BD4D8L8
     U8R8":RETURN
1100 LOCATE 1,15:PRINT "A          B
           C":LOCATE 13,5:PRINT "D
         E          F":DRAW "BM120,12C2R
     160G80L160E80D4BD4D4BD4D4BD4D4BD4D4
     BD4D4BD4D4BD4D4BD4D4BD4D8L8U8R8BD8E
     4BE4E4BE4E4BE4E4BE4E4BE4E4BE4E4BE4E
     4BE4E4BE4E4BE4D8R8U8L8BD8D4BD4D4"
1110 DRAW "BD4D4BD4D4BD4D4BD4D4BD4D4BD4D
     4BD4D4BD4L8U8R8D8":RETURN
1120 LOCATE 1,15:PRINT "A          B":LO
     CATE 13,6:PRINT "C        D
      E         F":DRAW "BM120,12C2R80F8
     0L240E80D8R8U8L8BD8D4BD4D4BD4D4BD4D
     4BD4D4BD4D4BD4D4BD4D4BD4D4BD4L8U8R8
     D8E4BE4E4BE4E4BE4E4BE4E4BE4E4BE4E4B
     E4E4BE4E4BE4E4BE4"
1130 DRAW "D4BD4D4BD4D4BD4D4BD4D4BD4D4BD
     4D4BD4D4BD4D4BD4D4BD4L8U8R8D8":RETU
     RN
1140 A$=INKEY$:IF A$="" THEN 1140 ELSE R
     ETURN
```

HCM

## GEOMETRIX

TI-99/4A

```
100 REM   *************
110 REM   * GEOMETRIX *
120 REM   *************
130 REM   COPYRIGHT 1985
140 REM   EMERALD VALLEY PUBLISHING CO.
150 REM   BY WAYNE KOBERSTEIN
160 REM   AND THE HCM STAFF
170 REM   HOME COMPUTER MAGAZINE
180 REM   VERSION 5.3.1
190 REM   TI BASIC OR
200 REM   TI EXTENDED BASIC
210 REM
220 RANDOMIZE
230 CALL CLEAR
240 PRINT TAB(8);"GEOMETRIX":::::::::::
    "INITIALIZING..."
250 READ R$,W$,I$
260 DATA THAT IS RIGHT!,THAT IS WRONG!,
    THE ANSWER IS:
270 FOR A=96 TO 119
280 READ A$
290 CALL CHAR(A,A$)
300 NEXT A
310 CALL CLEAR
320 PRINT TAB(9);"GEOMETRIX":::::"ENTER
     YOUR CHOICE:"::"1) RECTANGLES"::"2
    ) RIGHT TRIANGLES"::
330 PRINT "3) PARALLELOGRAMS"::"4) TRAP
    EZOIDS"::"5) EXIT"::::::
340 GOSUB 4070
350 IF (K<49)+(K>53)THEN 340
360 IF K<>53 THEN 390
370 CALL CLEAR
380 END
390 SHAPE=K-48
400 CALL CLEAR
410 PRINT "SELECT A LEVEL:"::::"1) EASY
    "::"2) AVERAGE"::"3) HARD":::::::::
420 GOSUB 4070
430 IF (K<49)+(K>51)THEN 420
440 LEV=K-48
450 ON SHAPE GOSUB 470,1350,2150,2870
460 GOTO 310
470 CALL CLEAR
480 ON LEV GOTO 490,490,880
490 S=LEV
500 V(1)=INT(RND*10)+1
510 V(2)=V(1)
520 T=0
530 IF LEV=1 THEN 550
540 V(2)=INT(RND*5)+1
550 GOSUB 4110
560 PRINT "AB=";V(1):"AC=";V(2)::
570 GOSUB 3720
580 IF A>0 THEN 600
590 GOTO 660
600 AN=V(1)*V(2)
610 IF A=AN THEN 710
620 PRINT W$
630 GOSUB 4030
640 T=T+1
650 IF T=3 THEN 690
660 CALL CLEAR
670 GOSUB 4110
680 GOTO 560
690 PRINT I$;AN
700 GOTO 720
710 PRINT R$
720 GOSUB 3980
730 CALL CLEAR
740 ON K GOTO 500,760,750
750 RETURN
760 CALL CLEAR
770 T=2
780 GOSUB 4110
790 INPUT "AB=":A$
800 GOSUB 3820
810 IF A=0 THEN 790
820 V(1)=A
830 INPUT "AC=":A$
840 GOSUB 3820
850 IF A=0 THEN 830
860 V(2)=A
870 GOTO 570
880 S=3
890 V(1)=INT(RND*20)+5
900 V(2)=V(1)-ABS(INT(RND*(V(1)-3))+1)
910 V(3)=INT(RND*20)+5
920 V(4)=V(3)-INT(RND*(V(3)-3))+1
930 IF V(4)>=V(3)THEN 910
940 T=0
950 GOSUB 4110
960 PRINT "AC=";V(1):"DE=";V(2):"AF=";V
    (3):"DG=";V(4)::
970 GOSUB 3720
980 IF A>0 THEN 1000
990 1150
1000 AN=(V(3)*(V(1)-V(2)))+(V(2)*(V(3)-V
    (4)))
1010 IF A=AN THEN 1100
1020 PRINT W$
1030 GOSUB 4030
1040 T=T+1
1050 IF T=3 THEN 1080
1060 CALL CLEAR
1070 GOTO 950
1080 PRINT I$;AN
1090 GOTO 1110
1100 PRINT R$
1110 GOSUB 3980
1120 CALL CLEAR
1130 ON K GOTO 890,1150,1140
1140 RETURN
1150 CALL CLEAR
1160 T=2
1170 GOSUB 4110
1180 INPUT "AC=":A$
1190 GOSUB 3820
1200 IF A=0 THEN 1180
1210 V(1)=A
1220 INPUT "DE=":A$
1230 GOSUB 3820
1240 IF A=0 THEN 1220
1250 V(2)=A
1260 INPUT "AF=":A$
1270 GOSUB 3820
1280 IF A=0 THEN 1260
1290 V(3)=A
1300 INPUT "DG=":A$
1310 GOSUB 3820
1320 IF A=0 THEN 1300
1330 V(4)=A
1340 GOTO 950
1350 CALL CLEAR
1360 ON LEV GOTO 1370,1370,1760
1370 S=4
```

Continued

```
1380 T=0
1390 V(1)=INT(RND*10)+1
1400 V(2)=INT(RND*10)+1
1410 GOSUB 4110
1420 PRINT "AB=";V(1):"BC=";V(2)::
1430 IF LEV=2 THEN 1470
1440 GOSUB 3720
1450 AN=V(1)*V(2)/2
1460 GOTO 1500
1470 SG$="AC"
1480 GOSUB 3750
1490 AN=INT((SQR(V(1)^2+V(2)^2))*100+.50
     001)/100
1500 IF A=AN THEN 1590
1510 PRINT W$
1520 GOSUB 4030
1530 T=T+1
1540 IF T=3 THEN 1570
1550 CALL CLEAR
1560 GOTO 1410
1570 PRINT I$;AN
1580 GOTO 1600
1590 PRINT R$
1600 GOSUB 3980
1610 CALL CLEAR
1620 ON K GOTO 1370,1640,1630
1630 RETURN
1640 GOSUB 4110
1650 INPUT "AB=":A$
1660 GOSUB 3820
1670 IF A=0 THEN 1650
1680 V(1)=A
1690 INPUT "BC=":A$
1700 GOSUB 3820
1710 IF A=0 THEN 1690
1720 V(2)=A
1730 T=2
1740 CALL CLEAR
1750 GOTO 1410
1760 S=5
1770 T=0
1780 V(1)=INT(RND*10)+1
1790 V(2)=INT(RND*10)+1
1800 V(3)=V(1)*2
1810 GOSUB 4110
1820 PRINT "AB=";V(1):"BD=";V(2):"AC=";V
     (3)::
1830 SG$="DC"
1840 GOSUB 3750
1850 AN=INT(SQR(V(1)^2+V(2)^2)*100+.5000
     1)/100
1860 IF A=AN THEN 1950
1870 PRINT W$
1880 GOSUB 4030
1890 T=T+1
1900 IF T=3 THEN 1930
1910 CALL CLEAR
1920 GOTO 1810
1930 PRINT I$,AN
1940 GOTO 1960
1950 PRINT R$
1960 GOSUB 3980
1970 CALL CLEAR
1980 ON K GOTO 1760,2000,1990
1990 RETURN
2000 GOSUB 4110
2010 INPUT "AB=":A$
2020 GOSUB 3820
2030 IF A=0 THEN 2010
2040 V(1)=A
2050 INPUT "BD=":A$
2060 GOSUB 3820
2070 IF A=0 THEN 2050
2080 V(2)=A
2090 INPUT "AC=":A$
2100 IF A=0 THEN 2090
2110 V(3)=A
2120 T=2
2130 CALL CLEAR
2140 GOTO 1810
2150 CALL CLEAR
2160 S=6
2170 T=0
2180 FOR Z=1 TO 3
2190 V(Z)=INT(RND*10)+4-Z
2200 IF Z<2 THEN 2220
2210 IF V(Z)>=V(Z-1)THEN 2190
2220 NEXT Z
2230 GOSUB 4110
2240 ON LEV GOTO 2250,2280,2310
2250 AN=V(1)*V(2)
2260 PRINT "AC=";V(1):"AE=";V(2)::
2270 GOTO 2330
2280 AN=INT((V(2)*SQR(V(1)^2-(V(2)-V(3))
     ^2))*100+.50001)/100
2290 PRINT "AD=";V(1):"DF=";V(2):"AB=";V
     (3)::
2300 GOTO 2330
2310 AN=INT((V(2)*(SQR(V(1)^2-V(2)^2)+V(
     3)))*100+.50001)/100
2320 PRINT "EB=";V(1):"BF=";V(2):"DE=";V
     (3)
2330 GOSUB 3720
2340 IF A=AN THEN 2430
2350 PRINT W$
2360 GOSUB 4030
2370 T=T+1
```

```
2380 IF T=3 THEN 2410
2390 CALL CLEAR
2400 GOTO 2230
2410 PRINT I$,AN
2420 GOTO 2440
2430 PRINT R$
2440 GOSUB 3980
2450 CALL CLEAR
2460 ON K GOTO 2150,2480,2470
2470 RETURN
2480 ON LEV GOTO 2490,2590,2730
2490 INPUT "AC=":A$
2500 GOSUB 3820
2510 IF A=0 THEN 2490
2520 V(1)=A
2530 INPUT "AE=":A$
2540 GOSUB 3820
2550 IF A=0 THEN 2530
2560 V(2)=A
2570 T=2
2580 GOTO 2230
2590 INPUT "AD=":A$
2600 GOSUB 3820
2610 IF A=0 THEN 2590
2620 V(1)=A
2630 INPUT "DF=":A$
2640 GOSUB 3820
2650 IF A=0 THEN 2630
2660 V(2)=A
2670 INPUT "AB=":A$
2680 GOSUB 3820
2690 IF A=0 THEN 2670
2700 V(3)=A
2710 T=2
2720 GOTO 2230
2730 INPUT "EB=":A$
2740 GOSUB 3820
2750 IF A=0 THEN 2730
2760 V(1)=A
2770 INPUT "BF=":A$
2780 GOSUB 3820
2790 IF A=0 THEN 2770
2800 V(2)=A
2810 INPUT "DE=":A$
2820 GOSUB 3820
2830 IF A=0 THEN 2810
2840 V(3)=A
2850 T=2
2860 GOTO 2230
2870 CALL CLEAR
2880 S=7
2890 T=0
2900 ON LEV GOTO 2910,2980,3050
2910 V(1)=INT(RND*10)+1
2920 V(2)=INT(RND*10)+1
2930 V(3)=V(1)+(V(2)*2)
2940 GOSUB 4110
2950 PRINT "AB=";V(1):"AD=";V(2):"CF=";V
     (3)
2960 AN=((V(1)+V(3))*V(2))/2
2970 GOTO 3140
2980 V(1)=INT(RND*10)+1
2990 V(2)=V(1)+(INT(RND*10)+1)
3000 GOSUB 4110
3010 PRINT "AB=";V(1):"CF=";V(2):"THE AN
     GLES C AND F ARE":"BOTH 45 DEGREE A
     NGLES"
3020 SG$="AD"
3030 AN=(V(2)-V(1))/2
3040 GOTO 3140
3050 V(1)=INT(RND*10)+1
3060 V(2)=INT(RND*10)+3
3070 IF V(2)<=V(1)THEN 3050
3080 V(3)=V(1)+INT(RND*10)+1
3090 GOSUB 4110
3100 PRINT "AB=";V(1):"DB=";V(2):"CF=";V
     (3)
3110 PRINT "ANGLE C IS EQUAL TO ANGLE F"
3120 SG$="AC"
3130 AN=INT(SQR(((V(3)-V(1))/2)^2+(V(2)^
     2-V(1)^2))*100+.50001)/100
3140 IF LEV>1 THEN 3170
3150 GOSUB 3720
3160 GOTO 3180
3170 GOSUB 3750
3180 IF A=AN THEN 3270
3190 PRINT W$
3200 GOSUB 4030
3210 T=T+1
3220 IF T=3 THEN 3250
3230 CALL CLEAR
3240 ON LEV GOTO 2940,3000,3090
3250 PRINT I$;AN
3260 GOTO 3280
3270 PRINT R$
3280 GOSUB 3980
3290 CALL CLEAR
3300 ON K GOTO 2870,3320,3310
3310 RETURN
3320 GOSUB 4110
3330 ON LEV GOTO 3340,3480,3580
3340 INPUT "AB=":A$
3350 GOSUB 3820
3360 IF A=0 THEN 3340
3370 V(1)=A
3380 INPUT "AD=":A$
3390 GOSUB 3820
```

*Continued*

## GEOMETRIX *Continued*

TI-99/4A

```
3400 IF A=0 THEN 3380
3410 V(2)=A
3420 INPUT "CF=":A$
3430 GOSUB 3820
3440 IF A=0 THEN 3420
3450 V(3)=A
3460 T=2
3470 GOTO 2940
3480 INPUT "AB=":A$
3490 GOSUB 3820
3500 IF A=0 THEN 3480
3510 V(1)=A
3520 INPUT "CF=":A$
3530 GOSUB 3820
3540 IF A=0 THEN 3520
3550 V(2)=A
3560 T=2
3570 GOTO 3000
3580 INPUT "AB=":A$
3590 GOSUB 3820
3600 IF A=0 THEN 3580
3610 V(1)=A
3620 INPUT "DB=":A$
3630 GOSUB 3820
3640 IF A=0 THEN 3620
3650 V(2)=A
3660 INPUT "CF=":A$
3670 GOSUB 3820
3680 IF A=0 THEN 3660
3690 V(3)=A
3700 T=2
3710 GOTO 3090
3720 INPUT "WHAT IS THE AREA?":A$
3730 GOSUB 3820
3740 RETURN
3750 PRINT "HOW LONG IS SEGMENT ";SG$;"
     ?"
3760 INPUT "LENGTH:":A$
3770 GOSUB 3820
3780 RETURN
3790 INPUT "WHAT IS LENGTH OF PERIMETER?
     LENGTH:":A$
3800 GOSUB 3820
3810 RETURN
3820 IF A$<>"" THEN 3850
3830 A=0
3840 RETURN
3850 FOR Z=1 TO LEN(A$)
3860 B$=SEG$(A$,Z,1)
3870 IF ((B$<"0")+(B$>"9"))*(B$<>".")THE
     N 3940
3880 NEXT Z
3890 P=POS(A$,".",1)
3900 IF P=0 THEN 3920
3910 A$=SEG$(A$,1,P+2)
3920 A=VAL(A$)
3930 RETURN
3940 CALL SOUND(200,110,0)
3950 PRINT "BAD NUMBER -- REDO INPUT!!"
3960 A=0
3970 RETURN
3980 PRINT "PRESS"::"1) ANSWER ANOTHER":
     :"2) ENTER VALUES"::"3) EXIT"::
3990 GOSUB 4070
4000 IF (K<49)+(K>51)THEN 3990
4010 K=K-48
4020 RETURN
4030 CALL SOUND(500,110,0)
4040 FOR TD=1 TO 500
4050 NEXT TD
4060 RETURN
4070 CALL KEY(0,K,ST)
4080 IF ST=0 THEN 4070
4090 CALL SOUND(-1,880,0)
4100 RETURN
4110 GOSUB 4180
4120 READ A$
4130 IF A$="" THEN 4160
4140 PRINT TAB(5);A$
4150 GOTO 4120
4160 PRINT ::::
4170 RETURN
4180 ON S GOSUB 4200,4220,4240,4260,4280
     ,4300,4320
4190 RETURN
4200 RESTORE 4390
4210 RETURN
4220 RESTORE 4400
4230 RETURN
4240 RESTORE 4410
4250 RETURN
4260 RESTORE 4420
4270 RETURN
4280 RESTORE 4430
4290 RETURN
4300 RESTORE 4440
4310 RETURN
4320 RESTORE 4460
4330 RETURN
4340 DATA 0102040810204080,8040201008040
     201,8182848890A0C080,8141211109050З
     01,80402010080402FF,FF4020100804020
     1
4350 DATA FF,00000000000000FF,0101010101
     010101,8080808080808080,FF020408102
     04080,0102040810204OFF
4360 DATA 80C0A09088848281,0103050911214
     181,FF80808080808080,808080808080800
     FF,FF01010101010101
4370 DATA 0101010101010101FF,FF818181818
     1FF,00000000000000055,55,8000008080
     00080,01000001010000001
4380 DATA 010000081000000080
4390 DATA AnIffpB," i    h"," i    h"," i
      h",CogggqD,,
4400 DATA AnIffffffpB," i    h"," i
            h" ,  i        h",Cogggggggg
     qD,,
4410 DATA A    B    C,gggggggggggg,i
      v     h,i   v    h,i  Dhffffff E
     ,i    h,oggggq,F"  G,;
4420 DATA "   mC"  "  h"   h"," "  ` h"
     ,"  `  h",  "  ` h",Akgggg r B,,`
4430 DATA "   D"," "  "  ua"
     ,"  `  u a " D,"  ` u  a,"  ` u
      a",kgggggrggggd,A   B  C,,
4440 DATA   A"  B" C" ," mff
     ffffjrffffj",","  ` v w u ` ,"
      `  v w   u  `  "
4450 DATA   "  `  v  w  u ` " ," ` v w
      u`",kggggrkgggrb,D  E   F,,
4460 DATA "   A ggggg B" ," mr  wl
     ",","  ` v w u a ,"  ` v w  u a
     ",","  `  v  w u  a
4470 DATA "  `  v  w  u   a",kggggrkggg
     grrgggd,C  D   E   F,,
```

HCM

## MERGE-64

COMMODORE 64

```
100 REM ************
110 REM * MERGE-64 *
120 REM ************
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO
150 REM BY RANDY THOMPSON
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 5.3.1
180 REM COMMODORE 64 BASIC
190 FOR I=49152 TO 49330:READ D:CK=CK+D
    :POKE I,D:NEXT
200 IF CK<>22624 THEN PRINT"** ERROR IN
     DATA STATEMENTS **":END
210 PRINT"{SHIFT CLR}"TAB(15)"MERGE-64{
    4CRSRDOWN}":PRINT"TO USE:{CRSRDOWN}
    "
220 PRINT"FIRST LOAD A PROGRAM INTO THE
     COMPUTER."
230 PRINT"NEXT,{SPACE}INSERT THE DISK T
    HAT CONTAINS THE"
240 PRINT"PROGRAM THAT YOU WISH TO MERG
    E INTO"
250 PRINT"MEMORY AND TYPE:{CRSRDOWN}"
260 PRINT"SYS 49152,"CHR$(34)"FILE NAME
    "CHR$(34)"{2CRSRDOWN}"
270 PRINT"THE PROGRAM ON DISK WILL THEN
     MERGE WITH"
280 PRINT"THE PROGRAM IN MEMORY.":NEW
290 DATA 32,253,174,32,158,173,32,143
300 DATA 173,169,100,160,101,32,219,182
310 DATA 160,0,177,100,72,200,177,100
320 DATA 72,200,177,100,72,32,231,255
330 DATA 169,87,141,2,3,169,192,141
340 DATA 3,3,169,174,141,44,3,169
350 DATA 192,141,45,3,169,8,170,168
360 DATA 136,32,186,255,104,168,104,170
370 DATA 104,32,189,255,32,192,255,32
380 DATA 183,255,208,59,162,8,32,198
390 DATA 255,32,228,255,32,228,255,162
400 DATA 8,32,198,255,32,228,255,32
410 DATA 228,255,32,183,255,201,0,208
420 DATA 30,32,228,255,133,20,32,228
430 DATA 255,133,21,169,5,133,2,32
440 DATA 228,255,164,2,153,251,1,201
450 DATA 0,240,35,230,2,208,240,169
460 DATA 131,141,2,3,169,164,141,3
470 DATA 3,169,47,141,44,3,169,243
480 DATA 141,45,3,169,8,32,195,255
490 DATA 32,204,255,76,116,164,132,11
500 DATA 32,19,166,76,167,164,169,1
510 DATA 76,49,243
```

HCM

## ACHILLES and the TURTLE — RACE 1

TI-99/4A

```
TO WINNER↵
TELL 1 CARRY 10 SH 0 FD
 18↵
WAIT 12↵
TELL 2 CARRY 16 SH 230↵
WAIT 6↵
TELL 1 CARRY 14 FD 9 WA
IT 6↵
TELL 2 CARRY 17 FD 1 WA
IT 2↵
TELL 1 CARRY 11 SH 180
FD 9↵
WAIT 6↵
TELL 2 CARRY 16 SH 235
WAIT 2↵
TELL 1 CARRY 15 FD 18↵
WAIT 12↵
TELL 2 CARRY 17 SH 235
FD 1↵
WAIT 6↵
WINNER↵
END↵
```

```
TO WINYET?↵
TELL 1 MAKE "A XCOR↵
TELL 2 MAKE "T XCOR↵
TEST :T + 8 > :A↵
IFF PRINT [ACHILLES WIN
S!! ] WINNER↵
END↵
```

```
TO COURSE TIMES↵
IF :TIMES = 0 THEN STOP
↵
PU↵
FD :ASPEED + 5↵
PD↵
LT 90 FD :ASPEED + 10↵
RT 120 FD 9↵
RT 120 FD 9↵
LT 60 FD :ASPEED + 1↵
LT 90↵
MAKE "TIMES :TIMES - 1↵
COURSE :TIMES↵
END↵
```

```
TO SHORTRACE↵
TELL 1 CARRY 10 FD 2 WA
IT 1↵
TELL 2 CARRY 12 FD 1 WA
IT 1↵
WINYET?↵
TELL 1 CARRY 11 FD 3 WA
IT 1↵
TELL 2 CARRY 13 FD 1 WA
IT 1↵
WINYET?↵
SHORTRACE↵
END↵
```

```
TO VANISH↵
TELL :ALL↵
CARRY 0 SC 0 SS 0 SH 0↵
END↵
```

```
TO TILECHANGE SHADE↵
MAKE "TILENO 48↵
REPEAT 6 [TELL TILE :TI
LENO SC THING :SHADE MA
KE "TILENO :TILENO + 8
]↵
END↵
```

```
TO MARKOUT↵
TELL TURTLE↵
SXY - 120 ( - 5 )↵
COURSE 240 / ( 5 + :ASP
EED )↵
END↵
```

```
TO FIRSTRACE↵
VANISH↵
CS↵
CB :BLUE↵
TELL TURTLE HT SH 90↵
MAKE "ASPEED 16↵
SETCOLOR :YELLOW↵
MARKOUT↵
TILECHANGE "WHITE↵
TELL 1 SC :RUST SH 90↵
SXY - 126 0 SS 0↵
TELL 2 SC :GRAY SH 90↵
SXY 0 0 SS 0↵
SHORTRACE↵
END↵
```

## ACHILLES and the TURTLE — RACE 2

TI-99/4A

```
TO NEXTRACE↵
VANISH↵
TILECHANGE "WHITE↵
CB :BLACK↵
CS↵
PRINT [FOR THE DISTANCE
 BETWEEN ]↵
PRINT [ACHILLES AND THE
 TURTLE ]↵
PRINT [ ]↵
PRINT [...CURRENT SCAL
E = 1:1 ]↵
PRINT [ ]↵
PRINT [............DISP
LAYED:ACTUAL ]↵
PRINT [ ]↵
MAKE "ASPEED 8↵
MAKE "SCALE 1↵
MARKERS↵
TELL 1 SC :RUST SH 90↵
SXY - 126 0 SS :ASPEED↵
TELL 2 SC :GREEN SH 90↵
SXY 0 0 SS 2↵
GETSETGO↵
END↵
```

```
TO VANISH↵
TELL :ALL↵
CARRY 0 SC 0 SS 0 SH 0↵
END↵
```

```
TO TILECHANGE SHADE↵
MAKE "TILENO 48↵
REPEAT 6 [TELL TILE :TI
LENO SC THING :SHADE MA
KE "TILENO :TILENO + 8
]↵
END↵
```

```
TO FLAGS HEIGHT↵
IF :MARKER > 31 THEN ST
OP↵
REPEAT :HEIGHT [PUTTILE
 99 :MARKER :HT MAKE "H
T :HT - 1]↵
PUTTILE 100 :MARKER :HT
↵
MAKE "HT 10↵
MAKE "MARKER :MARKER +
( 2 * :HEIGHT )↵
FLAGS :HEIGHT↵
END↵
```

```
TO TURT↵
TELL 2↵
CARRY 13↵
FD 1 WAIT 2↵
CARRY 12↵
FD 1↵
END↵
```

```
TO CO2B↵
TELL 1↵
CARRY 14 WAIT 1↵
TELL 2↵
CARRY 16 WAIT 1↵
TELL 1 CARRY 15 WAIT 1↵
TELL 2 CARRY 17 WAIT 1↵
END↵
```

```
TO CO2↵
TELL 1↵
CARRY 10 WAIT 1↵
TELL 2↵
CARRY 13 WAIT 1↵
TELL 1 CARRY 11 WAIT 1↵
TELL 2 CARRY 12 WAIT 1↵
END↵
```

```
TO RACE↵
CARRY 11 WAIT 3↵
CARRY 10 WAIT 1↵
MAKE "APLACE XCOR↵
END↵
```

```
TO MARKERS↵
TELL TILE 99 SC :YELLOW
↵
MAKE "MARKER 0↵
MAKE "HT 10↵
MAKE "HEIGHT :ASPEED /
8↵
FLAGS :HEIGHT↵
END↵
```

```
TO CLOSEOUT↵
TELL 1 SX - 120 SH 90↵
TELL 2 SX 20 SH 90↵
REPEAT 3 [CO2 ]↵
TELL 1 SX 120 SH - 90↵
TELL 2 SX - 20 SH - 90↵
REPEAT 3 [CO2B ]↵
CLOSEOUT↵
END↵
```

```
TO RESUME↵
TYPE :SCALE PRINT [ :1 ]
↵
PRINT [ ]↵
MAKE "ASPEED :ASPEED +
8↵
IF :SCALE > 1024 THEN M
AKE "ASPEED :ASPEED - 8
↵
MARKERS↵
TELL 2 SS 5↵
TELL 1 SS :ASPEED↵
GETSETGO↵
END↵
```

```
TO ENOUGH!↵
TYPE 4096 PRINT [ :1 ]↵
PRINT [ ]↵
PRINT [...........AND B
EYOND! ..... ]↵
PRINT [ ]↵
TELL 1 SS 90↵
TELL 2 SS 40↵
CLOSEOUT↵
END↵
```

```
TO ACHILLES↵
TELL 1↵
SS :ASPEED↵
RACE↵
END↵
```

```
TO GETSETGO↵
TURT↵
ACHILLES↵
WITHIN26?↵
END↵
```

```
TO WITHIN26?↵
TELL 2↵
MAKE "TPLACE XCOR↵
MAKE "TVAL :ASPEED + 18
↵
TEST :SCALE > 64↵
IFT FASTSCALE↵
IFF TEST GREATER :TPLAC
E - :APLACE :TVAL↵
IFT GETSETGO↵
IFF BLOWUP↵
END↵
```

```
TO BLOWUP↵
MAKE "SCALE :SCALE * 4↵
TELL 1 SX - 104 SS 0↵
TELL 2 SX 0 SS 0↵
CS↵
PRINT [NEW SCALE FOR DI
STANCE]↵
PRINT [BETWEEN TURTLE A
ND ACHILLES: ]↵
PRINT [ ]↵
TYPE "...........↵
TEST :SCALE = 4096↵
IFT ENOUGH!↵
IFF RESUME↵
END↵
```

```
TO FASTSCALE↵
IF :SCALE = 256 THEN WA
IT 3↵
IF :SCALE = 1024 THEN W
AIT 2↵
IF :SCALE > 1100 THEN I
F :SCALE < 16000 THEN W
AIT 1 ELSE BACK 10↵
BLOWUP↵
END↵
```

## CHARACTER GRAHPICS DEMONSTRATION

APPLE // Family

```
100 REM  ********************
110 REM  * CHARACTER GRAPHICS *
120 REM  *   DEMONSTRATION      *
130 REM  ********************
140 REM  COPYRIGHT 1985
150 REM  EMERALD VALLEY PUBLISHING CO
160 REM    BY STEVE CORDON
170 REM  HOME COMPUTER MAGAZINE
180 REM  VERSION 5.3.1
190 REM  APPLE // FAMILY APPLESOFT
200 IF  PEEK (104) = 64 THEN 230
210 POKE 104,64: POKE 103,1: POKE 16384
    ,0
220 PRINT  CHR$ (4);"RUN CHARDEMO": REM
         YOU SHOULD SAVE THIS PROGRAM AS
     'CHARDEMO'.
230 GOSUB 370
240 HGR : HOME
250 CALL CHAR,1,"FFFFFFFFFFFFFFFF": CAL
    L CHAR,2,"2A552A552A552A7F"
260 CALL CHAR,4,"0C0C083F0C0C1233": CAL
    L CHAR,5,"0C0C241E0D0C1330": CALL C
    HAR,6,"0C0C081E2D0C1233": CALL CHAR
    ,7,"0C0C091E2C0C3203": CALL CHAR,8,
    "0C0C041E2D0C1233": CALL CHAR,9,"0C
    0C251E0C0C3300"
270 CALL CHAR,14,"0000000000000800"
280 CALL CHAR,16,"1C22322A26221C": CALL
     CHAR,17,"080C080808081C": CALL CHA
    R,18,"1C2201804023E": CALL CHAR,19
    ,"3E2010182021C": CALL CHAR,20,"10
    1814123E1010": CALL CHAR,21,"3E021E
    2020221C"
290 CALL CHAR,22,"3804021E22221C": CALL
     CHAR,23,"3E201008040404": CALL CHA
    R,24,"1C22221C22221C": CALL CHAR,25
    ,"1C22223C20100E"
300 CALL PRNT
310 FOR LOOP = 1 TO 4: FOR CNUM = 4 TO
    9: FOR VPS = 2 TO 20 STEP 2: CALL H
    CHAR,VPS,5,CNUM,30,0: NEXT VPS,CNUM
    ,LOOP
320 FOR LOOP = 1 TO 4: FOR VPS = 0 TO 2
    0: CALL HCHAR,VPS,0,1,40,2: NEXT VP
    S,LOOP
330 HGR : HCOLOR= 3: HPLOT 0,0 TO 279,0
     TO 279,159 TO 0,159 TO 0,0
340 FOR VPS = 1 TO 17:RNUM = INT ( RND
     (1) * 25) + 10: CALL HCHAR,VPS,0,2
    ,RNUM,0: HTAB 38: PRINT RNUM: NEXT
350 FOR LOOP = 1 TO 30:X = INT ( RND (
    1) * 1000) / 10: VTAB 19: HTAB 25:
    PRINT X;" ": FOR PAUSE = 1 TO 100:
     NEXT PAUSE,LOOP
360 TEXT : HOME : CALL OFF: END
370 IF  PEEK (2048) < > 76 THEN FOR K
     = 2048 TO 2336: READ P: POKE K,P:
    NEXT : READ P: IF P < > 9999 THEN
     PRINT "DATA ERROR": END
380 PRNT = 2048:OFF = PRNT + 3:CHAR = O
    FF + 3:HCHAR = CHAR + 3: RETURN
390 DATA 76,151,8,76,157,8,76,194,8,32,
    76,231,134,37,32,76,231,134,36,32,7
    6,231,134,8,32,76,231,134,6,32,76,2
    31,138,41,3,170,189,187,8,141
400 DATA 105,8,165,37,32,34,252,32,60,8
    ,169,160,32,240,253,198,6,208,244,9
    6,169,7,133,7,160,0,36,50,48,2,160,
    127,132,9,165,8,10,10,10,170
410 DATA 164,36,24,165,40,133,38,165,41
    ,105,24,133,39,24,165,39,105,4,133,
    39,189,25,9,69,9,17,38,145,38,232,1
    98,7,16,235,96,141,191,8,142,192
420 DATA 8,140,193,8,201,160,144,11,201
    ,191,176,7,41,31,133,8,32,60,8,173,
    191,8,174,192,8,172,193,8,76,240,25
    3,160,115,169,8,208,4,160,240,169
430 DATA 253,174,9,191,224,76,208,12,17
    4,15,191,208,7,140,48,190,141,49,19
    0,96,132,54,133,55,76,234,3,145,49,
    81,17,0,0,0,32,76,231,134,8,32
440 DATA 190,222,165,8,10,10,10,170,202
    ,160,255,132,6,165,6,73,255,133,6,2
    08,6,232,169,0,157,25,9,32,177,0,20
    1,34,240,31,73,48,201,10,144,8
450 DATA 105,152,201,16,144,2,169,0,36,
    6,208,4,10,10,10,10,29,25,9,157,25,
    9,76,213,8,232,138,41,7,240,7,169,0
    ,157,25,9,240,243,76,177
460 DATA  0,0,0,0,0,0,0,0,0,9999
```

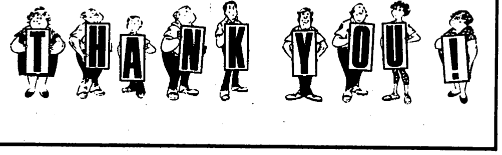
HCM

## CHARACTER GRAPHICS EDITOR

APPLE // Family

```
100 REM ********************
110 REM * CHARACTER GRAPHICS *
120 REM *         EDITOR        *
130 REM ********************
140 REM COPYRIGHT 1985
150 REM EMERALD VALLEY PUBLISHING CO.
160 REM BY STEVE CORDON
170 REM HOME COMPUTER MAGAZINE
180 REM VERSION 5.3.1
190 REM APPLE II FAMILY APPLESOFT
200 TEXT : HOME : SPEED= 255
210 X = 3:Y = 3:TBL = 768:HEX = 0: DIM
    CH%(8,8),CM%(7): FOR K = 0 TO 6:CM%
    (K) = 2 ^ K: NEXT : FOR K1 = 0 TO 7
    : FOR K2 = 0 TO 6:CH%(K1,K2) = 160:
     NEXT K2,K1
220 POKE 0,169: POKE 2,76: POKE 3,218:
    POKE 4,253: FOR I = 768 TO 780: POK
    E I,0: NEXT
230 INVERSE : FOR K = 3 TO 7: VTAB K: H
    TAB 8: PRINT SPC( 26): NEXT
240 VTAB 4: HTAB 9: PRINT "THE CHARACTE
    R SET EDITOR": NORMAL : VTAB 9: HTA
    B 4: PRINT "THE HEX CODE IS : 00000
    00000000000"
250 INVERSE : FOR K = 11 TO 20 STEP 9:
    VTAB K: HTAB 3: PRINT "----------":
    NEXT
260 FOR K = 12 TO 19: VTAB K: HTAB 3: P
    RINT "!";: HTAB 11: PRINT "!";: NEX
    T : NORMAL
270 VTAB 11: HTAB 20: PRINT "KEYS USED
    :": HTAB 14: FOR K = 1 TO 26: PRINT
     "-";: NEXT : PRINT
280 PRINT : HTAB 14: PRINT "A, Z, <-, -
    > = MOVE CURSOR": PRINT : HTAB 15:
    PRINT " SPACE BAR = INVERT DOT"
290 PRINT : HTAB 21: PRINT "C = CLEAR C
    HARACTER": PRINT : HTAB 21: PRINT "
    X = EXIT"
300 VTAB 12 + Y: HTAB 4 + X: GET A$:A =
     ASC (A$)
310 IF A = ASC ("A") THEN Y = Y - 1: I
    F Y < 0 THEN Y = 7
320 IF A = ASC ("Z") THEN Y = Y + 1: I
    F Y > 7 THEN Y = 0
330 IF A = 8 THEN X = X - 1: IF X < 0 T
    HEN X = 6
340 IF A = 21 THEN X = X + 1: IF X > 6
    THEN X = 0
350 IF A = ASC ("C") THEN RUN
360 IF A = ASC ("X") THEN VTAB 23: EN
    D
370 IF A = ASC (" ") THEN 390
380 GOTO 300
390 XX = CH%(Y,X):XX = XX - 32 + (64 *
    (XX = 160)):CH%(Y,X) = XX: VTAB 12
    + Y: HTAB 4 + X: PRINT CHR$ (XX)
400 VTAB 9: HTAB 2 * Y + 22:PK = 0: FOR
     X1 = 0 TO 6: IF CH%(Y,X1) = 192 TH
    EN PK = PK + CM%(X1)
410 NEXT : POKE 1,PK: CALL HEX: POKE TB
    L + Y,PK: GOTO 300
```

HCM



HCM Program Bug

# DeBUGS on DisPlay

During the production of every issue, corrections and/or enhancements to our programs are completed and tested in our programming laboratory. As the new version of a program is compared to the *last published version* by our "cross-checking" computer, a listing of all the differences is produced, transmitted to the computerized typesetter, and formatted in the same fashion as our standard listings.

This procedure for "DeBugs on Display" offers two advantages: (1) a standard presentation for updating your *HCM* programs that is clear and straightforward, and (2) inclusion of all published changes in "update files" which are placed *ON DISK*(TM) at the same time that the corrections appear in print. This is of special significance to Apple, IBM, and TI (Extended BASIC programs only) *ON DISK*(TM) subscribers, because the correction file can be directly "merged" with the original file—automatically updating it! The procedures for accomplishing this are included with the appropriate media.

Good news for Commodore 64 users with disk drives! The long-awaited utility program for merging files from disk is now available (see the Commodore Tech Note on page 67 of this issue for details). We are making progress on a similar application for merging files from tape on the Commodore 64, and hope to have it ready soon.

If you are going to type the corrections from "DeBugs on Display" directly into the original program, follow these steps:

1.) Load the original program into your computer's memory.

2.) Key-in the corrections as directed in the "Program Typing Guide" at the beginning of the Listings section.

3.) Any lines in the listing of corrections that state "***DELETED LINE," are to be deleted from the original program by entering the line number only and pressing either the (ENTER) or (RETURN) key (depending on your computer).

Each set of program corrections is prefaced by an identification bar that tells you: (1) the program name, (2) the volume and number of *HCM* in which the program was first published, (3) the number of the *last published version*, and (4) the computer brand to which the correction applies. Make sure that you are working with the right listing to ensure satisfactory results.

## OUTLINE EDITOR

HCM Vol. 5, No. 1 / Last level of DeBugs published—version .1

COMMODORE 64

```
 190 REM VERSION 5.1.4
 470 PRINT#15,"P"+FC$+RC$+PT$:RETURN
 560 RC=RN:GOSUB 460:GOSUB640:PT$=TP$:GO
     SUB 470:PRINT#2,TX$:GOSUB 420:RETUR
     N
 760 PRINT">■SHIFT CRSRLEFT■";:IF K$=ES$
      THEN MD=1:RETURN
1070 REM *** DELETED LINE
1280 IF(GC=0)*(MD=1)*(HF%(4))THEN HF=HF%
     (4):HL=HF%(5):GOSUB 1890:GOTO 1310
1300 IF MD=2 THEN HF%(6)=0:HF%(7)=0:GOSU
     B 450:GOTO 1320
1460 SC=SC+1:R=SC+2:GOSUB 5790:PRINT"@";
     :HL=SC%(SC):IF(SC-1)<=LP THEN 1360
1570 IFMD=2THENGOSUB1020
1590 IF (PA%)AND(MD=1)THEN GOSUB1820:GOT
     O1680
1601 IF MD=1 THEN 1610
1605 LNK%(TP,4)=0:LP=0:RN=TP:GOSUB420:GO
     SUB450:FT=0:GOSUB950:RETURN
1606 RETURN
1645 IF (PR%) OR (NR%=0) THEN 1660
1650 LNK%(NR%,2)=0:RN=NR%:GOSUB420:IFMD=
     1THENSF=NR%:HF%(8)=SF:GOTO1810
1655 LNK%(TP,4)=NR%:GOTO1810
1660 LNK%(NR%,2)=PR%:RN=NR%:GOSUB 420:LN
     K%(PR%,3)=NR%:RN=PR%:GOSUB 420
1770 IF NR% THEN LNK%(HL,3)=0:RN=HL:GOSU
     B 420
1810 GOSUB 450:IFMD=1THENSF=LNK%(PA%,1):
     GOSUB 5430:RETURN
1811 FT=LNK%(TP,4):GOSUB950:RETURN
1812 GOSUB950:RETURN
1900 CP%=HL:GOTO 1920
2040 HF%(1)=HF%(1)-1:GOSUB 5370:GOSUB 45
     0:RETURN
2050 REM *** DELETED LINE
2140 REM *** DELETED LINE
2150 REM *** DELETED LINE
2160 REM *** DELETED LINE
2170 REM *** DELETED LINE
2180 REM *** DELETED LINE
2190 REM *** DELETED LINE
2200 REM *** DELETED LINE
2209 REM --TEXT GRAB--
2210 TF=HF%(6):TL=HF%(7)
2211 IFHF%(3)=0THENHF%(3)=TF:FORZZ=0TO4:
     LNK%(HF%(3),ZZ)=0:NEXT:GOTO2213
2212 LNK%(HF%(3),3)=TF
2213 HF%(1)=HF%(1)-1:HF%(3)=TF
2214 IFTF=TLTHENLNK%(TF,3)=0:LNK%(TF,2)=
     0:RN=FT:GOSUB420:GOSUB450:RETURN
2215 RN=TF:GOSUB420:GOSUB450:TF=LNK%(TF,
     3):GOTO2211
2240 GOSUB 5850:PRINT" --INSERTING LIN
     E--";:IF MD=2 THEN 2370
2430 GOSUB 4810:NEXT:LP=TL:GOSUB 4810:SC
     %(0)=0:SC$(0)="":GOSUB 4950
2510 NR%=SC%(LP):PR%=LNK%(NR%,2):IF GC T
     HEN6380
2650 GOSUB 450:GOSUB950:RETURN
2660 REM *** DELETED LINE
2670 REM *** DELETED LINE
2680 REM *** DELETED LINE
2700 IF(MD=1)*(HF%(4)=0)THEN RETURN
2705 IF(MD=2)*(HF%(6)=0)THEN RETURN
2760 FOR I=1 TO 21:RN=HF:GOSUB 520:R=I:C
     =2:GOSUB 5790:PRINT TX$
2820 GOSUB 5850:PRINT" --RETURNING TO ED
     ITOR--";:IFMD=1THEN GOSUB 5430:RETU
     RN
2825 GOSUB 950:RETURN
2850 REM DELETE LINE
3050 GOSUB 3220:GOSUB 420:HF%(1)=HF%(1)-
     1:GOTO 3190
3100 SF=NR%:IF PA% THEN 3120
3130 IF PR%=0 THEN LNK%(TP,4)=NR%:RN=TP:
     GOSUB 420:FT=NR%
3140 GOSUB 3220:RN=RE:GOSUB 420:IF PR%=
     0 THEN HF%(1)=HF%(1)-1
3150 GOSUB 450:GOSUB 4920:GOSUB 950:RETU
     RN
3190 GOSUB 450:GOSUB 5430:RETURN
3250 PRINT"■SHIFT CLR■":R=1:C=10:GOSUB 5
     790:PRINT"SORT UTILITY":GOSUB 5860
3260 REM *** DELETED LINE
3550 IF K$=ES$ THEN MD=1:RETURN
3610 FORK=1TO8:IFK$=FU$(K)THEN3630
3630 ON K GOSUB 3650,3650,1280,1280,2240
     ,2470,2700,2860
3635 IF LNK%(TP,4)=0 THEN MD=1:RETURN
3850 IF CX=LEN(T$)+1 THEN 3510
3855 IFMD=2 THEN 3880
3860 IF(CX=38)+(LEN(T$)>36)THEN 3520
3870 T$=(LEFT$(T$,CX-2))+SP$+MID$(T$,CX-
     1):PRINTK$;:GOTO 3510
3880 T$=LEFT$(LEFT$(T$,CX-2)+SP$+MID$(T$
     ,CX-1)+CL$,38):PRINTSP$+MID$(T$,CX)
     ;
3890 IF(LEN(T$)<38)+(RIGHT$(T$,1)<=SP$)T
     HEN 3510
3900 TL=LP:TC=CX:GOSUB 4010:LP=TL-1:CX=T
     C:GOTO 3480
4320 IFZ=1THENN$=RIGHT$(N$,LEN(N$)-1):GO
     TO 4300
4340 LB=LEN(WD$):IFLB+1<=SPTHEN4420
4350 REM *** DELETED LINE
4410 GOTO 4180
4440 IFN$=""THEN4460
4445 IFLB=LEN(N$)THENN$="":GOTO4460
4450 N$=RIGHT$(N$,LEN(N$)-LB-1)
```

Continued

## OUTLINE EDITOR

***Continued***

**HCM Vol. 5, No. 1 / Last level of DeBugs published—version .1**

COMMODORE 64

```
4460  RN=BR:TX$=M$:GOSUB 560:RN=GR:TX$=N$
      :GOSUB 560:GOTO 4180
4500  IFLNK%(NR,3)=0THENHF%(3)=NR:GOSUB 4
      50:GOTO 4520
4530  IF(M$<>"*P")AND(M$<>"*B")THENRETURN
4570  RN=BR:GOSUB 520:M$=TX$:GOTO 4530
4580  IF(N$<>"*P")AND(N$<>"*B")THENRETURN
4590  IFM$>" "THENBR=LNK%(BR,3):IFBR=GRTHE
      N 4650
4610  REM *** DELETED LINE
4620  DN=0:BR=LNK%(BR,3):GOSUB 4690:IFLNK
      %(GR,3)=0THENDN=1:RETURN
4630  GR=LNK%(GR,3):GOSUB4710:IF(N$="*B")
      OR(N$="*P")THEN4600
4650  IFLNK%(BR,3)=0THENDN=1:RETURN
4660  DN=0:BR=LNK%(BR,3):GOSUB 4690:IF DN
      =1THENRETURN
4690  GOSUB 4695:GOSUB 4530:RETURN
4695  IFBR<2THENDN=1:M$="":RETURN
4710  GOSUB 4715:GOSUB 4580:RETURN
4715  IFGR<2THENDN=1:N$="":RETURN
4780  FORSS=LEN(N$)TO1STEP-1:IFMID$(N$,SS
      ,1)>CHR$(32)THEN4800
4800  N$=LEFT$(N$,SS):RETURN
5650  DN$=CHR$(17):PL$=CHR$(112):TP$=CHR$
      (1):PT=1:FC$=CHR$(2)
5860  R=3:C=1:GOSUB 5790:PRINT "SELECT ON
      E:"
5870  R=5:C=3:GOSUB 5790:PRINT "1) CHILDR
      EN OF CURRENT PARENT"
5880  R=7:GOSUB 5790:PRINT "2) ONE GENERA
      TION THROUGHOUT OUTLINE"
5890  R=9:GOSUB 5790:PRINT "3) ALL CHILDR
      EN IN THE OUTLINE"
5900  GET A$:IF A$<"1" OR A$>"3" THEN 590
      0
5910  R=12:C=0:GOSUB 5790:PRINT "ARE YOU
      SURE YOU WANT OPTION #"A$" (Y/N)"
5920  GET AN$:IF AN$="" THEN 5920
5930  IF AN$="N" THEN R=12:GOSUB 5790:PRI
      NT CL$:GOTO 5860
5940  IF AN$="Y" THEN OP=VAL(A$):GOTO5960
5950  GOTO 5920
5960  GOSUB 5850:PRINT " -- SORTING. . .
      PLEASE WAIT --";
5970  ON OP GOSUB5980,6140,6280:GOSUB 543
      0:RETURN
5980  SP=PA%
5990  IF SP=0 THEN SC=HF%(8):F1=SC:RN=SC:
      GOSUB 520:SC$=TX$:GOTO6020
6000  IF LNK%(SP,1) THEN SC=LNK%(SP,1):RN
      =SC:GOSUB 520:SC$=TX$:GOTO6020
6010  RETURN
6020  SD=LNK%(SC,3):IF SD THEN RN=SD:GOSU
      B 520:SD$=TX$:GOSUB6080:GOTO6040
6030  RETURN
6040  IF LNK%(SD,3) THEN SD=LNK%(SD,3):RN
      =SD:GOSUB 520:SD$=TX$:GOTO6070
6050  SC=LNK%(SC,3):IF SC THEN RN=SC:GOSU
      B 520:SC$=TX$:GOTO6020
6060  RETURN
6070  GOSUB6080:GOTO6040
6080  IF SD$>SC$ THEN RETURN
6090  TS$=SC$:SC$=SD$:SD$=TS$:TL=LNK%(SC,
      1):LNK%(SC,1)=LNK%(SD,1):LNK%(SD,1)
      =TL
6100  IF LNK%(SC,4) THEN LNK%(LNK%(SC,4),
      0)=SC
6110  IF LNK%(SD,4) THEN LNK%(LNK%(SD,4),
      0)=SD
6120  RN=SC:TX$=SC$:GOSUB 560
6130  RN=SD:TX$=SD$:GOSUB 560:RETURN
6140  R=16:C=0:GOSUB 5790:PRINT "ENTER GE
      NERATION TO SORT     ■3SHIFT CRSRLEF
      T■";
6150  LO$="0":HI$="9":LN=3:OT$="":GOSUB 1
      160
6160  IF OT$="" THEN6140
6170  R=19:GOSUB 5790:SL=1:SP=2:SR=VAL(OT
      $):PRINT "GEN: "SR;
6180  IF SR=0 THEN SP=0:GOSUB5990:RETURN
6190  IF SL>=SR THEN6230
6200  IF LNK%(SP,1) THEN GOTO6220
6210  IF LNK%(SP,3)>1 THEN SP=LNK%(SP,3):
      GOTO6200
6220  IF LNK%(SP,1)THENSP=LNK%(SP,1):SL=S
      L+1:C=5:GOSUB5790:PRINT SL;
6230  IF SL=SR THEN GOSUB5990:IF LNK%(SP,
      3) THEN SP=LNK%(SP,3):GOTO6230
6240  IF LNK%(SP,0)=0 THEN RETURN
6250  SP=LNK%(SP,0):SL=SL-1:C=5:GOSUB 579
      0:PRINT SL;
6260  IF LNK%(SP,3) THEN SP=LNK%(SP,3):GO
      TO6190
6270  GOTO6240
6280  SP=0:S=0:R=19:C=1:PRINT "GEN: "S:
6290  GOSUB5990:SP=HF%(8):GOSUB5990
6300  IF LNK%(SP,1) THEN6370
6310  IF LNK%(SP,3) THEN SP=LNK%(SP,3):GO
      SUB5990:GOTO6300
6320  IF LNK%(SP,0) THEN SP=LNK%(SP,0):S=
      S-1:C=36:R=24:GOSUB 5790:PRINT S;
6330   IF LNK%(SP,0)=0 THEN IF LNK%(SP,3)
      =0 THEN6360
6340   IF LNK%(SP,3)THEN SP=LNK%(SP,3):GO
      SUB5990:GOTO6300
6350   IF LNK%(SP,3)THEN SP=LNK%(SP,3):GO
      SUB5990:GOTO6300
6360  RETURN
6370  SP=LNK%(SP,1):S=S+1:C=36:R=24:GOSUB
       5790:PRINT S;:GOSUB5990:GOTO6300
6380  IF MD=2 THEN6820
6390  PP=PA%:CL=0:RE=HF:GOSUB6590:FC=CR
6400  IF(PA%)*(PR%=0) THEN LNK%(PA%,1)=CR
      :SF=CR:RN=PA%:GOSUB 420:GOTO6420
6410  IF(PR%=0) THEN HF%(8)=CR:SF=CR
6420  IF LNK%(RE,1)=0 THEN6450
6430  RE=LNK%(RE,1):IF RE=FC THEN RE=NR%
6440  CL=CL+1:PP=CR:GOSUB6650:GOTO6420
6450  IF(LNK%(RE,3)=0)+(RE=HL) THEN6500
6460  PR%=CR:RE=LNK%(RE,3):IF RE=FC THEN
      RE=NR%:GOTO6480
6470  IF RE=HL THEN6560
6480  GOSUB6590:IF RE=HL THEN6560
6490  GOTO6420
6500  IF LNK%(RE,0)=0 OR RE=HL THEN CC%=C
      R:GOSUB 5430:RETURN
6510  RE=LNK%(RE,0):CR=LNK%(CR,0):PP=LNK%
      (CR,0):CL=CL-1
6520  IF CL=0 THEN6550
6530  IF LNK%(RE,0) THEN PR%=CR:RE=LNK%(R
      E,3):GOSUB6590:GOTO6420
6540  GOTO6500
6550  IF RE<>HL THEN PR%=CR:RE=LNK%(RE,3)
      :GOSUB6590:GOTO6420
6560  LNK%(CR,3)=NR%:RN=CR:GOSUB 420
6570  LNK%(NR%,2)=CR:RN=NR%:GOSUB 420
6580  CC%=CR:GOSUB 5430:RETURN
6590  GOSUB6670;GOSUB 420
6600  LNK%(CR,0)=PP:LNK%(CR,2)=PR%:LNK%(C
      R,3)=0:RN=CR:GOSUB 420
6610  IF PR% THEN LNK%(PR%,3)=CR:RN=PR%:G
      OSUB 420
6620  IF(NR%=0)+(RE<>HL)THEN RETURN
6630  LNK%(NR%,2)=CR:RN=NR%:GOSUB 420
6640  LNK%(CR,3)=NR%:RN=CR:GOSUB 420:RETU
      RN
6650  GOSUB6670:LNK%(PP,1)=CR:RN=PP:GOSUB
       420
6660  LNK%(CR,0)=PP:LNK%(CR,1)=0:LNK%(CR,
      2)=0:LNK%(CR,3)=0:RN=CR:GOSUB 420
6670  IF HF%(2)=0 THEN 5050
6680  CR=HF%(2):HF%(2)=LNK%(HF%(2),3):HF%
      (1)=HF%(1)+1:GOSUB 450
6690  RN=RE:GOSUB 520:RN=CR:GOSUB 560
6700  IF LNK%(RE,4) THEN GOSUB6720
6710  RETURN
6720  TR=LNK%(RE,4):GOSUB6780:LNK%(CR,4)=
      TC:RN=CR:GOSUB 420
6730  LNK%(TC,0)=CR:FOR I=1 TO 4:LNK%(TC,
      I)=0:NEXT:RN=TC:GOSUB 420
6740  RN=LNK%(RE,4):GOSUB 520:RN=TC:GOSUB
       560
6750  IFLNK%(TR,3)=0 THEN RETURN
6760  TR=LNK%(TR,3):PC=TC:GOSUB6780:LNK%(
      PC,3)=TC:RN=PC:GOSUB 420
6770  LNK%(TC,2)=PC:RN=TC:GOSUB 420:GOTO6
      750
6780  IF HF%(2)=0 THEN 5050
6790  TC=HF%(2):HF%(2)=LNK%(HF%(2),3):HF%
      (1)=HF%(1)+1:GOSUB 450
6800  RN=TR:GOSUB 520
6810  RN=TC:GOSUB 560:RETURN
6820  GOSUB 1040:CR=HF
6830  GOSUB 4950:IF PR% THEN LNK%(PR%,3)=
      RN:LNK%(RN,2)=PR%:PR%=RN:GOTO6860
6840  LNK%(TP,4)=RN:FT=RN:RE=RN:RN=TP:GOS
      UB 420
6850  RN=RE:LNK%(RN,0)=TP:LNK%(NR%,0)=0:P
      R%=RN
6860  IF CR=HF THEN FC=PR%
6870  RE=RN:RN=CR:GOSUB 520:RN=RE:GOSUB 5
      60:GOSUB 420
6880  IF(CR<>HL)*(LNK%(CR,3)<>FC)THEN CR=
      LNK%(CR,3):IF CR THEN6830
6881  IF(CR<>HL)*(LNK%(CR,3)<>FC) THEN IF
       CR=0 THEN CR=HL
6910  IF CR<>HL THEN6940
6920  LNK%(NR%,2)=RN:LNK%(RN,3)=NR%:GOSUB
       420
6930  RN=NR%:GOSUB 420:GOSUB 950:RETURN
6940  CR=NR%:GOTO6830
```

## OUTLINE EDITOR

HCM Vol. 5, No. 1 / Last level of DeBugs published version—.1

COMMODORE 64

The Explanation of the Program originally published for the *Outline Editor* for the C-64 was in error. This issue we've included an extensive update for that program. Below is the explanation that covers the completely updated version of the program.

### *Outline Editor* (C-64) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-210 | Program header. |
| 220-230 | Main program loop. |
| 240-300 | Chain back to menu. |
| 310-400 | Open file. |
| 410-430 | Update a link. |
| 440-450 | Update pointers. |
| 460 | Calculate lo-byte, hi-byte for positioning disk head. |
| 470 | Position disk head. |
| 480-490 | Read in records. |
| 500 | Initialize pointers. |
| 510-540 | Get a line of text. |
| 550-560 | Print a line of text. |
| 570-580 | Initialize links. |
| 590-600 | Reconstruct pointer string. |
| 610-660 | Text string manipulation. |
| 670-870 | Ouline editor control loop. |
| 880-890 | Single line editing. |
| 900-1000 | Text entry. |
| 1010-1050 | Save text screen. |
| 1060 | Get disk status. |
| 1080-1150 | Disk error—reRUN program. |
| 1160-1260 | Input routine. |
| 1270-1606 | Grab and copy records. |
| 1610-1670 | Copy for outline mode. |
| 1680-1850 | Grab for outline mode. |
| 1860-2130 | Delete line. |
| 2209-2220 | Grab for text mode. |
| 2230-2450 | Insert a line. |
| 2460-2650 | Paste a line. |
| 2690-2825 | Display hold buffer. |
| 2830-2840 | Clear a screen line. |
| 2850-3210 | Delete line. |
| 3220-3240 | Clear links and set buffer. |
| 3250-3270 | Sort routine vector. |
| 3280-3330 | Move cursor up a line. |
| 3340-3370 | Move cursor down a line. |
| 3380-3410 | Go back a generation. |
| 3420-3450 | Go forward a generation. |
| 3460-3830 | Text editor control loop. |
| 3840-3990 | Insert a character. |
| 4000-5040 | Text editing subroutines. |
| 5050-5150 | Records are all used message. |
| 5160-5300 | Create a new record. |
| 5310-5410 | Print status line. |
| 5420-5600 | Display the screen. |
| 5610 | Clear screen buffer. |
| 5620-5690 | Initialize variables. |
| 5700-5770 | Initialize screen. |
| 5780-5790 | Plot cursor at C,R. |
| 5800-5840 | Disk error—reRUN program. |
| 5850 | Clear a line on the screen. |
| 5860-6710 | Sort routines. |
| 6720-6940 | Paste for a text copy. |

## THE ORGANIZER MAIN MENU

HCM Vol. 5, No. 1 / Last level of DeBugs published version—.1

COMMODORE 64

```
  190 REM VERSION 5.1.2
  220 POKE 46,110:CLR
```

## SLITHER

HCM Vol. 4, No. 5 / Last level of DeBugs published version—.1

APPLE // Family

In *HCM* Vol. 4, No. 5 the Explanation of the Program for the Apple *II* Family version of *Slither* was incorrect. Below is the correct explanation for that program.

### *Slither* (Apple *II* Family) Explanation of the Program

| Line Nos. | |
|---|---|
| 100-170 | Program header. |
| 180-190 | Relocate program if it isn't above the high resolution screen. |
| 200-270 | Define the shape tables. |
| 280-320 | Load the sound routine. |
| 330-370 | Initialize the variables. |
| 380-440 | Initialize the screen for the game. |
| 450 | Draw level 1. |
| 460 | Draw level 2. |
| 470 | Draw level 3. |
| 480 | Draw level 4. |
| 490-500 | Draw level 5. |
| 510 | Set up snake. |
| 520-580 | Position the egg on the screen. |
| 590-670 | Main program control loop; scan the keyboard for movement. |
| 680-740 | Move the snake on the screen. |
| 750-890 | Check for the snake colliding with obstacles. |
| 900-940 | Select a direction for the egg to move. |
| 950-960 | Move the egg up. |
| 970-980 | Move the egg right. |
| 990-1000 | Move the egg down. |
| 1010-1020 | Move the egg left. |
| 1030-1160 | Routine to give the player a chance to take a break between levels. |
| 1170-1230 | Snake crashes into an obstacle. |
| 1240-1290 | End of the game routine. |
| 1300-1380 | Introduction routine. |

## BACKGAMMON

HCM Vol. 5, No. 1 / Last level of DeBugs published version—.1

APPLE // Family

```
  170 REM VERSION 5.1.2
 1320 IF (X + A6 > A5 AND P(X + A6) < 2)
      THEN 2000
```

## MYSTERY WORDS

HCM Vol. 4, No. 2 / Last level of DeBugs published—version .2

APPLE // Family

The listing from *HCM* Vol. 5, No. 2 of *Mystery Words*, in "DeBugs on Display" requires a change at the end of line 630: GOTO 3900 REM VERSION 4.2.2 *should read:* GOTO 390

## REPORTS

HCM Vol. 5, No. 1 / Last published version :1

APPLE // Family

At the end of Line 240 of *The Organizer Dummy Reports* EN HOME : GOTO 300 should be deleted (no version change).

## MAIN MENU

**HCM Vol. 5, No. 1 / Last published version .1**

APPLE // Family

Line 820 of *The Organizer Main Menu* should read: GOTO 810 (no version change).

## MYSTERY WORDS

**HCM Vol. 4, No. 2 / Last level of DeBugs published version—.1**

IBM PC & IBM PCjr

```
160 REM VERSION 4.2.2
170 REM IBM PCjr CARTRIDGE BASIC FROM
         DOS 2.1 OR
180 REM IBM PC BASICA WITH COLOR
         GRAPHICS ADAPTER AND
190 REM COLOR MONITOR
200 SCREEN 1:COLOR 0,1:CLS:DIM S$(59):D
    EF SEG=&HF000:ZZZ=PEEK(&HFFFE):DEF
    SEG:IF ZZZ=253 THEN BEEP OFF:SOUND
    ON
380 PLAY "MF"
500 GOSUB 730:LOCATE 18,1:PRINT "OUT OF
     TIME!":PRINT:SOUND 220,9:PRINT "TH
    E CORRECT ANSWER IS: ";AN$:FOR TD=1
     TO 3000:NEXT TD:CLS:GOSUB 710:GOT
    O 400
570 T(PL)=1:IF T(0)=1 AND T(1)=1 THEN T
    (0)=0:T(1)=0:CLS:LOCATE 12,7:PRINT
    "YOU BOTH BLEW IT!":PRINT:PRINT TAB
    (7);"THE WORD WAS ";AN$:GOTO 615
610 GOSUB 730:PLAY "L36O3CDEFGABO4CDEFG
    ABL6O2"
615 LOCATE 18,1:PRINT P1$;"!":PRINT "YO
    UR SCORE IS: ";BLUE:PRINT:PRINT P2$
    ;"!":PRINT "YOUR SCORE IS: ";RED:FO
    R TD=1 TO 3000:NEXT TD:CLS:GOSUB 71
    0:GOTO 400
700 Y=HP*8:DRAW "C3;BM=Y;,=Z;XNOTE$;":P
    LAY "O=OCT;XN$;":RETURN
710 CLS:FOR Z=20 TO 60 STEP 10:LINE (0,
    Z)-(319,Z),1:NEXT Z:FOR Z=80 TO 120
     STEP 10:LINE (0,Z)-(319,Z),1:NEXT
    Z
720 DRAW "C2BM17,52;XTREB$;BM18,52;XTRE
    B$;BM19,522;XTREB$;BM8,90;XBASS$;BM
    9,90;XBASS$;BM10,90;XBASS$;":RETURN
730 LINE (0,127)-(319,199),0,BF:RETURN
```

## SEA OF STATES (disk version only)

**HCM Vol. 4, No. 2 / Last level of DeBugs published version—.4**

## SNAP—CALC

**HCM Vol. 4, No. 3 / Last level of DeBugs published version—.4**

TI-99/4A

When RUN on a system without memory expansion, both of these programs require the CALL FILES (1) command prior to LOADing. *Snap-Calc* has 5673 bytes free for data and logic input. The *Snap-Calc* files offered *ON DISK* (TM) 4.5—SEATTLE and TRIPCOST—for example, leave 1668 bytes free after being LOADed into memory.

## SWITCH 'N' SPELL

**HCM Vol. 5, No. 2 / Last level of DeBugs published—version .1**

IBM PC & IBM PCjr

```
 180 ' VERSION 5.2.2
1270 FOR I = 1 TO LEN(WORD$)-1:L=LEN(WOR
     D$):R=INT(RND*L+1):SW$=SW$+MID$(WOR
     D$,R,1):WORD$=LEFT$(WORD$,R-1)+MID$
     (WORD$,R+1):NEXT:SW$=WORD$+SW$:IF S
     W$=ORIGINAL$ THEN 1260 ELSE WORD$=S
     W$
```

## LASERITHMETIC

**HCM Vol. 5, No. 2 / Last level of DeBugs published version—.1**

IBM PC & IBM PCjr

```
180 ' VERSION 5.2.2
540 SOUND RND*100+110,2:NEXT:PLAY "MF"
560 LOCATE 20,1:PRINT "YOUR SCORE IS:";
    RIT;:LOCATE 22,1:PRINT "PRESS ";CHR
    $(17);CHR$(217);" TO CONTINUE":DEF
    SEG=0:POKE 1050,PEEK(1052):GOSUB 62
    0:GOTO 260
```

## OUTLINE EDITOR

**HCM Vol. 5, No. 1 / Last level of DeBugs published version—.2**

TI-99/4A

In *HCM* Vol. 5, No. 2, we included an update for the TI-99/4A version of the *Outline Editor* program. Here is an Explanation of the Program for that version when completely updated.

### *Outline Editor* (TI-99/4A)*
### Explanation of the Program

| Line Nos. | |
|---|---|
| 100-250 | Program header. |
| 260-310 | Initialize the program. |
| 320-390 | Load the file links. |
| 400-510 | Main control routine for the outline editor. |
| 520-820 | Delete a line. |
| 830-990 | Insert a line |
| 1000-1560 | Paste a line. |
| 1570-1660 | Main menu for sort the outline. |
| 1670-1840 | Sort the children under one parent. |
| 1850-1920 | Locate all parents on a given level throughout the outline and branch to the sort routine. |
| 1930-1980 | Locate every level of the outline and branch to sort each one. |
| 1990-2350 | Format the text from the text editor. |
| 2360-2840 | Grab and copy a line. |
| 2850-2930 | Display the hold buffer. |
| 2940-2970 | Edit line routine. |
| 2980-3070 | Main control loop for the text editor. |
| 3080-3160 | Main control loop for the line pointer. |
| 3170-3250 | Get a screen of text for the text editor. |
| 3260-3310 | Save a screen of text from the text editor. |
| 3320-3520 | Line entry routine for the text editor. |
| 3530-3590 | Delete character for the text editor. |
| 3600-3720 | Insert a character in the text editor. |
| 3730-3760 | Erase a line in the text editor. |
| 3770-3990 | Word wrap routine. Used to wrap a word down to the next line. |
| 4000-4030 | Display a line of text. |
| 4040 | Get a line of text. |
| 4050 | Display the screen header for the program. |
| 4060-4160 | Control disk I/O operations. |
| 4170-4330 | Get a screen of outline records. |
| 4340-4420 | Control loop to fill an outline screen. |
| 4430-4530 | Get the next outline record for the outline editor. |
| 4540-4630 | Get the next text record for the text editor. |
| 4640-4670 | Display out of space message when the file is full. |
| 4680 | DATA for the header display of the three modes of operation. |
| 4690-4710 | Subroutine will return the link value from within a string. |
| 4720-4780 | Subroutine will place a new link value back into a string. |

***Extended BASIC required**

## LOAN CALCULATOR—BASIC or EXTENDED BASIC VERSION

**HCM Vol. 5, No. 1 / Last level of DeBugs published version—.1**

TI-99/4A

```
170 REM VERSION 5.1.2
180 REM TI BASIC OR
190 REM EXTENDED BASIC
560 INPUT "YEARS OF LOAN ":N
```

## EVACU-POD

**HCM Vol. 5, No. 2 / Last level of DeBugs published version—.1**

TI-99/4A

```
 170 ! VERSION 5.2.2
2450 DATA "YEAR OF MISSION:",,1) 2485,,2
     ) 2248,,3) 2157,,4) 1995,,,SELECT O
     NE
```

# SENSATIONAL SOFTWARE GIVEAWAY

## 1 YOUR CHOICE OF THIS ISSUE'S PROGRAMS AT A REMARKABLE PRICE—

ONLY $4.95!
plus shipping & handling
Price will change to $6.95 upon becoming a "Back Issue."
(See date on order form below)

**FOR SUBSCRIBERS & NEWSSTAND PURCHASERS ONLY***

To participate in our monthly Software Giveaway, you need to be a bonafide purchaser of ***Home Computer Magazine*** and fill out the questionnaire on the reverse side.

Are you presently a magazine subscriber?
□ Yes □ No

Have you taken advantage of our Software Giveaway before?
□ Yes □ No

You will receive all the programs ON TAPE™ or ON DISK™ (which have versions for your selected machine) whose listings appear in this issue.

**IMPORTANT:** *The order form below and the questionnaire on the reverse side must be completed. Tear out this entire page and enclose in an envelope along with $5.95 ($6.95 in Canada, $9.95 Foreign Airmail). Payment must be made by check, money order, or VISA/MasterCard. Proof of purchase (subscriber label number, sales receipt or any reasonable facsimile thereof) must also accompany this form.*
**Non-subscriber and non-purchaser price is $9.95 in the U.S.*

**TO PROPERLY HANDLE THE VOLUME OF SOFTWARE REQUESTS, ORDERS MUST BE SENT IN ON THIS FORM, NO PHOTOCOPIES. SORRY, WE CANNOT ACCEPT TELEPHONE ORDERS FOR THIS SERVICE.**

## *FREE SOFTWARE!

## When Subscribing To 2 HOME COMPUTER magazine™

Subscribe or Renew today, and with your paid subscription you will receive **FREE** software—**ON TAPE™** or **ON DISK™**. With a 1-year subscription to ***Home Computer Magazine*** you get **2 FREE issues** (a $11.90 Premium Value) of **ON TAPE™** or **ON DISK™**. Subscribe for 2 years and receive **4 issues** (a $23.80 Premium Value). And with a 3-year subscription we'll give you **6 full issues** (a $35.70 Premium Value) of this convenient software on cassette tape or floppy disk.

***DON'T HAVE A COMPUTER? TAKE A RAINCHECK!**

We'll give you a raincheck for the FREE software so that when you buy a computer, we will send you your choice of ON TAPE™ or ON DISK™ as a FREE BONUS as stated in this offer.

**And You Save Up To 40% Off The Single-Copy Price of the Magazine!**

## SAVE EVEN MORE!

## 3 AND ENJOY THE CONVENIENCE OF A PROGRAM SUBSCRIPTION

By subscribing to **ON TAPE™** or **ON DISK™** you will save money off the single-copy price **and** receive the same high-quality programs published in each issue of the magazine—**delivered right to your door!**

This cassette tape or floppy disk program service is the convenient, accurate, and affordable way to save hundreds of typing hours.

**The Perfect Addition To Your Magazine Subscription!**

**NOTE:** SEE SPECIAL LIMITED-TIME GET-ACQUAINTED OFFER ON CENTER BIND-IN CARD.

3 FANTASTIC OFFERS

# 3-IN-1 ORDER FORM

PLEASE PRINT

Name ______________________________

Address ______________________________

City ______________________ State _____ Zip __________

**TOTAL** ______________________

□ Check or Money Order Enclosed

**MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK**

Bill my □ VISA □ MasterCard

□□□□□□□□□□□□□□□□

Account No.

Phone No. ______________________________

______________________________ ______________
Signature Exp. Date

**Enclose payment or credit card information and mail entire page with completed form to:**

HOME COMPUTER MAGAZINE
P. O. Box 70288, Eugene, OR 97401
—OR—
Use our TOLL-FREE LINE for VISA/MasterCard only:
1-800-828-2212 (Minimum $10. Order)
In Oregon, Alaska, Hawaii Tel.(503)485-8796
Open Monday through Friday—7:00AM to 4:00PM
Pacific Standard Time

**THIS SECTION MUST BE COMPLETED**

**INDICATE YOUR CHOICE OF MEDIA: (check one box only)**

**ON TAPE™:** □**C-64** □**TI-99/4A**

**ON DISK™:** □**Apple *II* Family** □**C-64** □**IBM PC** □**IBM PCjr** □**TI-99/4A**

□ **Please send a raincheck for the FREE software.**

Vol. 5 No. 3

**1 YES!** Send me all the programs in this issue: which have versions for my selected machine. I have indicated my choice of media and have enclosed $4.95 ($5.95 in Canada, $9.95 Foreign Airmail).

**THIS PRICE GOOD THROUGH June 15, 1985**
**After this date, order at back-issue prices from center bind-in card.**
*Proof of purchase and completed questionnaire must be included.*

**2 YES!** Enter my subscription to ***Home Computer Magazine*** for the term below:
Please check one: □ New □ Renewal (include subscriber number below)

Label Number: ______________________________

| | | |
|---|---|---|
| □ 1-yr (10 issues) $25 | □ 2-yr (20 issues) $45 | □ 3-yr (30 issues) $63 |
| ***PLUS 2 FREE***<br>ON TAPE or ON DISK | ***PLUS 4 FREE***<br>ON TAPE or ON DISK | ***PLUS 6 FREE***<br>ON TAPE or ON DISK |

**Canada add $11 per year; Foreign Surface add $21 for 1-yr magazine subscription. Free software offer available in U.S. & Canada only.**

**3 YES!** I want to save time and money. Please enter my program subscription* to **ON TAPE** or **ON DISK** for the term listed below:
□ 8 ISSUES — *ONLY $38* "An Extension To Your 2 Free Issues of Software"
□ 10 ISSUES — *ONLY $45* "A Full Year Of Program Convenience"
*Does not include magazine.

SEE SPECIAL OFFER ON CENTER BIND-IN CARD

**Canada add $5 for software subscription.**
**Software subscription not available in other countries at this time.**

***For subscriptions, please allow 6-8 weeks for your first issue—Magazine & Media shipped separately.***

Satisfaction Guaranteed—or the unfilled portion of your subscription will be refunded, less the cost of any premiums you have received.

Prices Subject To Change Without Notice *Defective media gladly exchanged. NO REFUNDS on media.* SGV4530485

# HOME COMPUTER™ magazine

## QUESTIONNAIRE

Complete and mail to: Home Computer Magazine • P.O. Box 70288 • Eugene, Oregon 97401

### FOR ALL READERS

1. **Where did you obtain this copy of Home Computer Magazine?** □Subscriber □Supermarket □Bookstore □Users group □Newsstand □Computer Store □Friend □Library □Other ____________
2. **What types of software are you most interested in?** □Educational □Entertainment □Computer Literacy □Household Management □Job-Related Applications □Business □Other ____________
3. **Are you** □Male □Female □14 or younger □15-24 □25-34 □35-44 □45-54 □55+
4. **Annual Household Income?** □Under $10,000 □$10,000-$14,999 □$15,000-$19,999 □$20,000-$24,999 □$25,000-$29,999 □$30,000-$39,999 □$40,000-$49,999 □$50,000+
5. **Occupation?** □Professional □Management □Teacher □Student □Other ____________
6. **What is your ZIP code?** □□□□□
7. **What is the current month and year?** ____________
8. **Do you presently own a Home Computer?** □No □Yes. It is a □TI-99/4A □Apple II/II+/IIe □Commodore 64 □VIC-20 □IBM PC □PCjr □Other ____________

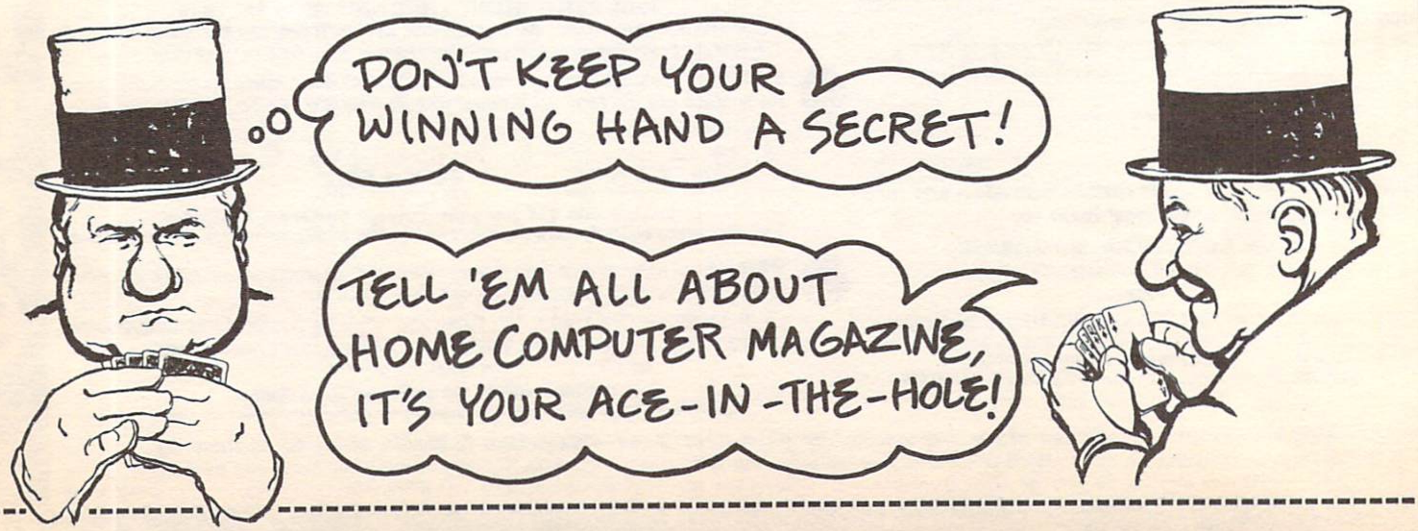
### FOR READERS WHO PLAN TO BUY A HOME COMPUTER

9. **Which model do you think you'll purchase?** □Apple IIe □Commodore 64 □VIC-20 □IBM PC □PCjr □TI-99/4A □Other ____________
10. **When do you expect that purchase to be?** □less than 3 months □3-6 months □7-12 months □at least 1 year
11. **What do you anticipate your primary use of a home computer will be?** □Entertainment □Education □Computer Literacy □Household Management □Job-Related Applications □Business □Other ____________

### FOR PRESENT HOME COMPUTER USERS

12. **Which home computer(s) do you currently own?** □Apple II/II+/IIe □Commodore 64 □VIC-20 □IBM PC □PCjr □TI-99/4A □Other ____________
13. **What is the primary use of your home computer?** □Entertainment □Education □Computer Literacy □Business □Job-Related Applications □Household Management □Other ____________
14. **How often is your computer in use?** □Less than 1 hour per week □1-4 hours □5-10 hours □11-15 hours □16-20 hours □over 20 hours
15. **On the average, about how many program listings in each issue of HCM do you key into your computer and use?** □None □1 □2 or 3 □4 or more
16. **What peripherals do you currently use?** □Disk System □Printer □Modem □Monochrome/Color Monitor □Other ____________
17. **What do you expect to buy within the next year?** □Software □Disk system □Printer □Modem □Books □Magnetic Media □Monochrome/Color Monitor □Furniture & Accessories
18. **How much do you expect to spend on computer-related products during the next year?** □Less than $25 □$25-$49 □$50-$99 □$100-$249 □$250-$490 □$500-$999 □$1000-$2499 □$2500 or more

**OPTIONAL:** If you would like to help us by participating in a telephone interview, please include your telephone number (____) - ____________ here and the most convenient time you can be reached ______:______ □AM □PM


DON'T KEEP YOUR WINNING HAND A SECRET!
TELL 'EM ALL ABOUT HOME COMPUTER MAGAZINE, IT'S YOUR ACE-IN-THE-HOLE!

# COLLECT ALL BACK ISSUES

HOME COMPUTER™ magazine

*Please Print*

Name ______________________________

Address ______________________________

City ______________ State ____ Zip ________

□ Check or Money Order Enclosed **Total** ________
MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK

Bill my □VISA □MasterCard Date Expires ________

*Account No.* □□□□□□□□□□□□□□□□

Tel. No. ________ Signature ______________

Enclose payment or credit card information & mail with completed form to:

**Home Computer Magazine**
P.O. Box 70288 ● Eugene, OR 97401

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:
1-800-828-2212 (Minimum $10. Order)
In Oregon, Alaska, Hawaii Tel. (503) 485-8796
*PLEASE ALLOW 4-6 WEEKS FOR MAGAZINE DELIVERY.

| ITEMS | PRICE |
|---|---|
| Home Computer Magazine Back Issues*<br>(Circle Issues Desired)<br>Vol. 4, No. 1 Vol. 4, No. 2 Vol. 4, No. 3<br>Vol. 4, No. 4 Vol. 4, No. 5 Vol. 5, No. 1<br>Vol. 5, No. 2 | $3.95 each — U.S.<br>$5.95 each — Canada<br>$5.50 each — Foreign Surface<br>$7.50 each — Foreign Air |
| ON DISK & ON TAPE Back Issues<br>(Circle Issues Desired)<br>Vol. 4, No. 1 Vol. 4, No. 2 Vol. 4, No. 3<br>Vol. 4, No. 4 Vol. 4, No. 5 Vol. 5, No. 1<br>Vol. 5, No. 2 | $6.95 each — U.S.<br>$8.95 each — Canada<br>$10.95 each — Foreign Air |
| SAVE EVEN MORE—Order Combined Sets<br>(Circle Magazine & Media Sets Desired)<br>Vol. 4, No. 1 Vol. 4, No. 2 Vol. 4, No. 3<br>Vol. 4, No. 4 Vol. 4, No. 5 Vol. 5, No. 1<br>Vol. 5, No. 2 | $8.90 each set — U.S.<br>$12.90 each set — Canada<br>$11.90 each set — Foreign Surface |

Indicate your choice of media: (Check one)
ON TAPE™: □C-64 □TI-99/4A
ON DISK™: □Apple □C-64 □IBM PC □IBM PCjr □TI-99/4A

Defective media gladly exchanged. NO REFUNDS on media.

**For more information see inside front cover.**

Offer & Prices Subject To Change Without Notice.

# The Best Of 99'er —Book & Tape Set— Special Close-Out Offer

See Page 10 See Page 10

BEST
OF 99'er
VOLUME 1
FOR THE TEXAS INSTRUMENTS HOME COMPUTER
Articles & Programs for the TI-99/4A

*Please Print*

Name ______________________________

Address ______________________________

City ______________ State ____ Zip ________

**YES!** Please send me *THE BEST OF 99'er ON TAPE* along with my FREE copy of the book *The Best Of 99'er*. and the SPECIAL BONUS (while supplies last) of Simon's Saucer™ and the 99'er Programmer's Guide.

Enclosed is $35. (Shipping and Handling FREE in U.S.!) (Canada add $5.)

**For more information see page 10.**

Offer & Prices Subject To Change Without Notice.
Defective media gladly exchanged. NO REFUNDS on book or media.

□ Check or Money Order Enclosed

**Total** ________
MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK

Bill my □VISA □MasterCard Date Expires ________

*Account No.* □□□□□□□□□□□□□□□□

Tel. No. ________ Signature ______________

Enclose payment or credit card information & mail with completed form to:

**Home Computer Magazine**
P.O. Box 70288 ● Eugene, OR 97401

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:
1-800-828-2212 (Minimum $10. Order)
In Oregon, Alaska, Hawaii Tel. (503) 485-8796
PLEASE ALLOW 6-8 WEEKS FOR DELIVERY.

# Save $ $ $ On BACK ISSUES of 99'er HOME COMPUTER magazine™

## SPECIAL CLOSE-OUT PRICES FOR MAGAZINES, DISKS & TAPES NOW IN EFFECT FOR TI-99/4A USERS!

*Please Print* ORDERS LIMITED TO QUANTITIES ON HAND.

Name ______________________________

Address ______________________________

City ______________ State ____ Zip ________

□ Check or Money Order Enclosed **Total** ________
MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK

Bill my □VISA □MasterCard Date Expires ________

*Account No.* □□□□□□□□□□□□□□□□

Tel. No. ________ Signature ______________

Enclose payment or credit card information & mail with completed form to:

**Emerald Valley Publishing Co.**
P.O. Box 70288 ● Eugene, OR 97401

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:
1-800-828-2212 (Minimum $10. Order)
In Oregon, Alaska, Hawaii Tel. (503) 485-8796
*PLEASE ALLOW 4-6 WEEKS FOR MAGAZINE DELIVERY.

**For more information see inside back cover.**

99'er Home Computer Magazine, Disk, & Tape Back Issues
**Exclusively For The TI-99/4A Home Computer**

| ISSUE | Magazine Only | Mag. & Media Set | Media Only (Disk or Tape) | ISSUE | Magazine Only | Mag. & Media Set | Media Only (D'sk or Tape) | ISSUE | Magazine Only | Mag. & Media Set | Media Only (Disk or Tape) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Vol. 1 No. 6 | NOT AVAILABLE | | | Mar. '83 | | | | Aug. '83 | | | |
| Nov. '82 | NOT AVAILABLE | | | Apr. '83 | | | | Sept. '83 | | | |
| Dec. '82 | | | | May '83 | | | | Oct. '83 | | | |
| Jan. '83 | | | | June '83 | | | | Nov. '83 | | | |
| Feb. '83 | | | | July '83 | | | | | | | |

☛ INDICATE CHOICE OF MEDIA: □ DISK □ TAPE

Place an "X" in the corresponding box for each item you wish to order.

| | MAGAZINE PRICES* | | | MAG/MEDIA SET PRICES | | | MEDIA PRICES | | |
|---|---|---|---|---|---|---|---|---|---|
| QTY | U.S. | Canada | Foreign‡ | U.S. | Canada | Foreign‡ | U.S. | Canada | Foreign‡ |
| 3 | $6.95 | $9.95 | $8.95 | $14.95 | $19.95 | $18.95 | $10.95 | $12.95 | $14.95 |
| 6 | $11.95 | $17.95 | $13.95 | $28.50 | $37.95 | $34.50 | $20.95 | $22.95 | $27.95 |
| 12 | $21.95 | $29.95 | $25.95 | $49.95 | $59.95 | $56.95 | $39.95 | $44.95 | $45.95 |

Defective media gladly exchanged.
NO REFUNDS on media.

‡ Magazines Shipped to Foreign Countries via Surface

Offer & Prices Subject To Change Without Notice.

BCV4530485

Offer Expires June 30th 1985

# SPECIAL LIMITED-TIME GET-ACQUAINTED OFFER

SAVE $14.50‡ Over Single-Issue Price

Enter or Renew a 10-issue* ON DISK™ or ON TAPE™
Program Subscription For Only $45 and Receive
Your Choice of One Gift Shown Below

‡ Program subscription price increase goes into effect after June 30, 1985

*10 issues will be added on to any remaining issues you might still be entitled to because of a current program subscription or media given as premiums for subscribing to Home Computer Magazine.

## For Apple II Family, C-64, IBM PC & PCjr Users

### ON DISK Revue™ Volume 1

- ● 10 Programs On A Ready-To-Run Disk
- ● Exciting Activities For The Entire Family—At An Affordable Price
- ● Entertainment, Productivity & Education Plus Valuable Programming Secrets

An Attractive Library-Storage Slipcase

Valuable Proof-of-Purchase Coupon For Gift Redemption

CONTENTS
1. Boolean Brain
2. Electronic Home Secretary
3. Savings Planner
4. Snap-Calc
5. Tablut
6. Wild Kingdom
7. Musical Mystery Words
8. Market Madness
9. Tower of Hanoi
10. Missile Math

64-page Full-Color Book

Disk Stored in Re-usable Protective Pouch

**A $14.95 Premium Value!**

## For TI-99/4A Users

### Your Choice of One Gift

4 Tapes or 4 Disks from 1 Group Below:

| Group A | Group B | Group C |
|---|---|---|
| December, 1982<br>to<br>March, 1983 | April, 1983<br>to<br>July, 1983 | August, 1983<br>to<br>November, 1983 |

OR

6 Magazines from 1 Group Below:

| Group 1 | Group 2 |
|---|---|
| December, 1982<br>to<br>May, 1983 | June, 1983<br>to<br>November, 1983 |

OR

Best of 99er—The 360-Page "Classic" Reference Book.

**An $11.95 to $19.95* Premium Value!**

* 4 Tapes or Disks, $14.95; 6 Magazines, $11.95; Best of 99'er, $19.95

**Yes, please enter my 10-issue program subscription immediately. My computer type is:**

□ Apple *II* Family □ IBM PC □ IBM PCjr □ C-64 (disk) □ C-64 (tape)*

* NOTE: C-64 tape subscribers will receive as their premium, the disk-based ON DISK Revue Volume 1.

*Please Print*
Name ____________________

Address ____________________

City ____________ State ____ Zip ________

□ Check or Money Order Enclosed Total ________
MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK

Bill my □ VISA □ MasterCard Date Expires ________

*Account No.* □□□□□□□□□□□□□□□□

Tel. No. ____________ Signature ____________

Enclose payment or credit card information & mail with completed form to:

Emerald Valley Publishing Co.
P.O. Box 70288 ● Eugene, OR 97401

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:
1-800-828-2212 (Minimum $10 Order)
In Oregon, Alaska, Hawaii Tel. (503) 485-8796

**—TI-OWNERS ONLY—**
**Select Your Premium Below:**
**(Mark only one (1) box for your choice)**

**Either**

99'er Home Computer Magazine‡

| | ON DISK | ON TAPE |
|---|---|---|
| Group A | □ | □ |
| Group B | □ | □ |
| Group C | □ | □ |

**OR**

6 Issues of 99'er Home Computer Magazine‡
□ Group 1
□ Group 2

**OR**

Best of 99'er □

‡For editorial and program information, refer to the inside back cover of this magazine.

Canada add $5 for software subscription.
Software subscription not available in other countries at this time.

***Please allow 4-6 weeks for your first issue.***

Satisfaction Guaranteed—or the unfilled portion of your subscription will be refunded, less the cost of any premiums you have received. Prices Subject To Change Without Notice. *Defective media gladly exchanged. NO REFUNDS on shipped media.* PSV1530485

SAVE up to 40% off the Single-Issue Price Plus get Free Software To Boot!

# SUBSCRIBE NOW AND SAVE

Subscribe or Renew today, and with your paid subscription you will receive **FREE** software—**ON TAPE™** or **ON DISK™**. With a 1-year subscription to ***Home Computer Magazine*** you get **2 FREE Issues** (a $11.90 Premium Value) of **ON TAPE™** or **ON DISK™**. Subscribe for 2 years and receive **4 Issues** (a $23.80 Premium Value). And with a 3-year subscription we'll give you **6 full issues** (a $35.70 Premium Value) of this convenient software on cassette tape or floppy disk.

## Spread The Word . . . Introduce Us To A Friend.

PLEASE PRINT

Name ______________________________

Address ______________________________

City ____________________ State ______ Zip __________

**TOTAL** ______________

□ Check or Money Order Enclosed

**MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK**

Bill my □ VISA □ MasterCard

□□□□□□□□□□□□□□□□

Account No.

Phone No. ______________________

Signature ______________________ Exp. Date __________

**THIS SECTION MUST BE COMPLETED**

**INDICATE YOUR CHOICE OF MEDIA: (check one box only)**

**ON TAPE™:** □ **C-64** □ **TI-99/4A**

**ON DISK™:** □ **Apple *II* Family** □ **C-64** □ **IBM PC** □ **IBM PCjr** □ **TI-99/4A**

□ **Please send a raincheck for the FREE software.**

**YES!** Enter my subscription to ***Home Computer Magazine*** for the term below:
Please check one: □ New □ Renewal (include subscriber number below)

Label Number: ______________________

| | | |
|---|---|---|
| □ 1-yr (10 issues) $25 | □ 2-yr (20 issues) $45 | □ 3-yr (30 issues) $63 |
| ***PLUS 2 FREE***<br>ON TAPE or ON DISK | ***PLUS 4 FREE***<br>ON TAPE or ON DISK | ***PLUS 6 FREE***<br>ON TAPE or ON DISK |

**Canada add $11 per year; Foreign Surface add $21 for 1-yr magazine subscription. Free software offer available in U.S. & Canada only.**

Satisfaction Guaranteed—or the unfilled portion of your subscription will be refunded, less the cost of any premiums you have received.

Prices Subject To Change Without Notice

BCV1530485

Attn: Subscription Dept.

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

# BUSINESS REPLY CARD

FIRST-CLASS PERMIT NO. 25 EUGENE OR, 97401

POSTAGE WILL BE PAID BY ADDRESSEE

HOME COMPUTER™
magazine

P.O. Box 70288
Eugene, OR 97401

LIMITED SUPPLY

# A Back-Issue/Software Bonanza Of 99'er HOME COMPUTER magazine™ At Unbeatable Prices

The original 99'er Magazine and 99'er Home Computer Magazine were the forerunners of the present-day Home Computer Magazine. Each of these magazine back issues—exclusively covering the Texas Instruments TI-99/4A—is now available with your choice of either a floppy disk or a cassette tape that contains all the programs in that issue.

## 12 MAGAZINE & MEDIA SETS

These 2 Magazine Issues Are Out Of Print

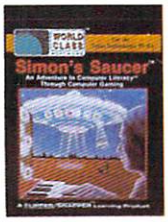
Tapes And Disks Are Still Available

**AS LOW AS $1.83 EACH MAGAZINE!**
See Order Card at Center of Magazine
**ABOUT $4 PER SET!**

**MAGAZINES OR MEDIA MAY BE ORDERED SEPARATELY**
To Order, Use Bind-in Card At Center Of Magazine.

## ✱ SPECIAL MIX OR MATCH BONUS ✱
### —WHILE SUPPLIES LAST—

**You will receive a FREE Simon's Saucer™ package and a FREE TI-FEST™ commemorative poster when your order includes at least 12 items from this page in any combination of magazines or media.**


WORLD CLASS
Simon's Saucer
An Adventure in Computer Literacy Through Computer Gaming
A FLIPPER/SNAPPER Learning Product

- A quality, ready-to-run game on cassette tape.
- A durable and attractive ring-binder collector's case for your software library.
- A complete, easy-to-use programming lesson on a deck of colorful flip cards.

A FLIPPER/SNAPPER™ Learning Product

IT'S A GAME!
Match Simon's tune, and he will beam down to reward you with a musical flourish and a colorful encounter. The more tones you match, the more spectacular your encounter becomes!

IT'S AN ADVENTURE IN COMPUTER LITERACY!
Each title in the FLIPPER/SNAPPER series contains a different lesson in programming. Newcomers to computing can start learning with *any* FLIPPER/SNAPPER—*there is no set order!*
In Simon's Saucer you will learn the secrets behind Simon's sight and sound effects. Soon, you too will be using CALL SOUND and CALL KEY to create this kind of World Class magic with your Home Computer.

**SAVE UP TO $44 AND RECEIVE FREE GIFTS WORTH OVER $15!**

**Issues No. 1-5 OUT OF PRINT**
**Contents available in book form as "Best of 99'er"—Vol. 1.**
See Home Computer Digest or Inquire

THESE 2 MAGAZINES OUT OF PRINT
—TAPES AND DISKS ARE STILL AVAILABLE—

**ISSUE #6 (Partial Contents)**
• How To Produce Sound Effects • Debugging a Game Program • How to Start a User's Group • Verbose: A speech Vocabulary Expansion Aid • Color Mapping • Dynamic Manipulation of Screen Character Graphics • The Beginner's Guide to Cassette Operation With the Home Computer • Pre-School Block Letters and Data Compaction • Picking the Ponies in TI BASIC • Battle Star Space Game • 3-D Animation on the Home Computer • Programming Tips • Who is LOGO for? • Tower of Hanoi in TI LOGO • A Review of the TI Lesson-Development Software • An Interview with a Game Designer • Learning Assembly Language with a Magic Crayon • and much, much more.

**NOVEMBER 1982 (Partial Contents)**
• Chatting with Your Micro: Languages for the Home Computer • A Review of the Smith Corona TP-1 Daisy Wheel Printer • The Micro Jaws Arcade Game • A Knight's Tour in TI BASIC • LOGO Has Style • ASPIC: A Language for Children • A p-System Beginners' Tutorial • An Interview with a p-System Pioneer • A Mini-Memory Screen Dump to the Home Computer Printer • Up Scope!—An exciting Undersea Combat Game • Strategy for Munch Man • A Brief Encounter with a TI Hand-Held Computer • 99'er Shopping Bus • A Pocket Battleship • Sub-Programs in Extended BASIC • Arcade & Adventure Game Reviews • and much, much more.

**DECEMBER 1982 (Partial Contents)**
• Tex-Scribe: A Text Editor for the Home Computer • A Christmas Computer Carol • Managing a Mailing List the Futura Way • Parsec: The Arcade Game • Plotting With the Home Computer—Pixel by Pixel • Preventing the Situation—Oh No! Memory Full • A Colorful Tour of TI-Fest: The Home Computer Show • Santa's Workshop: The Making of a Home Computer • The Turtle Arcade: Movies & Video Games in LOGO • Controlling a BASIC Termite • The 99'er Gold Rush—An Arcade/Adventure in the Home • 99'er Digest of News & Happenings in the TI World • Plus Games, Reviews, and much, much more.

**JANUARY 1983 (Partial Contents)**
• Computer Assisted Instruction for the Handicapped • p-System Basics • Debugging in LOGO • The Dow-4 Gazelle Flight Simulator • Note Whiz and Pitch Master Musical Game Reviews • Learning With the PLATO Computer Library • Strategies for Adventure Gaming • Death Drones • Using the Line-By-Line Assembler • Close Encounters of the Simon Kind • Electrical Engineering Education Program • Interview With an Arcade Game Designer • TI Invaders • Programming With Pascal • Cyber-Dice • News and Happenings in the Home Computer World • Arcade Game Reviews • The Thief Adventure Game • Programming Tips • and much, much more.

**FEBRUARY 1983 (Partial Contents)**
• Texas Instruments at the Winter Consumer Electronics Show • Home Computer Printers on Review • How to Create Math Daisies in LOGO • Vectors in LOGO • ASPIC: A Language for Teachers • The Joys of Adventuring—Part 2 • Programming Pointers with Chuck-A-Luck—Part 4 • Interview With the Voice of Parsec • Why You Need a Printer for Your Home Computer • Lifeline to Titan Space Game • Night Blockade Battleship Game • Tower of Hanoi Pocket Program • Computer Gaming Software Reviews • News of Late Developments in the World of Home Computers • and much, much, more.

**MARCH 1983 (Partial Contents)**
• An Introduction to the TI-99/2 Basic Computer • The Hex-bus and the 4/A Connection • Making Your Own Say and Spell Game • Disabled Children Learn and Grow • Super Cataloger—A Review of a Disk Library Utility Program • TI's New CC-40 Compact Computer • Robots and Their Social Impact • Twenty Questions With Robot Redford • The Gravity of LOGO • Joystick Jockey—An Overview of Remote Controllers • Parsec Strategy • Converting Extended BASIC to Assembly Language • Matrix Muncher • Mini Memory Disassembler Utility • Pulling the Shade on Sprites • Letters on LOGO • Tiny Tutorials • Games programs, reviews, and much, much more.

**APRIL 1983 (Partial Contents)**
• Computer Assisted Savings Planning to Build Your Nest Egg • Tex-Cipher Writes and Decodes Secret Messages • Crossbytes—Computer Vocabulary Crossword Puzzle • Cutting Corners On Your Food Budget Using Coupons • Introducing Financial Planning with Multiplan • The Design Philosophy of the Compact Computer • LOGO Takes On the Popular Fifteen Puzzle • Super Language—Programming Sprites in Mini Memory • Colorful Word World—Reading Readiness for Pre-schoolers • Gameware Buffet's A-Maze-ing Boa Alley Game • Giant and Dwarfs Entrapment Game • Game Reviews • Programming Tips • Money Saving Hints • and much, much more.

**MAY 1983 (Partial Contents)**
• A Consumer's Guide to Word Processing • Word Processing Market Basket • A Generalized Filing Program for VIPs • The Multiplan Medium Balances Your Checkbook and Budget • Activity Accountant Helps School Secretaries with Extracurricular Activities • Maximizing Your Mini Memory's 4K of RAM • Exploring Enhanced BASIC on the Compact Computer • The LOGO Tortoise Debates the BASIC Hare • A Pocket Program to Organize Data with Linked Lists • Mentally Handicapped Learners Team Up with the TI-99/4A • The Wonders of Diskette Storage • Beeline—a Multi-Screen Strategy Game • Lost Ruins—an Archeological Dig Game • 3-D Illusions with Sprites in Depth • Game reviews, Group Grapevine, and much, much more.

**JUNE 1983 (Partial Contents)**
• Children and Computers Make the 99/4H Connection • Tune Your Guitar with Our TI Tuning Fork • Talk to Your Computer—Voice Technology Is Here • Gameware Buffet's Eat or Be Eaten Aardvark Game • Protect Your Station in the Space Zapper Game • What Multiplan Can and Can't Do • Understanding Inputs and Outputs in Drive For Diskettes—Part 2 • Calculate Loan Schedules on the CC-40 • Go on a LOGO Vacation • Letters on LOGO • A Review of Upper Room Software's Programs for Special Learners • Construct an RS232/Joystick Interface • Group Grapevine • Shopping Bus • A Natural Language Interface for the Professional • Game Reviews • and much, much more.

**JULY 1983 (Partial Contents)**
• The Evolution of Home Computer Graphics Comes Alive in Graphics Grows Up • Five Data Organizers in Never Out of Sorts • TI & 99'er at the Consumer Electronics Show • WarGames: The Movie and the Book • Editing with Multiplan • The LOGO Logician Presents To Model Is to Learn • LOGO Mosaic Designs Fill the Screen • Your Speech Synthesizer as a Spelling and Foreign Language Teacher • Software for Your Low-cost Printer Port • Gameware Buffet's Treasure Island and the Colorful Switch-A-Row • A Book Review of Learn BASIC for CC-40 Users • 3-D Animation with the TMS 9918A Video Chip • Games Reviews • Group Grapevine • and much, much more.

**AUGUST, 1983 (Partial Contents)**
• The Home Computer Goes To Work • Bit One, Purl Two at the Fashion Factory • Better Business Bar Graphs in Graphic Persuasion • An Ensemble of Assemblers • Cashflow Helps Money Management • Keystrokes for Thrifty Folks—a Review of Typwriter • Game Reviews of Cavern Quest and Starprobe '99 • Counting Fun for Preschoolers • Peripheral Vision 99 • Mean Machines and Small Potatoes • Multiplan Medium Groups Cells into Rectangles • Turtle Text, a LOGO Word Processor • Group Grapevine • Hello, Little Brother—CC-40 Speaks to 99/4A • Plato's Progress Looks at Student Assessment • Pocket Sunrise • Public Investigator • Gameware Buffet's Jungle Jim, Success Formula, and much, much more!

**SEPTEMBER, 1983 (Partial Contents)**
•Adding on to Your Home Computer • 99'er Directory of Commercially Available Software, Accessories and Peripherals • Peripheral Vision 99 Hardware Reviews • Pocket Sunrise, Part Two in Extended BASIC • Byte Lightning—Mini Memory Plays a Game • TI-WRITER Tutorial • LOGO's Functions, Sets and Turtles • The CC-40 and 99/4A Take the Data, and RUN • PLATO's Progress Looks at Geometry Courseware and the Shape of Things to Come • Gameware Buffet's Challenge of Camelot and The Fly • Game Reviews of Nit Wit and Crime and Punishment • Group Grapevine • 99'er Hall of Fame • 99'er Digest Update on New Products, and much, much more!

**OCTOBER, 1983 (Partial Contents)**
• Adventures and Fantasy with Your Home Computer • Do-It-Yourself Adventure • Picoprocessor Emulator, a 4-Bit Microprocessor • Have No Fear: Assembly Language Won't Byte, Part 1 • Make Your Mark • Les Izmore and Debug Computing Cartoon • Once Upon a Tortoise Shell—A LOGO Adventure • Turtle Strut • Number Nibbler for Children • Lots of Plots on Your CC-40 and HX1000 Four-Color Printer/Plotter • Multiplan Medium—Bartender • A Grisly Adventure Bear Hunt • Escape From Wizard's Keep in Extended BASIC • Game Reviews • Shopping Bus • and much, much more.

**NOVEMBER, 1983 (Partial Contents)**
• Education with your Home Computer • Five Creative Learning Activities for Children • Let's Build America • Have No Fear: Assembly Language Won't Byte, Part 2 • Squeezing the Most Out of TI BASIC • TI-WRITER Tutorial • LOGO Lexicon • Interview with Dale Osborn • TI-WRITER: At Home In the Office • PLATO's Progress • Les Izmore and Debug • The Multiplan Medium • Computer Arrested Instruction—99'er Interviews the Kids • Gameware Buffet: Taco Man and Robo Chase • Game Reviews of Jail Break and Arithmetex • Hall of Fame • 99'er Digest • and much, much more.

**Hurry—Supplies Are Limited. Order Yours Today!**

Offer & Prices Subject To Change Without Notice.

# ALL PROGRAMS IN THIS MAGAZINE

HOME COMPUTER™ magazine
ON™ DISK
ISSUE
□APPLE □IBM □COMMODORE □TI
Copyright © 1984, Emerald Valley Publishing Co.
ISSUE
FORMAT: □C-64 OTHER: □TI-99/4A
Copyright © 1984, Emerald Valley Publishing Co.
SIDE A
HOME COMPUTER™ magazine
ON™ TAPE

# ONLY $4.95* ON DISK OR TAPE!

The same high-quality Apple, Commodore, IBM, and Texas Instruments programs with type-in-and-RUN listings in this issue are now available ON DISK™ or ON TAPE™ to newsstand purchasers or subscribers of this magazine.

For only $4.95* plus $1.00 for shipping & handling, you receive all the programs for your particular brand of computer—rushed right to your door by First-Class mail. —Truly A "Software Giveaway!"

To Order, Use The Bind-In Card Inside Rear Cover. Offer & Prices Subject To Change Without Notice.

* Current Single-Issue Price Only — See Center Bind-In Card For Back-Issue Prices & Special Program Subscription Offer.