

# HOME COMPUTER<sup>TM</sup> magazine

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

Vol. 5 No. 2

\$3.50 in USA  
\$4.50 in Canada

## Number Crunching

—The Building Blocks  
Of All Computing

**Featuring A Ready-To-Use  
Numerical Problem Solver for:**

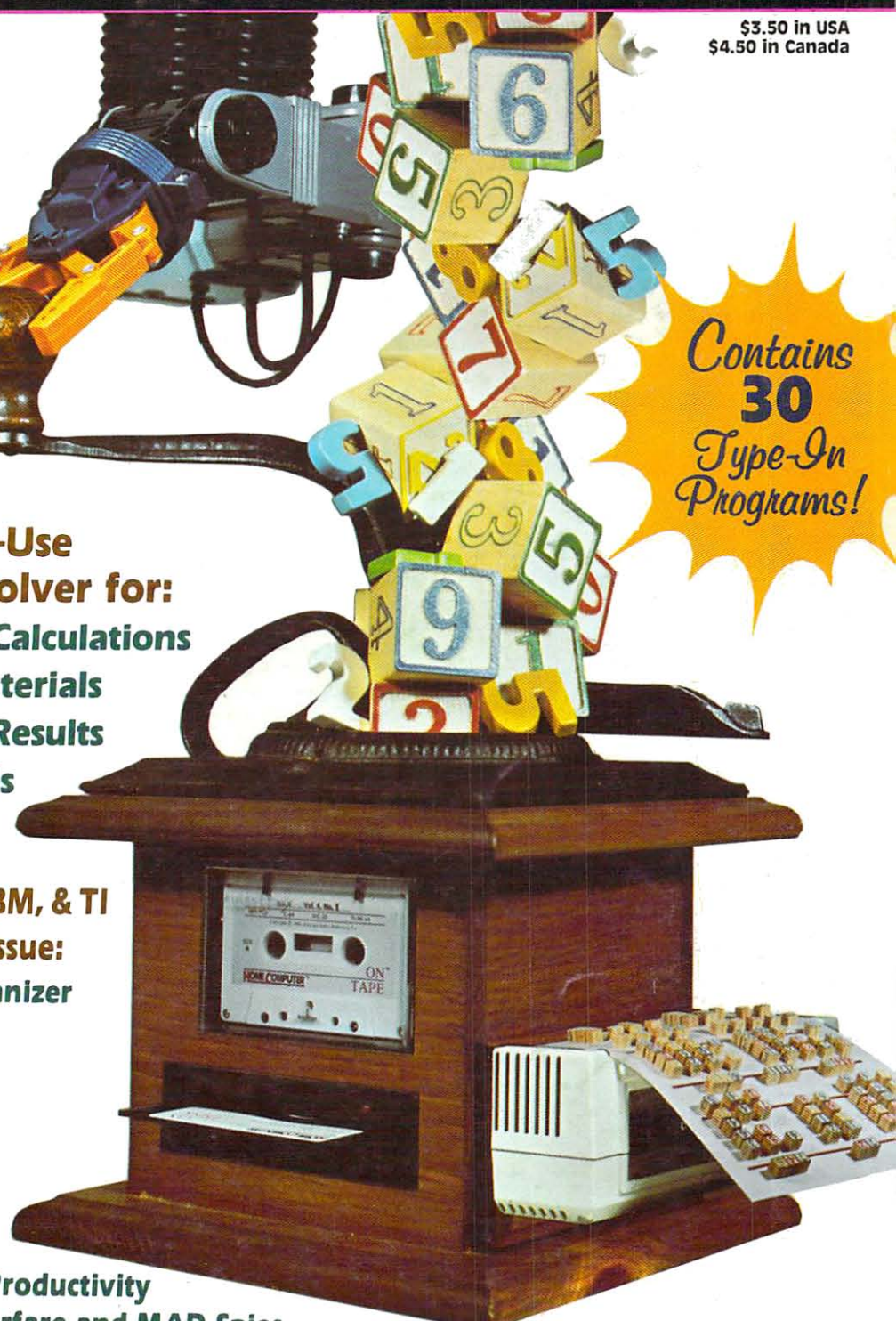
- Performing Complex Calculations
- Estimating Cost & Materials
- Projecting Financial Results
- Enhancing Math Skills

**Apple, Commodore, IBM, & TI  
Software Programs in this Issue:**  
• The Organizer  
• The Flight Simulator  
• Irony Anagrams  
• Arithmetic Edu-Gaming

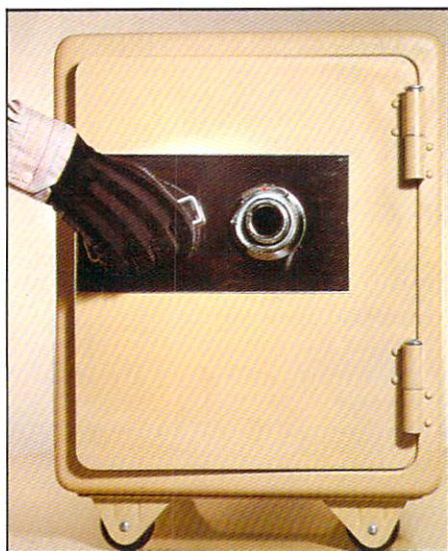
**—Reviews Galore:**

- ★ Triple-Threat Productivity
- ★ Creative Warfare and MAD Spies
- ★ Computerized Driver-Ed
- ★ Musical Keyboards for Home Computing
- ★ A Re-Birth of CP/M Add-Ons

*Contains  
**30**  
Type-In  
Programs!*



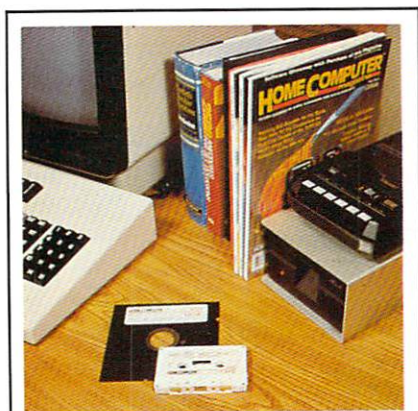




# MISSING ANY VALUABLES?

If you're missing any back issues of **HOME COMPUTER**<sup>™</sup> 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<sup>™</sup> for your home computer.

Back issues of HCM's program service—**ON DISK**<sup>™</sup> or **ON TAPE**<sup>™</sup> are also available.



**ON DISK**<sup>™</sup> and **ON TAPE**<sup>™</sup> 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.





# The Plain & Simple Truth About **HOME COMPUTER**<sup>™</sup> 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.

Each separate contribution to this March 1985 issue and the issue as a collective work is Copyright © 1985 by Emerald Valley Publishing Co. All rights reserved. Copying done for other than personal or internal reference use without the permission of Emerald Valley Publishing Co. is prohibited. Requests for special permission or bulk orders should be addressed to the publisher.

**Limited License for use of programs in Home Computer Magazine.** Emerald Valley Publishing Co. (EVP) is the owner of all rights to the computer programs and software published in this magazine. To allow for use of the software by the purchaser of the magazine, EVP grants to such purchaser only, the limited license to enter these programs into the purchaser's computer, and to place such programs on a diskette or cassette for the purchaser's personal use.

Any other use, distribution, sale, or copying of these computer programs without the written consent of EVP is expressly prohibited and in violation of this limited license and the copyright laws.

Home Computer Magazine, HCM, and Home Computer Digest are trademarks of Emerald Valley Publishing Co.

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

Coffee grinder provided by The Treasure Market (Eugene, OR).



## Outside HCM

As the cybernetic hand turns the crank, numbers spill and crunch into manageable bytes. Everything—be it conceptual, visual, or audible—is reduced to numbers in today's electronic grinder, the computer. Thus, as the mighty hand of software processes these numbers into useable tools (such as those found in the pages of this magazine) we wind our way from the Mechanical to the Information Age.

## Inside HCM

Cccrrrrunnnnnnnchhhh! It's the sound and fury of numbers crunching—although you don't really get to hear them as they grind away inside your computer. Numbers are to computers what Rice Crispies are to a pack of hungry kids: *delicious!* In fact, a computer prefers pure numbers to any other fare, happily chewing its numerics morning, noon, and night.

To see this, you need only pick up our featured software package in this issue: *It Figures!* Give this handy program a bunch of values and a formula to go by, and it will crunch out an answer quicker than you can take a byte of your cereal. Then let the youngsters do some number-crunching of their own with *Laserithmetic*, an educational game pitting fast math against a pack of pesky aliens.

A computer can also turn numbers into letters—shuffling and dealing alpha characters like a stack of marked cards. In *Switch 'n' Spell*, another software program in this issue, your challenge is to take a "hand" of letters dealt by the computer and rearrange them into a bonafide word.

Some of our programs start anew, and others enhance some previously developed. Last issue, we premiered a new program, called *The Organizer*, that has been helping you (we hope) keep your thoughts in order. This time, with *Organizer Reports*, you can put those organized thoughts on paper—and do it with a variety of formatting options.

Once you're organized, be sure and check out our four "mini-columns"—

one for each brand of computer we cover—for some quick-and-easy practical software procedures.

Providing software is "numero uno" at *Home Computer Magazine*; but we know our readers also count on our reliable reviews for the "true story." In this issue, we take a broad but unsupercilious look at *The Music of Sound* on the Commodore 64. This article explores the new software for turning the C-64 into a home organ/synthesizer by using its amazing sound chip. This should be of interest even to those who own other machines, because of the general trend these musical programs portend for the future of home computing. (Be sure to catch the first installment of our *Commodore Hornblower* column, which begins a series of modules to build a BASIC synthesizer.)

Among our other reviews, we delve into the world of CP/M, examining CP/M packages for the Apple IIe, and TI-99/4A—as well as take a brief look at CP/M on the PC and PCjr and the new version 3.0 running on the forthcoming Commodore 128. Apple users who want to get some real work out of their Apple IIe or IIc will benefit from our review of *AppleWorks*, an integrated word processor, database, and spreadsheet package. PCjr owners wishing to further enhance their machines can look at the *Legacy II PCjr Expansion*, which adds a disk drive and more memory to "Little Blue."

All this and more adds up to quite a number. You can always count on us at *Home Computer Magazine* to turn number-crunching to your advantage.

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

HCM



**By Gary M. Kaplan**

*Publisher & Editor-in-Chief*

*"... each of you is invited to mail in a written response to this column, communicating your ideas..."*

One of the benefits of publishing a magazine without outside advertising is the total editorial and artistic freedom such a format permits. You have already had the opportunity to see several issues in this format and to discover some of what's possible. So now it's time for all of us to sit down, take stock, and fine-tune this approach. With this goal in mind, each of you is invited to mail in a written response to this column, communicating your ideas on how this magazine can better serve your needs.

We have decided against using preprinted forms or questionnaires for this response, and instead are requesting a more "free-flowing" type of feedback. Preprinted forms may be easier to quantify and analyze, but we wish to gain a better "feel" for your reactions than a purely statistical approach can provide. To make this formidable job easier—saving us from having to wade through what could be tens of thousands of multi-page letters in all shapes, sizes, and forms—we hope you will stay within the spirit of the following guidelines:

1.) All comments should be typed or computer-printed using a dark ribbon on one sheet of 8-1/2- by 11-inch paper. Print on one side only and leave us ample room to make notes in the margins. PLEASE DO NOT ENCLOSE ANYTHING ELSE IN YOUR ENVELOPE EXCEPT THE ONE SHEET OF PAPER.

2.) On the top of the sheet of paper, clearly indicate your brand of computer(s), how long you have been a reader of our magazine, and whether you are a subscriber, a single-copy purchaser, or a pass-along reader. Indicate whether you type in programs, buy the media, or neither. Please also include your age (approximations are okay if you're sensitive about it) and see that your name, address, and telephone number are present.

3.) Please respond within two to three weeks of receiving this issue. All mail is subject to the same provisions as "Letters to the Editor" (as set forth on the Masthead page).

4.) Try not to be too wordy. Short phrases, an outline form, or brief notes are all appropriate. We're not looking for perfect essays or grammatically correct sentences. Remember, though, to be as specific as possible. Don't forget to fully explain your opinions.

5.) Please leave out any problems that are of a "customer service" nature—i.e., timeliness of receiving merchandise, issues, etc. The purpose of your response here is to comment on the *content* of the magazine.



6.) Things we'd especially like to know include impressions of each of our regular features or sections—whether they're of use, what you'd like to see eliminated, expanded, and/or replaced with what, and what types of programs you'd like to see in the future. Would you like more programs, less programs, more or less space in the magazine taken up with program listings? You can comment on the length of our features. Also comment on how we review products. Are we doing enough different product reviews... are we doing too much... are they too short, too long? How do you feel about our review criteria? Should we cover other programming languages in our articles? Are some of our articles too difficult? Too easy? Why? What are they?

We know we're asking you to think a lot here, but it's *your* magazine, and we want to produce what you want—so let us know. To make things a little more interesting, we are going to select the best constructive response, and award that person a prize—a free trip. No, it's not to Hawaii, Europe, or the Caribbean... but to none other than Eugene, Oregon—to show the winner first-hand how we put together the magazine, and to discuss his or her suggestions. So stick your single sheet in an envelope plainly marked **ON SCREEN FEEDBACK** and address it to Home Computer Magazine, P.O. Box 70288, Eugene, OR 97401.

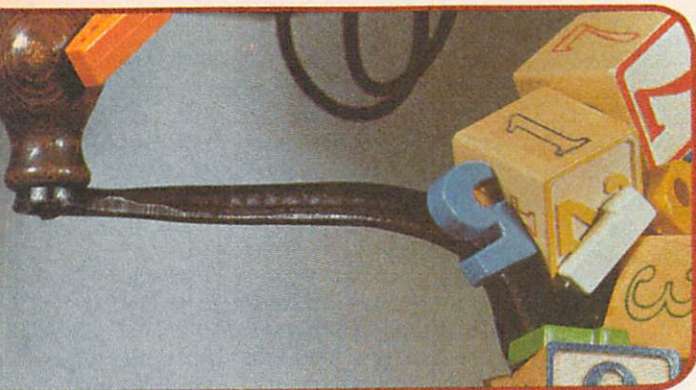
The feedback we receive will be acknowledged in this column as soon as possible, so you'll be able to find out exactly what other readers think—and as a result, what we are going to be implementing.

Starting with our next issue, we will be undergoing a major expansion in distribution. You'll be able to find us at many more mass-market magazine outlets and bookstores throughout the U.S. and Canada. This is an important action—ensuring that the widest possible audience of computer users are aware of our commitment to excellence, thus fueling our future growth. And as we grow in circulation, we will also strive to grow in quality. Your help and guidance will make this all possible. So become an active participant in this process—send in your constructive ideas; "show 'n' tell" us to your friends, fellow computer users, and favorite magazine/software outlets; and, above all—

*Come Grow With Us.*



# HOME COMPUTER<sup>TM</sup> magazine



## FEATURES

### 14 It Figures!

What does it do? How does it do it? It Figures!



by Robert Paschelke  
and the HCM Staff

### 18 Evacu-Pod

Pilot your Evacu-Pod to rescue miners on 4 other worlds.



by William K. Balthrop  
and the HCM Staff

### 21 Switch 'n' Spell

A spelling aid . . . with an entertaining twist.



by Randy Thompson  
and the HCM Staff

### 24 Laserithmetic

Will math skills and laser blasts keep alien beasts at bay?



by R. G. Christensen  
and the HCM Staff

### 26 Organizer Reports

Flexibility and organization highlight your outline printouts.



by William K. Balthrop  
and the HCM Staff

### 38 Razzle Dazzle

Don't just play with your 99/4A, play it, maestro.



by William K. Balthrop

### 43 What Is CP/M?

Is this operating system headed for rebirth?



by the HCM Staff

### 48 Apple Seedlings

Sort your ProDOS catalogs.



by Hendrik Broekhoff

### 54 Commodore Hornblower

Inside the SID chip.



by Roger Wood

### 62 IBMpressions

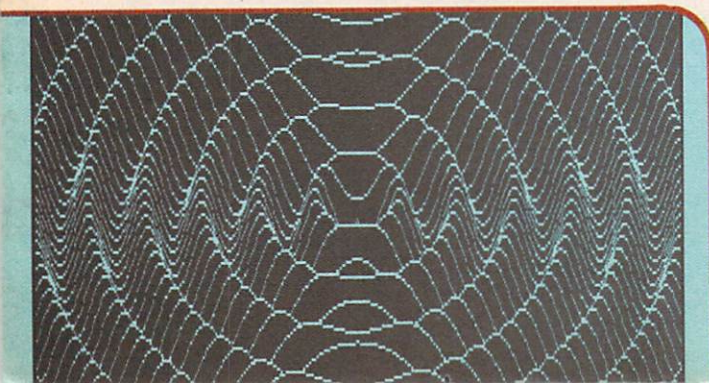
Create 3-D surface drawings with BASIC.



by William K. Balthrop

### 78 Field & Screen: Using a Data Base System

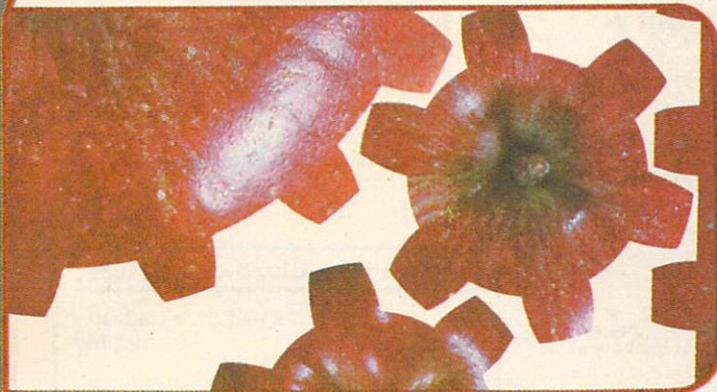
A general introduction to using data bases—correctly. by Bill Crouch





# CONTENTS

VOLUME 5 NUMBER 2



## PRODUCT REVIEWS

- 32 The Music of Sound**  
**A Review of 4 Music/Sound Software Series with Keyboards for the C-64**  
Creative fun with Commodore's amazing sound capabilities. *A Review*



- 40 Lost in CP/M Land:**  
**A Review of the**  
**Microsoft Premium Softcard IIe**  
Are we being too hard on the Softcard? *A Review*



- 44 A CP/M Dawn for the TI-99/4A**  
**A Review of**  
**Morning Star's CP/M Package**  
Finally, the CP/M window is opened to 99/4A users. *A Review*



- 47 Spy vs Spy**  
Does the software capture the comic MADness? *A Review*



- 50 AppleWorks**  
An easy-to-use integrated package reaches the home. *A Review*



- 56 The Ancient Art of War**  
Here it's okay to pick a fight. *A Review*



- 58 Legacy II for the PCjr**  
Junior continues to grow in power. *A Review*



- 61 The Factory**  
Punch, stripe, and rotate objects in your own factory. *A Review*

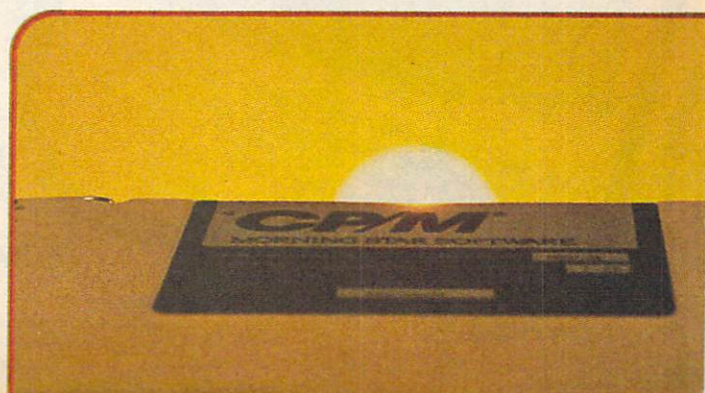


- 66 Keys to Responsible Driving**  
Better than a driver's manual? *A Review*



## DEPARTMENTS

- |                                |                                    |
|--------------------------------|------------------------------------|
| <b>3 Welcome to HCM</b>        | <b>83 Program Listing Contents</b> |
| <b>4 Inside/Outside HCM</b>    | <b>128 Debugs on Display</b>       |
| <b>5 On Screen</b>             |                                    |
| <b>9 Letters to the Editor</b> | <b>Home Computer Tech Notes:</b>   |
| <b>13 HCM One Liners</b>       | <b>72 Apple</b>                    |
| <b>31 HCM Review Criteria</b>  | <b>73 Commodore</b>                |
| <b>64 Industry Watch</b>       | <b>74 IBM</b>                      |
| <b>67 HCM Product News</b>     | <b>75 TI</b>                       |
| <b>82 Program Typing Guide</b> |                                    |





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

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

# THE BEST

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

OF **99'er**™

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

VOLUME 1

- 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 \$19.95 + \$3.00  
(Book alone) shipping

Best Of 99'er \$35.00 + \$2.50  
(Tape Set alone) 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**

**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**

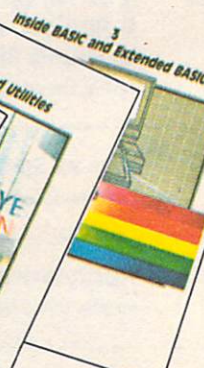
**PLUS**

2  
Programming Techniques and Languages

4  
LOGO

**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.

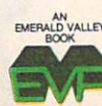
TO ORDER—USE BIND-IN CARD  
AT CENTER OF MAGAZINE

People Who Know the Home Computer Best™

**HOME COMPUTER**™  
magazine

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

### Program Version Confusion

Dear Sir:

I have two questions for you, but first I would like to say I enjoy your magazine very much and I am very glad to see that you are continuing to give good coverage to several home computers—especially the TI-99/4A. Thanks again for the great programs for free—I am not a subscriber, but I intend to become one the first of the year. Now for my questions:

1. In the August 1984 issue you ran the program Snap-Calc for the TI-99/4A (version 4.3.1). In issue Vol. 4, No. 5 you ran some "DeBugs on Display" for Snap-Calc for several computers. The Snap-Calc was version 4.3.3. What happened to 4.3.2? Also, several of the other debugs were not 4.3.2—Apple was 4.3.4, IBM 4.3.5, and Commodore 4.3.5. I know I did not miss any issues. Is this a misprint?

2. If I purchase ON TAPE for August 1984, will it be revised or will I have to edit it myself from the "DeBugs on Display" column?

G. Preuss  
Bridgeport, CT 06606

*The answer to your second question is shorter, so we'll talk about that first. At any given point in time that you order ON DISK or ON TAPE media, you will receive the latest version of all programs on that media. If more debugs are discovered and published after your purchase, those debugs will have to be merged into the program yourself.*

*Now, the answer to your first question: There was no misprint in the version numbers for the various machines. It is entirely possible for software to be revised several times between the publication of any two issues of Home Computer Magazine. In other words, between the time that Snap-Calc was originally published in the Vol. 4, No. 3 issue until the debugs appeared in Vol. 4, No. 5, the TI-99/4A version of Snap-Calc underwent 2 revisions, while the Apple program underwent 3 revisions, and the IBM and Commodore programs underwent 4 revisions. The debugs for Snap-Calc as listed in Vol. 4, No. 5 include all changes necessary to bring up-to-date the .1 version of each of the listings.*

*Starting with this issue of the "DeBugs on Display" column, we will include a line in each debug update listing that will tell you the previously published version number. For instance, if there are debugs for a version of Snap-Calc in this issue, they will only include*

*the changes that have occurred since the last time debugs for Snap-Calc were published.*

*Even though you have a fully functional version of a program, be sure to watch "DeBugs on Display" for changes to that program. Often, readers and HCM staff come up with significant enhancements that get published in that column . . .*

### Once She Compared . . .

Dear Sir:

I just had a friend loan me 3 copies of your magazine. Since I am a relative newcomer to the ranks of "computer owner," I really needed a magazine with good, usable information. I had bought off the newsstand several other magazines to get a feel for each one and to see how well-suited each was to my level of experience as a neophyte Apple owner/user.

Your magazine beat all the magazines catering to Apple owners. It is by far the best magazine I have ever seen for the home computer user anywhere.

Paula Farrell  
Williamsville, NY 14221

*Thanks, Paula. Why don't you tell your friends about us too . . . The larger the circulation, the stronger we become, and the better we are able to serve our readers.*

### . . . There Was No Comparison

Dear Sir:

This is my first time writing to this or any magazine. I wouldn't write unless I felt strongly about something. I first heard about your magazine when I bought my TI-99/4A. I recently bought a C-64, but I won't change my subscription to a Commodore-only magazine. You seem worried about losing some readers to one-computer magazines, but I think you have little to worry about. During the time that it took to convert your magazine, I did not receive any magazines for quite awhile. I called your telephone number, and the operator told me why it was taking so long.

Impatient as I was, coupled with the fact that I had bought a new computer, I decided to try a few other publications. I was placated. When I finally received my issue of Home Computer Magazine, though, I breathed a sigh of relief. It was like coming home. Your magazine is head and shoulders above the rest.

Since I realized how much I enjoy your magazine, especially since the ads have been removed, I have started recommending HCM to all of my friends.

Not only am I recommending HCM, I carry a subscription page with me, and give it to anyone who seems interested.

Please keep up the good work and feel secure in the knowledge that if you don't improve your magazine for five years, no one will threaten you.

David Sheldon  
Jacksonville, FL 32205

*That's quite a story of loyalty, David, and we appreciate it. We appreciate it so much so that we will not just sit on our laurels over the next five years and wait for other magazines to catch up. We will continue to improve Home Computer Magazine—to not only stay on top, but to increase the distance between us and our competitors.*

### TI Fitness Programs Stocked

Dear Sir:

As a footnote to your product source information in the recent article on fitness software in Home Computer Digest, three excellent fitness and nutrition programs are available for the TI-99/4A from Tex-Comp:

The TI Physical Fitness module leads the user through a complete exercise program with a graphical demonstration of each exercise; the TI Weight Control & Nutrition module actually works out a daily diet to reach a desired weight and comes with a recipe book; and Exer-Log is an all new database-type program which keeps track of your exercise activities and calculates calories burned.

Jerry Price  
Tex-Comp/P.O.Box 33084  
Granada Hills, CA 91344

*Thank you, Jerry, for your update on the software availability.*

### Simon's BASIC Cartridge

Dear Sir:

I am requesting information about an article on page 45 of Vol. 4, No. 4 called "Simon Sez." I am very interested in receiving information about the Simon's BASIC cartridge.

Nancy L. Taylor  
Alvin, TX 77512

Continued on next page



# Letters

to the Editor CONTINUED

A Simon's BASIC cartridge and manual is available through your Commodore dealer, Nancy, for a suggested retail price of \$34.95. This cartridge plugs into your Commodore 64 and adds 114 additional commands and functions to Commodore BASIC, making the C-64 much easier to program.

## PCjr Second Drive Problem

Dear Sir:

I would like to address a problem I encountered when attempting to add a second drive to my PCjr.

After adding the modifications to the Junior's disk controller card, I got repeated error H's when the system ran its self-tests. At this point, I set up a digital multimeter to monitor the drive select signal for drive 0. I noticed that the select line was being pulled to approximately .9v, which is not a valid logic false level. After checking the data book for the specs on the 74LS10, I decided that the chip might not be able to sink enough current to activate the drive. At this point, I changed the 74LS10 to a 7410 device. From that time on, the system has worked fine.

I would like to add that I contacted Mark Beifuss after reading in *HCM* that he too was encountering the Error H problem. He too made the change to the 7410 and phoned me to say that it also fixed his problems.

I had almost written off purchasing the Junior because of the need and expense of purchasing the special equipment required to add the second drive. After reading your article, I decided to purchase the Junior and I am totally pleased with the machine. Keep up the good work.

Randall Baxter  
Garland, TX 75040

Interesting point, Randall. The systems that we have in operation at Home Computer Magazine do use the "LS" chip made by Texas Instruments. Perhaps the specifications between manufacturers of the same chip are somewhat different. The LS (low-power Schotky) series of integrated circuits was selected because of its noise immunity and low power consumption. In this particular application, the 7410 does work better as a "line driver" and may solve problems for people having trouble with the modifications. For anyone that is having a problem with their conversion, we suggest replacing the 74LS10 with the 7410 integrated circuit as Randall suggests.

In addition, the kit that Home Computer Magazine offers will substitute a 7410 in place of the 74LS10 in the future. (The kit includes the special cable required and the two integrated circuits for \$49.95. Refer to this kit as the PCjr Disk Drive Kit.) If you have already purchased the kit from Home Computer Magazine and are having the problem described above, let us know via a letter and we will mail you a 7410 integrated circuit at no cost.

## POKEing Around Mac

Dear Sir:

I am the owner of an Apple IIe and an Apple Macintosh. To protect LISTing on the IIe all I need to do is type in POKE 1011,0:POKE 214,255. It prevents anyone from pressing [CRTL] [Reset] or [CRTL C] and LISTing the program. Neither of the POKEs work on the Mac. What is needed to do the same thing with the Mac? Do you know of any other POKEs that would be helpful?

I live in Arlington, Washington and have yet to see a Home Computer Magazine on a newsstand. I don't know why, because it is the best computer magazine I have ever read.

Dan Stuart  
Arlington, WA 98223

Although we use the Macintosh on a daily basis in our Editorial Department, Dan, we have yet to "get inside the box." Perhaps another reader that is a "Mac hacker" will write to us with an answer to your question.

Starting with our next issue, you will be able to find Home Computer Magazine in many more mass-market type outlets. This major expansion in our distribution has been delayed until now because of many changes we've undergone to "arrive" at our new format and frequency.

## Braille Processing Needed

Dear Sir:

I need assistance in locating any information that is available on a braille reader and a braille typewriter that could be interfaced into the Commodore 64 home computer. The interfacing of the two would not be a problem, but finding a 6-point strike head printer seems to be impossible.

Brad M. Johnson  
Marengo, IL 60152

We don't have the answer to this one for you, Brad, but it sure sounds like a good question to submit to our readers. How about it out there? Anyone know of a braille typewriter and reader that can be interfaced to a home computer?

## TI Short-Wave Links

Dear Sir:

I would like to combine my hobbies but I don't know how. I would like to connect my TI-99/4A to a short-wave receiver which is tuned to receive teletype signals from news service transmitters. I would like the computer to interpret and present the teletype signals as text on my monitor.

If an article about this subject has been written, I would appreciate it if you would send me the date of the magazine. Otherwise, this might be a suitable subject for an article.

I.J. Kenner  
Severn, MD 21144

Yes, we thought that was a good idea for an article too. Please refer to the Vol. 5, No. 1 issue of Home Computer Magazine, page 43, where you will find an article entitled, "Computer Links to Amateur Radio" which covers several combination hardware/software solutions to the reception and conversion of short wave signals to on-screen text for home computer owners. You will find information applicable to IBM, Commodore, Apple, and TI computers in this article. One additional note, which is repeated in the article itself: Most large news organizations that transmit via radio teletype encode their information so that it is only decipherable by subscribers to their news service.

## Axiom Delivers a Lemon

Dear Sir:

I bought a GP550TI printer from Axiom Corporation because of the article in your magazine.

The print is bad, to say the least. And, as you can see from the enclosed memo from Axiom, "What you see is what you get."

I wish your article would have been more detailed about the print before I put down my hard-earned money.

Let others know just how bad the printer is—if you can stand the heat.

G. Jester  
Warner Robins, GA 31093



# Letters

## to the Editor

CONTINUED

We are sorry, Mr. Jester, that you have been having problems with your Axiom printer, but we wish to point out that the printer you purchased is a different model from the one that we reviewed. We reviewed the GP100 TI II, not the GP550TI. We talked to the factory about your printer problems. They recommended that a lighter weight paper, such as 12- to 15-pound bond, be used rather than the 20-pound bond that your letter was printed on. They also suggested the use of a high-quality ribbon to improve the print quality. We hope this helps to get you better results.

### King's Quest Completed on Jr.

Dear Sir:

I read your article in the December issue of *Home Computer Magazine* about King's Quest. I am writing to inform you that we at Shillito Rikes Computer Depot in Cincinnati's Tri-County Mall have solved the popular adventure game. The group included Tom Walton, Tim McNamee, Doug Pemberton, Eric Begehr, and myself, Dennis Krabbe. We gathered the three primary treasures: the treasure chest, the magic mirror, and the magic shield, then headed for the King. But to our surprise, the adventure had one more task to be accomplished before Sir Grahame could inherit the throne.

We started our quest on December 1, 1984 and ended up solving the adventure on January 8, 1985. We all shared our discoveries with each other, which made it possible to solve this complex adventure so quickly.

Dennis G. Krabbe  
Cincinnati, OH 45246

What can we say, Dennis, except, "Hail the conquering heroes and long may they live!" Congratulations on your completion of the Quest.

### TI/IBM Shared Printer

Dear Sir:

Although I have been a follower of 99'er *Magazine* and *Home Computer Magazine* for a number of years, this is the first time I have felt the urge to write you. I was introduced to your magazine a few years ago by my brother, who said it was an excellent source of information in regard to my recently purchased TI-99/4A. I have since added several attachments, including the 99/4 Impact Printer. Unfortunately, my wife's and

my work required that we have an IBM PC compatible. The 99/4 Impact Printer would not work with our Compaq and neither the store I bought it from nor TI-CARES could help. I was, therefore, resigned to purchasing a second printer at \$300 plus.

While glancing through your Vol. 4, No. 5 issue (I miss the dates!) I chanced upon "C-64 to TI Printer—The Missing Link" in your "Letters to the Editor" section. Remove the serial card and you have a Centronics parallel Epson MX-80. Holy compatibility! A forced march to the land of Big Blue and \$49.95 netted us a parallel printer cable looking strong enough to stop a Tiger tank. IT WORKS! Now if I could find a sneaky little switch that would allow me to disable the serial card externally, I would have a printer for all reasons. Know any?

Your magazine has paid for itself many times over in the past, but never quite like this. On the down side, I do miss the ads, which were as much a source of information as the rest of your excellent publication. Please reconsider this position. In any event, your magazine is still one of the best around for the home computer user.

Christopher R. Law  
Wilmington, DE 19810

*Christopher, we don't know of any simple way to disable the serial interface. But there may be an alternative. If your TI home computer system has the peripheral expansion box and the TI RS-232 interface, you already have a partial solution. By obtaining the special cable that connects the parallel interface port between the RS-232 board of the TI peripheral expansion box and your printer's parallel Centronics port, the same interface can serve both computers. This port is then accessed via the device name of "PIO" in your TI software instead of using RS-232 . . . Then when you want to switch from one machine to the other, you need only unplug the IBM cable from the printer and plug in the TI cable.*

### C-64 Bulletin Board

Dear Sir:

Should you be compiling a list of Commodore bulletin boards, you might want to include mine. My system has been in operation for a bit shy of a year. It operates 24 hours a day and is rarely taken down as I have a backup computer with which to do my maintenance.

Write to the following address for more

information:

Twin City Express  
Box M  
Mendota, MN 55150

The (612) 778-0506 Commodore Info & Message Bases number operates using a standard Commodore 64, a Westridge modem, three 1541 disk drives and a Comrex mini-monitor. The user log is recorded on a Manesman Tally MT-80 printer.

James Meehan  
St. Paul, MN 55106

*Thanks for the information, James. Commodore owners with modems take note: If you wish to use the bulletin board you should drop a line to the address above to receive an access code from James, so that you can log-on to the system.*

### Apple Double Hi-Res Plotting

Dear Sir:

I own an Apple IIe and an 80-column card (Rev B). I am trying to use the double high resolution for better plotting. The molex-type pin is in place, but I don't know where the Annunciator 3 soft switch is.

Once the double high resolution is activated, do I need to do anything special to plot on the screen? Would HPLOT 340,30 work?

Brian Kramer  
Boise, ID 83704

*The double hi-res mode on the Apple IIe and IIc requires quite a bit of software to access, Brian, and is much too complicated to explain here. It requires special assembly-language drivers to interface with Applesoft BASIC. Unless you are an experienced assembly language programmer, we don't recommend trying to access this capability on your own. A number of software packages are available (notably the Doublestuff package from Doublestuff Software, Inc. 2053 West 11th Street, Brooklyn, NY 11223) that make accessing double hi-res a breeze. For \$39.95, the Doublestuff package is a good buy. For a complete review of this package, refer to Volume 4, Number 3 of Home Computer Magazine.*

### Parallel Port Shortcut for Jr?

Dear Sir:

I am trying to determine the most economical way to equip my PCjr with a parallel printer port that would

Continued on next page



accommodate a printer such as the Epson FX-80. My reason for writing is that I knew your magazine had construction articles from time to time and I thought I might check to see if you had done one on this subject.

Louis F. Ostendorff  
Bluefield, WVA 24701

*No, Louis, we have not run an article on how to add a parallel printer port to the PCjr and we would be interested in knowing if anyone out there has done such a project. If so, they might care to submit their design and an article for publication in Home Computer Magazine. We would certainly consider publishing it.*

#### Fast 99/4A Auto Spell-Check

Dear Sir:

I am writing to compliment you on your fine review of my product, 99/4A Auto Spell-Check. [See Vol. 4, No. 5—Ed.] Your review was concise, objective, and most importantly, accurate. Interestingly, you reported what, in my opinion, is the only significant flaw in version 1.0—SPEED!!! Now available is the faster 99/4A Auto Spell-Check version 1.1. Version 1.0 checked a 100-sector document in about 21 minutes, while version 1.1 checks the same document in about 6.5 minutes.

I wanted to let you, and particularly your readers who own version 1.0, know about this significant improvement. Owners of version 1.0 should send their original diskette and \$3 to Dragonslayer ASC, 2606 Ponderosa Drive, Omaha, NE 68123 for an update. Others may purchase 99/4A Auto Spell-Check version 1.1 from their local dealer or favorite mail-order outlet.

Again, thanks for your continued support of 99/4A owners.

Thomas W. Kirk  
Dragonslayer ASC  
Omaha, NE 68123

*Thanks, Tom, for sharing this information with us on the upgrade of your Auto Spell-Check product.*

#### Low-Cost Word Processor

Dear Sir:

I have been subscribing to your publication ever since I purchased my TI-99/4A home computer and have been very satisfied. I have several questions which I thought you might be able to help me answer. I have tried

several sources and am now so totally confused that I need a "disinterested" party to help.

My wife will be starting some graduate school programs in the next few months and will need to do some manuscript word processing. There is not the need for an especially sophisticated system, yet there is no sense getting "junk." I do not have disk or memory expansion (and would like to avoid spending the \$500+ needed to get them at this time). I do have Extended BASIC, however.

I would very much appreciate your suggestions on the best way to go, given my needs and restrictions. I realize that a \$50 program will not do what one ten times as expensive/fancy will, but at least it will get us started.

Hoyt E. Allen, M.D.  
Kaufman, TX 75142

*We suggest, Dr. Allen, that you stay tuned to Home Computer Magazine. In the next issue we will be doing a review of a product called Missing Link, a package that includes simple word processing software and the necessary hardware to allow connection between a printer and the joystick port on the TI machine.*

#### Apple "Hello" Broken?

Dear Sir:

Your Home Computer Magazine for port home computers is the best that I have seen so far. I feel your magazine will be hard to beat.

In 1984 my wife and I had a big year. We were blessed with our first child, we purchased some land to build us a home on in a few years, and we bought an Apple IIc. As you can see, there had to be a shift in our spending priorities. The belt has been tightened, but your magazine makes owning and operating a home computer affordable. The program listings and the reviews are very appreciated.

On page 6 of your August 1984 issue (Vol. 4, No. 3), I found a "hello" program for the Apple. When I tried using it, I got the message BREAK IN 320. I am wondering, was there an error in line 320, or was it written for Apple IIe's only?

Ken Brantley  
Shreveport, LA 71107

*Thanks for the compliments, Ken, and congratulations on the additions to your family. The problem you experienced in try-*

*ing to run the HELLO program is probably due to the fact that you are running under ProDOS, and the program you typed in is only DOS 3.3-compatible. If you order our ONDISK for 4.3 (which is a DOS 3.3-based disk), and key in the HELLO program that you found on page 6, it will work as stated in that letter.*

#### C-64 Software Rx Search

Dear Sir:

We are trying to locate a Physician Accounts Receivable program for a Commodore 64. Please help us locate such a program and perhaps other programs set up for physicians.

James R. Gwilliam  
Huerfano Memorial Hospital  
Walsenburg, CO 81089

*We are not aware of the availability of the particular program you describe, James, but this does not mean that it does not exist in the large world of Commodore software. In fact, it probably does. A call to Commodore, however, did not turn up any leads. We suggest that you contact the Commodore users group that is closest to you. In this case, we believe it to be the Western Slope Commodore User Group, 535 Main Street, Grand Junction, CO 81501, (303) 242-0083. If any readers know of any medical-accounting software for the C-64, please let us know.*

#### SUBSCRIBER LABEL CHANGES

Starting with this issue, the first line of the subscriber mailing label has been modified to clearly indicate which issue will be the last in your current subscription. The sample mailing label below illustrates the new format:

95000ALEXAND V06N08 H12987090  
DONALD ALEXANDER III  
4321 BROWNING STREET  
CONTINENTAL CA 95000

Notice that the first line is composed of three sections: (1) The first section is a computer search code for our use, (2) the last section is the actual subscriber number, and (3) the middle section tells you with which issue your subscription expires. In this case, V06N08 means that the Vol. 6, No. 8 issue will be your last.



# It Figures!

by Robert Paschelke  
and the HCM Staff



*Whatever your problem, you can reach a solution . . .  
... but you must have the right formula!*

**A**re you a physics student working with motion equations? Or a worker planning a savings strategy? Or perhaps a home owner figuring the cost of a new carpet? Many of us often encounter situations in which we want quantitative results only a mathematical formula will provide. If only we could just run to the computer, type in the proper formula, enter some values, and get a quick solution . . .

*It Figures!* is a handy mathematical tool designed to do just this job. It allows you to use up to 8 variables to create even a complicated formula, and then calculate its answer. Think of a variable as being a bucket: we identify each bucket with a name or symbol, then we fill them with different numeric values. (We're not going to consider string variables in this program.) When we express some mathematical relationship between the different buckets (variables), we have put together a formula that can be evaluated for all the possible variations in the contents of each bucket.

In this program, each variable can be assigned both a value and a formula. If there is no value (or a value of zero) assigned to a variable, its equation, when calculated, will yield a value for it—depending on the values in the other variables. If the other values in a formula are changed, and the formula is recalculated, the value of the current variable will also change. For example, we will assign these 3 variables the following values:

A = 12  
B = 20  
C = 25

A has the formula  $B+C=$  and the value 12 assigned to it. If we calculate A, it will use the current values for the variables within the formula  $B+C=$  and place the result back in A. After the calculation, A will have a value of 45. A variable can also appear within its own

formula. For example, the formula for A could have been  $A+B+C=$ , which—when calculated—would have placed 57 into A. Notice that when A was used inside its own formula, that the current value for A (12) was used to calculate a new value. The program does not update a value for a variable encountered within its own formula until the entire calculation is complete. Only the current value (the one last calculated) is used.

## Numeric Functions

Every numeric function available in BASIC is incorporated into this program. A list of available functions is as follows:

Command	Function
ABS	ABSolute value
ATN	ArcTanGent
COS	COSine
EXP	EXPonent
INT	INTeger
LOG	LOGarithm
RND	RaNDom number
SGN	SIGN (+ or -) returns 1, -1, or 0
SIN	SINe
SQR	SQUare Root
TAN	TANgent

For a detailed explanation of each function, consult your BASIC reference manual. The syntax for each function is:  $FN(P)$

FN is the function name, and P is the parameter. For example, you would designate the sine of A as  $SIN(A)$ .

## Using the Program

*It Figures!* screen is divided top to bottom into 3 main "windows": a variable list, a formula window, and a "help" window. The upper window is also divided into a value field and a variable label field. Eight variables are displayed in this upper window. Each variable row



13





includes (left to right) the variable letter, an equal sign, the value of the variable, a colon, and the variable label field. The value and label fields can be edited with the edit keys (insert, delete, erase, etc.) which are listed in your system's Control Capsule. To toggle between these two fields, press either [ENTER] or [RETURN], depending on your system.

To make it all easier to understand, we will take you through each step of the program, using an example formula that calculates the "future value" of a savings deposit. The formula is:

$$B \left( (C + 1)^D \right) = A$$

where:

- A is the future value of the deposit.
- B is the present value.
- C is the interest rate.
- D is the number of years compounded.

To assign a value to any variable, simply move to its value field and type in the value—then toggle to the label field and type in a label. For example, move to A, leave the value as 0 (it's our unknown), and then toggle over and enter FUTURE VALUE as a label. Next, toggle back to the value field, move down to B, enter 1000 as a value and PRESENT VALUE as a label. Continuing on, enter C equal to .0525 and label it INTEREST PER YR; then enter D equal to 12 and label it YEARS.

You can also assign a formula to any variable. To do this, move the cursor to the row containing the variable and press the Edit key for your system. In this case, move to A and press the Edit key. The cursor will appear in the formula window and the current value field for A will be displayed below the cursor (as A=0). The current value to be calculated will always be displayed here, along with its assigned formula.

If there were already a formula assigned to this variable, the formula would be displayed in the formula window, and the cursor would be on the first character of the formula. Because there is no formula yet, the cursor is on the first character position of the formula to be entered.

From this point, the formula can be created or edited using the editing keys. To complete entry in the formula window, press the appropriate key to return to the variable field you left when the formula field was called. The last formula you worked on will continue to be displayed in the formula window until another formula is worked on or created. Now, try entering the above Future Value formula.

## Calculating

You can calculate any formula for a variable by moving the cursor to a row that contains the variable you want to calculate. To solve that variable's formula, press the Calculate key for your system. As yet, we have only entered a formula for A, to calculate the future value of our \$1000 deposit. Try moving to A in the value field and press the Calculate key for your system. After a few moments, you will see a calculated value appear both in the value field in the upper window, and after the current variable displayed in the formula window. You can now assign formulas to any other variables, such as this formula for D (years or times compounded):

$$(\text{LOG}(A/B))/(\text{LOG}(C+1)) =$$

Or, you could define a new unknown variable—like E

***"If only we could just run to the computer, type in the proper formula, enter some values, and get a quick solution . . ."***

for INTEREST EARNED—and assign it a formula, such as:

$$(B*((C+1)^D))-B=$$

OR

$$A-B=$$

You might notice that if you enter both of these formulas and try some examples, they may not come up with precisely the same answer. This is due to the limited accuracy of your computer. [For more on this subject see "Microcomputer Accuracy" in *Home Computer Magazine*, Vol. 4, No. 1—Ed.]

As an aid to increased understanding of how this versatile tool may be called upon to perform its useful, numerical magic, we have provided more examples in Figure 1. You may use these just for practice—or, for a practical purpose. (Although the sample formulas we give here are fairly simple, the program can handle any complicated formula as large as 78 to 84 characters, depending upon your system.)

**Figure 1**  
**Sample Formulas**

1. To calculate the height a rocket will reach in a certain time if the initial velocity is known:

$$(B^2C) - ((D/2)*(C^2)) = A$$

where

- A is the height reached (feet).
- B is the initial velocity (feet/second).
- C is the time in seconds.
- D is the gravitational acceleration (32/ft./sec./sec).

2. To calculate the cost of material (carpet, siding, etc.) covering a certain area:

$$B^2C = A$$

where

- A is the total cost.
- B is the cost of material (\$ per sq. ft.).
- C is the surface area (sq. ft.).

If B is in square yards, the formula is:

$$B^2(C/9) = A$$

3. To convert ounces to grams:

$$B/C = A$$

where

- A is the number of grams.
- B is the number of ounces.
- C is the conversion factor (.035 in this case).

This formula can convert virtually any U.S. weight or measure to metric with the appropriate conversion factor obtainable from most common dictionaries.



## Further Options

By pressing the Print key for your system, you can get a hard copy of the current contents of your variables, their present values, their labels, and the formula.

You can save and recall your variables and formulas with a disk or cassette tape (on systems with a cassette interface). Every time you save a file, you are saving *everything* in memory, including all present values. You may update a file by saving to the same file name; or you may save to a new name, creating a new file to hold your latest changes.

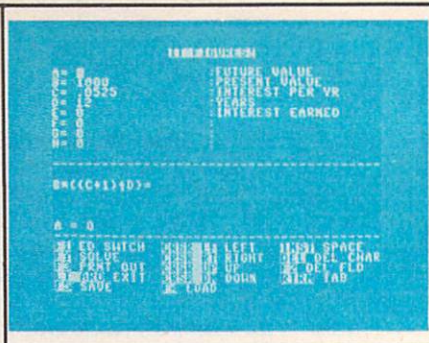
## A Note On Formulating

This program works through any formula from left to right—hence, the equal sign sits all the way to the right (you don't need to add the unknown variable after the equal sign). You must take this into account when entering a formula, because the order of operations is very important in getting the proper result. Technically speaking, you must "force the precedence of operators" with the placement of parentheses. For example, in the formula  $B*((C+1)^D)=$  had we not placed a parenthesis on both sides of  $(C+1)^D$ , the program would have multiplied  $B$  times  $(C+1)$  and then would have used the resulting value under the exponent  $D$ . Placing the outside parentheses around the entire statement  $(C+1)^D$  ensures that  $B$  will be multiplied by the value derived from that statement.

If an equation does not end with an equal sign, the full input field of 80 characters will be read and then calculated. Otherwise, the program will read to the equal sign and calculate immediately.

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

This representative screen photo taken from the C-64 version of the program shows the formula for figuring the value of a savings deposit and the relevant values.



## CONTROL CAPSULE It Figures!

KEY	FUNCTION
F1	Move to formula-edit field (create or edit formula).
F2	Blank a line.
F3	Print formulas (hard copy).
F4	Load formulas (disk or tape).
F5	Save formulas (disk or tape).
F7	Calculate a variable (update results in variable table).
[RETURN]	Tab between variable and label fields.
Ins	Insert character (all fields).
Del	Delete character (all fields).
—	Escape (from formula-edit field to variable-edit table; from variable-edit table to a prompt asking users whether they are sure they want to exit the program).
Crsr right	Move right one character.
Crsr left	Move left one character.
Crsr up	Move cursor up one line.
Crsr down	Move cursor down one line.

## CONTROL CAPSULE It Figures!

KEY	FUNCTION
Ctrl E	Move to formula-edit field (create or edit formula).
Ctrl B	Blank a field.
Ctrl P	Print formulas (hard copy).
Ctrl L	Load formulas (disk or tape).
Ctrl O	Save formulas (disk or tape).
Ctrl R	Calculate a variable (update results in variable table).
[Return] or Ctrl I	Tab between fields
Ctrl A	Insert character (all fields).
Ctrl Z	Delete character (all fields).
Esc	Escape (from formula-edit field to variable-edit table; from variable-edit table to a prompt asking users whether they are sure they want to exit the program).
↑	Cursor up.
↓	Cursor down.
→	Cursor right one character.
←	Cursor left one character.

### SPECIAL FOR II+ USERS:

Ctrl K	Cursor up
Ctrl J	Cursor down
Ctrl U	Cursor right
Ctrl H	Cursor left

## CONTROL CAPSULE It Figures!

KEY	FUNCTION
F1	Delete character (all fields).
F2	Insert character (all fields).
F3	Blank a field (all fields).
F4	Print formulas (hard copy).
F5	Load formulas (disk or tape).
F6	Save formulas (disk or tape).
F7	Calculate a variable (update results in variable table).
F8	Move to formula-edit field (create or edit formula).
F9	Escape (from formula-edit field to variable-edit table; from variable-edit table to a prompt asking users whether they are sure they want to exit the program).
[ENTER]	Tab between variable and label fields.
—	Move cursor right one character.
←	Move cursor left one character.
↑	Move cursor up one line.
↓	Move cursor down one line.





### CONTROL CAPSULE *It Figures!*

#### KEY FUNCTION

FCTN 1	Delete character (all fields).
FCTN 2	Insert character (all fields).
FCTN 3	Blank a field (all fields).
FCTN 5	Load formulas (disk or tape).
FCTN 6	Save formulas (disk or tape).
FCTN 7	Calculate a variable (update results in variable table).
FCTN 8	Move to formula-edit field (create or edit formula).
FCTN 9	Escape (from formula-edit field to variable-edit table; from variable-edit table to a prompt asking users whether they are sure they want to exit the program).
[ENTER]	Tab between variable and label fields.
FCTN D	Move cursor right one character.
FCTN S	Move cursor left one character.
FCTN E	Move cursor up one line.
FCTN X	Move cursor down one line.
FCTN P	Print formulas (hard copy).

A small deviation from standard BASIC occurs on the TI-99/4A version of *It Figures!* On the TI machine there is no parameter in BASIC for the RND function. To simplify the code in the program, however, you will need to supply a parameter for RND. The parameter you pass will determine the size of the random number. For example, RND(5) would give you a random number between 0 and 5.

HCM

### *It Figures!* (Apple II Family) Explanation of the Program

Line Nos.	Explanation
100-190	Program header.
200	Set up error handling.
210-240	Main program-control sequence.
250-630	Initialize program.
640-970	Display main screen for <i>It Figures!</i>
980-1000	Display answer on the screen.
1010-1090	Main control for entry of fields.
1100-1930	Variable value entry routine.
1940-2010	Evaluate formula from number entry fields.
2020-2070	Print hard copy of formulas and variables.
2080-2210	Disk-access control routines.
2220-2980	Variable description entry.
2990-3110	Solve formula from variable label.
3120-3760	Formula entry.
3770-3850	Evaluate formula from formula field.
3860-3890	Print hard copy.
3900-4210	Parser and solver.
4220-4320	State search definition.
4330-4550	Syntax error prompt.
4560-5880	Evaluate formula subroutines.
5890-6010	Print-out-hardcopy control routine.
6020-6070	Get a character from the keyboard.
6080-6260	Error routine.
6270-6770	Disk-access subroutines.

### *It Figures!* (C-64) Explanation of the Program

Line Nos.	Explanation
100-200	Program header.
210-240	Main program-control sequence.
250-500	Initialize program.
510-780	Display main screen for <i>It Figures!</i>
790-880	Cursor subroutines.
890-920	Display answer on the screen.
930-1010	Main control for entry of fields.
1020-1890	Variable value entry routine.
1900-1970	Evaluate formula from number entry fields.
1980-2030	Print hard copy of formulas and variables.
2040-3010	Variable description entry.
3020-3720	Formula entry.
3730-3820	Evaluate formula.
3830-3860	Print hard copy.
3870-3900	Save from formula entry.
3910-4280	Parser and solver.
4290-4430	State search definition.
4440-4640	Syntax error prompt.
4650-6050	Evaluate formula subroutines.
6060-6240	Print-out-hard-copy control routine.
6250-6660	Disk-access subroutines.
6670-6810	Get a character from the keyboard.

### *It Figures!* (IBM PC & IBM PCjr) Explanation of the Program

Line Nos.	Explanation
100-220	Program header.
230-380	Initialize program and branch to main control loop.
390-400	End-program routine.
410-450	Main program-control loop.
460-550	Entry-control subroutines.
560-610	Edit-screen routines.
620-1240	Evaluate-formula routine.
1250-1330	Disk-access routines.
1340-1410	Syntax error messages.
1420-1840	Formula-entry routine.
1850-1860	Function key initialization DATA.
1870-2060	Display subroutines.
2070-2340	Keyboard-input subroutines.
2350-2390	Disk-error routines.
2400	Printout routine.

### *It Figures!* (TI-99/4A) Explanation of the Program

Line Nos.	Explanation
100-210	Program header.
220-320	Initialize program, and branch to main control loop; end-program routine.
330-350	Display main screen.
360-490	Entry-control subroutines.
500-560	Edit-screen routines.
570-1200	Evaluate-formula routine.
1210-1270	Disk-access routines.
1280-1340	Syntax error messages.
1350-1700	Formula entry routine.
1710-1860	Keyboard input, and screen display subroutines.
1870-1880	Print-out routine.
1890-1930	Disk-error routines.



*It Figures!* requires TI Extended BASIC.





# EVACU-POD

by William K. Balthrop  
HCM Staff

*On another planet,  
in another time,  
you delicately guide your ship  
on a hazardous rescue mission.  
Will you make it on time?  
Will you make it at all?*

This game is based on an old all-time favorite—*Interplanetary Rescue*—originally published in our forerunner, *99'er Magazine*. Even our old fans of the original will greatly enjoy this much-improved, option-filled version.

**T**he call is out! Send *Evacu-Pod*!  
The time: in the near to distant future.  
The place: a neighboring planet.

Mining operations have proven profitable—but risky. Interplanetary construction crews have built full-scale communities on 4 planets, each with its own hospital serviced by crack rescue units. This is where you come in. You've trained back home to fly *Evacu-Pod* missions, but nothing on the relatively gentle Earth has prepared you for the varying rugged terrain that surrounds each medical center. *Evacu-Pod* pilots have rated the terrains for their hazards. You may volunteer to fly at any level: beginner, advanced, expert, or professional.

After indicating your skill as a pilot, you can choose the planet to which you would like to be assigned: Earth's moon, Mars, or Venus. A fourth option allows you to designate the size of another planet where you would like to work with gravity ranging from 1 to 9. For comparison purposes, Earth's moon is a 2 and Venus is a 6.

Once you've selected a planet, you will be able to select a time period for your mission. Your choices range from the year 1995 to 2485. As the space program develops over the centuries, more efficient rockets and rocket fuels are invented; thus, the later the year, the easier the assignment will be. This efficiency is incorporated into your rescue vehicle. In 1995, the rockets are fairly inefficient, and so require more fuel to lift the same amount of weight as a rocket built in 2485.

## Winning The Game

To win in *Evacu-Pod*, you must pilot your rescue craft from your home base, over the mountains and valleys of the planet's surface, to the mining camp where you must rescue an injured miner. There you must land at the mining camp, take off again, and safely make it back to your home base. You must do this without crashing, running out of fuel, or running out of time. If you take too long, the injured miner will die while waiting, and

you will have failed your mission. The time remaining is not displayed on the screen. This adds a little suspense to the program—you never know when it may be too late . . .

## Radar Map

Most of the screen is taken up by the radar map. This is like a topographical map of the local area. Your home base is in the upper-left corner, with the mining camp in the lower-right corner. The radar map is made up of different colors, each color representing a different altitude in 2000-meter increments.



Screen is taken from the  
IBM version of *Evacu-Pod*

## The Radar Altimeter

To the right of the screen is the radar altimeter, which records your altitude in relation to the elevations on the main radar map. The maximum altitude on this altimeter is 10,000 meters. The altimeter serves two purposes: If you are 200 meters or more directly above the planet's surface, the colors of the radar map are displayed vertically, showing their relative altitudes, with a picture of your ship

next to the display showing its vertical position. If you are next to one colored portion of the altimeter, you will need to climb before entering the like-colored portion of the radar map. (You will need to be above that color.)

If your ship is within 200 meters of the surface (say, you are at 6150 meters altitude in relation to your base and the terrain you're flying over is at 6000 meters), a different scale will appear on the radar altimeter. The new scale ranges from 0 to 200 meters in relation to the surface directly beneath you. This scale is especially helpful in landing your rescue pod.

## Instrumentation

In addition to the radar map and radar altimeter, 6 additional readouts will help you pilot your ship:



**BASE** — BASE altitude is your altitude in relation to both your home base and the mining camp. Even if you are flying over a 6000-meter mountain top at just 25 meters above the top of the mountain, your BASE altitude would be 6025 meters.

**SURF** — The SURF altitude indicates how close you are to the actual surface of the planet. Using our earlier example, if you were flying over a 6000-meter mountain top at a BASE altitude of 6025 meters, your SURF altitude would be 25 meters. Under this condition, the radar altimeter would be displaying a range of 0 to 200 meters because your surface altitude is below 200 meters.

**FUEL** — This readout indicates the number of kilograms of fuel that you have remaining. This fuel is added into the weight of your ship. As your fuel is depleted, it will take less power to lift your ship. Once you run out of fuel, your ship will cease to output any thrust and will drop like a rock to the surface.

**POWER** — The amount of fuel being used by the rockets is displayed here. Three units of power are equal to one kilogram of fuel used every second. The amount of lift generated by the power setting will be determined by the efficiency of your rockets. As mentioned earlier, you can adjust this by selecting a year for the mission. With an efficiency of 1, 1 unit of power would generate 1000 newtons of thrust. The efficiency factor is .66 in 1995 and 2.66 in 2485. This means that only 666 newtons of thrust are generated for a unit of power in 1995, while 2660 newtons of thrust are generated with each unit of power in 2485.

**H.VEL** — Horizontal VELOCITY is the speed at which you are traveling across a planet's surface. This speed has no bearing on whether you are going up or down, it's just your speed across the surface. Velocity is measured in meters per second. One meter per second is roughly 2 miles per hour.

**V.VEL** — Vertical VELOCITY is the speed at which you are either climbing or descending in altitude. If your vertical velocity is a positive number, you are rising—moving farther from the surface. If your vertical velocity is a negative number, you are falling back toward the surface. A vertical velocity of 0 would indicate that you are hovering at one altitude. Velocity is measured in meters per second. See H.VEL.

---

*"... nothing on the  
relatively gentle Earth  
has prepared you for  
such a rugged terrain."*

---

## Controls

The Evacu-Pod is like a rocket-powered helicopter—it always flies nose up and has five rocket engines to enable you to maneuver. Four small rockets, one on each side of the pod, give you control over your horizontal direction of travel and velocity. One large rocket beneath the pod supplies all of the lift.

For the keys to use to control your Evacu-Pod, see the Control Capsule.

## Landing

Landing the ship is very tricky and difficult. You must touch down at a velocity of -4 meters per second (mps) or less to have a perfect landing. You can land at any location on the map, but if you are not directly over the base and you land at a rate faster than -4 mps, you will crash. The results of landing on a base or mine are a little more bearable:

### V. VEL AT TOUCHDOWN

	RESULT
0 to -4 mps	Perfect landing.
-4.01 to -10 mps	Landing gear damaged.
-10.01 to -15 mps	Fuel leak—lose half of fuel.
-15.01 to -25 mps	Ship is inoperative. You can't take off again.
-25.01 and up	Crash. You and your crew die in the explosion.

Now get out there, mind your controls, and rescue those miners. Happy Landing!

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

### CONTROL CAPSULE Evacu-Pod

KEY	FUNCTION
U	Increase power light.
I	Increase power medium.
O	Increase power heavy.
J	Decrease power light.
K	Decrease power medium.
L	Decrease power heavy.
E	Thrust ship to the North.
S	Thrust ship to the West.
D	Thrust ship to the East.
X	Thrust ship to the South.
T	Initiates takeoff thrust.
1	Pause game. Continue by pressing any key.



The Apple II version of *Evacu-Pod* uses unique character-oriented, machine-language graphics routines, which are located in DATA statements at the end of the program. The two variables, PRNT and REST, contain the addresses used in CALLs. The PRNT routine redirects the output of a PRINT statement from the normal text screen to the high-resolution (hi-res) graphics screen. In addition, the routines change the characters PRINTed into the graphics which appear on the screen in *Evacu-Pod*. The characters are contained in the DATA statements from lines 1540-2370.

After the PRINT to the hi-res screen, the system must have its output routine restored, and this is the purpose of the REST routine.

This program is designed to run under both ProDOS and DOS 3.3. The PRNT routine has to keep the Disk Operating System (DOS) aware of where the screen output is being directed. However, the 2 operating systems have different addresses where this information must be placed, so a special flag is set at zero-page location 6 telling which operating system is in charge.

By PEEKing 2 locations (49505 and 49511) in line 250, the operating system in charge can be discovered. These locations are part of the ProDOS System Global Page. Even in future updates of ProDOS, these locations will contain the same values, so you can feel confident that the program will remain compatible with future versions.

At the same time, we have found that these same locations do not contain these values when DOS 3.3 is the DOS, so this program will function properly whether keyed in or converted to either system.





One of the Commodore 64's strengths is its 10-character keyboard buffer. In a game like *Evacu-Pod*, however, the program needs to isolate each keypress, and not keep a queue. Instead of using the **GET** statement, this program bypasses the buffer and directly **PEEKs** the keyboard (location 197) for the key presently being pressed.

If no key is pressed, a **PEEK** of location 197 yields a 64. The following 2-line program allows you to see exactly what values are present when you press a key.

```
10 K = PEEK(197):IF K = 64 THEN 10
20 PRINT K:GOTO 10
```

When you press a key while running this program, the value **PEEKed** from location 197 is printed to the screen. A look at lines 930-1080 will reveal how *Evacu-Pod* uses this same input method.



Quite a bit of mathematical manipulation is involved in *Evacu-Pod*, and the IBM machines make displaying the results to the control panel very easy—the key is the **PRINT USING** statement. The control panel is displayed in lines 1100-1160. For example, here's line 1100:

```
1100 LOCATE 23,1:PRINT USING "& #####.###";"BASE:
";ABS.ALT::LOCATE 1,1
```

Here the **ABSolute ALTitude** from the **BASE** is being displayed. Because this number depends upon the *Evacu-Pod*'s acceleration, it could have many decimal places. The **#** symbols in the **PRINT USING** statement each stand for one digit. They tell the BASIC interpreter to round off the number to the number of digits indicated and display only that number. The **&** symbol tells the system to display the entire string indicated—in this case, the word **BASE**.



*Evacu-Pod* makes excellent use of the TI Extended BASIC **DISPLAY AT** statement. To depict the surface of the planet, an entire screen of colorful graphics is required. Because 21 rows are needed to display the screen, you might expect to see lines and lines of **PRINT** statements for each of the 4 possible screens available. Instead, the entire screen is displayed with just one line of code for all 4 screens.

This one line is set up with a series of 4 **RESTORE** statements in lines 1370-1400. The location **RESTORED** to is determined by the skill level chosen by the player. The option is placed on the **O(0)** array, and that value is used in the **ON GOTO** statement in line 1360. One of 4 sets of **DATA** is **READ** in line 1410 to actually display the screen:

```
1410 CALL CLEAR::CALL COLOR(9,10,12)::FOR A = 1 TO 21::READ
A$: DISPLAY AT(A,1):A$: NEXT A
```

If you look at the **DATA** in lines 1430-2300, you will see a number of lower-case letters and punctuation marks. These characters were redefined earlier as either solid blocks or spaces, then **CALL COLOR** was used to set up foreground and background colors.

HCM

## **Evacu-Pod (Apple II Family) Explanation of the Program.**

Line Nos.	
100-200	Program header.
210-410	Initialize program.
420-450	Main menu.
460-590	Main program loop.
600-740	Display subroutines.
750-920	Keyboard input routines.
930-1150	Routines to calculate ship's state.
1160-1210	Update control-panel routine.
1220-1510	Check landing condition.
1520-2370	Terrain display and data.
2380-2470	Screen messages.
2480-2680	Get player options choices.
2690-2830	Message and machine language data.

## **Evacu-Pod (C-64) Explanation of the Program.**

Line Nos.	
100-190	Program header.
200-380	Program initialization.
390-550	Get level input and draw screen.
560-660	Main program loop.
670-790	Display control panel.
800-1370	Display altimeter and update variables.
1380-1800	Landing routines.
1810-2000	Output subroutines.
2010-2920	Radar map print routines.
2930-3250	Sprite DATA.
3260-3420	Sound and message DATA.

## **Evacu-Pod (IBM PC & IBM PCjr) Explanation of the Program.**

Line Nos.	
100-230	Program header.
240-310	Program initialization.
320-540	Get level input and draw screen.
550-740	Main program loop.
750-890	Update variables.
900-1090	Display altimeter.
1100-1190	Display control panel.
1200-1690	Radar map DATA.
1700-1800	Option input routine.
1810-2020	Print messages, end game routines.

## **Evacu-Pod (TI-99/4A) Explanation of the Program.**

Line Nos.	
100-190	Program header.
200-290	Initialize program.
300-370	Main control loop.
380-500	Display routines.
510-670	Get input for ship controls.
680-730	Calculate velocity and altitude.
740-790	Check for a collision.
800-890	Check ship position, update radar.
900-950	Display control-panel values.
960-1010	Landing and crash away from base.
1020-1070	Option to play again. Display score.
1080-1260	Landing and crash at base.
1270-1420	Get options for a new game.
1430-2310	Terrain data and display.
2320	Key input, and pause routine.
2330-2450	Data for option messages.



*Evacu-Pod* requires TI Extended BASIC.





# WITCH'N'SPELLS

by **Randy Thompson**  
and the HCM Staff

*Simply knowing how to spell words isn't enough  
to conquer this challenging word game—  
you've got to know how to spell them forwards and backwards!*

Here's a real challenge! Spelling can be hard enough, but you can count on your computer to make it even more challenging—or perhaps *intriguing* is a better word. We are accustomed to having the computer put everything in order; but it is also quite good at putting things in *disorder*—such as the letters of a word. Then, by putting things right again, you also end up learning how to spell that scrambled word, from the “inside out.”

*Switch 'n' Spell* is just such a spelling aid, with an unexpected twist. In an entertaining way it combines logic and the language arts to create a unique educational game for use with your computer. Good spelling is the main object in this program, but unlike many simple drill programs, it's fun! As for age level, the words can be simple for the young or difficult for us more experienced spellers.

When the program begins you are given a menu with two options:

- (1) Create Word List
- (2) Load Word List

Since the game cannot be played unless a word list is in memory, one of these options must be chosen before the program will proceed.

## (1) Create Word List

This mode allows you to create and edit a new or previously prepared word list. If a word list is already in memory when you select this mode, you will be asked whether you want to add to the current list or create a new one. If you choose to create a new one, the list currently in memory will be erased.

The editing screen will display 10 words at a time. Once a screen of 10 words is filled, a new screen will appear. To return to the previous screen, simply press the proper key (as shown in the Control Capsule for your machine). Similarly, another keypress will get you to the next screen if the current one has already been filled.

To input words, simply type them in. Each machine has its own editing functions detailed in the proper Control Capsule. You may enter a list of up to 50 words.

## (2) Load Word List

To use this option, you must have created a list previously and saved it on tape or disk. Upon entry of

this option, you will be prompted for a file name. Use a file name appropriate to your system.

Once a list is in memory, the main menu will expand with 4 more options, creating a total of 6. The last 4 options are:

- (3) Save Word List
- (4) Print Word List
- (5) Play Switch 'n' Spell
- (6) Exit Program

## (3) Save Word List

This mode allows you to save the word list that is currently in memory. As in the Load Word List option, you will be asked for a file name, and again, respond with a file name appropriate to your system.

## (4) Print Word List

Use this option when you want either a hard copy of the word list or to simply view it on the screen. You can select the printer or the screen to receive the output.

If you choose to print the list on the screen, words 1 through 10 will be displayed first. This mode is very similar to the Create Word List option. A specified key will allow you to see the previous 10 words, and another will show the upcoming screen of 10 words.



Screen taken from the  
IBM version of *Switch 'n' Spell*

## (5) Play Switch 'n' Spell

Immediately after you enter this mode, the computer will display a word on the screen and then erase it. This is the word you are to spell.

Now, here's where the *switch* comes in. In the upper section of the screen the same word that you were just shown will be printed—the same word, except that all of its letters have been mixed up. Your object is to switch letters, starting from the left, until the word is spelled correctly. To switch the letters, you are asked to input a number. If you input a 3, the first three letters of the scrambled word are reversed. A 5 would reverse the first five letters. Here's an example: The word SAG might be scrambled as AGS. To switch this correctly, first input a 2. The first two letters, A and S, are now reversed to turn AGS into GAS. Now, when you enter a 3, the



whole word GAS is reversed, and you get the desired result, SAG, in just 2 switches.

When the scrambled word is switched correctly into the right spelling, the computer will give a whistle and tell you how many switches you made. A new word will again be displayed, and if you're ready, you may try your hand at another scrambled mess.

If you can't remember the original word, the proper control key will allow you to view it again for a brief period of time. For those of you who give up on switching the scrambled word back into order, pushing another key will make the computer solve it for you.

The computer chooses words sequentially from the word list. This is helpful when you want the words to increase or decrease in difficulty. When the whole list has been used, the cycle starts over.

### (6) Exit Program

If you have created or changed a quiz without saving it before choosing option 6, you will first be asked whether you really wish to exit the program. If you answer anything but yes, you will be returned to the menu. If you do answer yes, the program will end. Typing RUN will start it up again, but any word list that might have been in memory will be lost.

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

***"We are accustomed to having the computer put everything in order; but it is also quite good at putting things in disorder—such as the letters of a word."***

they are all the same length—and can therefore be PRINTed to the screen starting in the same column (HTAB). Only the row number (VTAB) must be altered to move the different elements around.

The WM\$( ) array is manipulated in lines 4230-4240, and printed to its proper locations on the screen in the subroutine in lines 4390-4400.



### CONTROL CAPSULE Switch 'n' Spell

KEY	FUNCTION
F7	Exit to main menu
<i>Play Mode:</i>	
F1	View word unscrambled.
F3	Autosolve.
<i>Edit Mode:</i>	
F1	Return to previous 10-word screen.
F3	Go on to next 10-word screen.
SHIFT INST	Insert letter
DEL	Delete (backspace).
Crsr Down or Return	Next word.
Crsr Up	Previous word.
Crsr Left	Cursor left.
Crsr Right	Cursor right.

### CONTROL CAPSULE Switch 'n' Spell

KEY	FUNCTION
Esc	Return to main menu.
<i>Play Mode:</i>	
Control E	View unscrambled word.
Control R	Autosolve.
<i>Edit Mode:</i>	
Control Q	Insert space.
Control Z	Delete a character.
Control B	Blank a word.
Control I	Insert a word space.
Control D	Delete a word.
↑	Previous word.*
↓	Next word.*
←	Cursor left.*
→	Cursor right.*
<i>*SPECIAL FOR II+ USERS:</i>	
Control K	Previous word.
Control J	Next word.
Control H	Cursor left.
Control U	Cursor right.

The Apple version of *Switch 'n' Spell* animates the reversal of letters by string manipulation in conjunction with three string arrays: WL\$( ), WM\$( ) and ET\$( ). WL\$( ) contains the word list, each element containing one word.

When a word is randomized, it is placed into the ET\$( ) array, one character at a time, with one character in each element. The 5 elements of the WM\$( ) contain the different parts of the word as it is being reversed. These different sections are all "padded" with spaces so that

One of the more striking aspects of *Switch 'n' Spell* is the animated way in which letters are rearranged on the screen when a reversal occurs. The Commodore 64 makes this easy because the screen can be directly accessed, via PEEKs and POKes, from BASIC.

The code for this reversal of letters is in lines 4290-4490. It is accomplished with two FOR-NEXT loops. The outer loop divides the section of the word where the switch is to occur in half. It moves the characters to be switched from the front and back of the section to above and below the word, replacing their original locations with spaces. The inner loop uses the C-64 ASCII characters that cause an insert (ASCII 148) and delete (ASCII 20) to move the letters to above and below their new locations. Then the subroutine at 4000-4060 places the new word in its proper location.



The IBM PC and PCjr machines have quick-acting graphic commands that are used in *Switch 'n' Spell* to create the borders around the various screens used in the game. No matter which part of the game is being used, the same routine is called to create the inverse-color effect that acts as a border: lines 1490-1520.

Before this routine is called, the title of the screen (e.g., Create Word List, Play *Switch 'n' Spell*, etc.) is displayed normally at the top of the screen. Line 1490 then uses the GET statement to copy the area of the screen to be inverted into an array. The PUT command then places an inverse image in the identical location using the PRESET option.



### CONTROL CAPSULE *Switch 'n' Spell*

KEY	FUNCTION
Fn 1	Return to main menu.
<i>Play Mode:</i>	
Fn 2	Autosolve.
Fn 3	View unscrambled word.
<i>Edit Mode:</i>	
Fn 2	View next 10 words.
Fn 3	View previous 10 words.
Backspace	Backspace
Del	Delete character at cursor.
Ins	Insert character.
↑	Edit previous word.
↓	Edit next word.
←	Cursor left.
→	Cursor right.

Similarly, lines 1500 through 1520 create vertical boxes on both sides using the LNE command. The name *Switch 'n' Spell*, contained in the LOGO\$ variable, is printed vertically within the boxes. Then the GET and PUT commands are used to create the inverse-color effect.



### CONTROL CAPSULE *Switch 'n' Spell*

KEY	FUNCTION
<i>Play Mode:</i>	
A	View unscrambled word.
B	Autosolve.
C	Return to main menu.
<i>Edit mode:</i>	
A	Add word.
B	Change a word.
C	View next 10 words.
D	View previous 10 words.
E	Return to main menu.
FCTN 1	Delete a character.
FCTN 2	Insert mode.
FCTN 3	Blank a word.
FCTN S	Cursor left.
FCTN D	Cursor right.

The TI-99/4A version of *Switch 'n' Spell* is written in TIBASIC, which lacks any simple instructions for moving the cursor on the screen. Any character placed at a specific screen location must be placed there using the HCHAR statement. By using the ASC function in conjunction with the SEG\$ function to extract ASCII values of characters in a given string, any set of letters can be selectively placed in any position on the screen.

We used this technique extensively in animating the "switch" in this program. Where I is the number of characters to be reversed, this FOR statement begins a loop to extract the ASCII values needed in the HCHAR statement to place the characters on the screen in their reverse order:

```
FOR Z=I TO 1 STEP -1
```

Followed immediately by this SEG\$ statement, the characters' ASCII values are placed one by one into the CH variable and are subsequently displayed on the screen with HCHAR:

```
CH = ASC(SEG$(SW$,Z,1))
```

HCM

### *Switch 'n' Spell (Apple II Family)* Explanation of the Program

Line Nos.	
100-190	Program header.
200-210	Set ProDOS flag and error entry.
220-330	Main program control.
340-600	Initialize variables.
610-740	Main menu.
750-1080	Create word list.
1090-1680	Peripheral-access routines.
1690-2860	Keyboard-word-entry routines.
2870-3140	Name-list entry.
3150-3300	Play-game control loop.
3310-4050	Input for game and prompts.
4060-4190	Autosolve.
4200-4640	Word-manipulation routines.
4650-5280	Peripheral-access subroutines.

### *Switch 'n' Spell (C-64)* Explanation of the Program

Line Nos.	
100-190	Program header.
200-540	Initialize variables and main menu.
550-950	Create word list.
960-1410	Disk and tape access.
1420-1910	Print word list.
1920-2590	Play <i>Switch 'n' Spell</i>
2600-3230	Screen-display subroutines.
3240-3600	Input routines.
3610-3720	List the words.
3730-3900	Ask if device is tape or disk.
3910-3990	Print word.
4000-4200	Erase and print to game screen.
4210-4830	Rearrange-word routines.
4840-4970	Autosolve.
4980-5010	Data for screen title, color, note.

### *Switch 'n' Spell (IBM PC & IBM PCjr)* Explanation of the Program

Line Nos.	
100-230	Program header.
240-370	Initialization and main screen.
380-900	Edit, input, and display routines.
910-980	Create word list.
990-1030	End-game routine.
1020-1210	Disk-access routines.
1220-1230	Convert lower case to caps.
1240-1530	Play-game control loop.
1540-1810	Display, music, and input routines.
1820-1900	Printer routine.
1910-2070	Autosolve.
2080-2100	Music and keyboard subroutines.
2110-2180	Print-to-screen routine.
2190-2240	List-cleanup routine
2250-2260	Singular or plural of "reversal."

### *Switch 'n' Spell (TI-99/4A)* Explanation of the Program

Line Nos.	
100-210	Program header.
220-450	Initialization and main screens.
460-560	Create word list.
570-1190	Edit word list.
1200-1410	Storage access routines.
1420-1670	Print-word-list routines.
1680-2160	Play game.
2170-2970	Autosolve
2980	Return to main menu.
2990-3110	End-game routine.
3120-3150	Data for menu screens.





# LASERITHMETIC

by R. G. Christensen  
and the HCM Staff

*By answering math problems right, you'll prove that you have the right stuff to defend your interstellar spacecraft—the Columbia.*

Man-**M**ankind has taken another giant leap into the future and launched its first interstellar spacecraft—manned by none other than YOU. Your ship carries the name of another pioneering spacecraft, the *Columbia* (the first space shuttle). You have no idea what sort of dangers lie beyond the protective boundaries of the solar system. Little do you know that the most deadly force in the universe is waiting quietly as your ship approaches.

Suddenly, you are under attack: not by a swarm of invaders from every direction, but from a single deadly invader—an alien almost unaffected by your laser blasts. Your best hope is to hold it off until help can arrive.

## Education in Space

Such is the scenario of *Laserithmetic*, an educational game which strengthens children's skills in addition, subtraction, multiplication, and division. Several skill levels allow the program to teach these math fundamentals to younger children, and to help enforce good practice skills in older children.

When you start this program, you will be presented with the title screen and a musical theme. Press (ENTER) or (RETURN) to skip the music and continue.

Next you will be asked to enter one of 4 problem types:

- 1) ADDITION
- 2) SUBTRACTION
- 3) MULTIPLICATION
- 4) DIVISION

To select the type of problems you want, simply press the number beside the option. You do not need to press [RETURN] or [ENTER].

Then you will be asked to select one of 4 difficulty levels.

- 1) EASY
- 2) HARD
- 3) HARDER
- 4) REAL HARD

After selecting a level, the computer will display a short message describing what is about to follow:

CAPTAIN,  
YOU'RE THE COMMANDER OF THE SPACE SHIP COLUMBIA.  
DESTROY THE ATTACKING ALIENS BY LOADING YOUR LASER  
CANNON WITH CORRECT ANSWERS TO THE PROBLEMS.

CAUTION! DON'T LET AN ALIEN GET TOO CLOSE!

## The Attack

After selecting these options, you will be ready to start Round 1 of the program. There are 4 rounds in all, each presenting 48 math problems. If you complete all 4 rounds without letting the alien reach your ship, you win.

You can hold the alien at bay by giving the correct answers to the math problems, which are displayed in the middle of your ship. When you enter an answer, it will appear to the right of the problem. If you answer correctly within the allotted time, a laser blast will be fired at the alien, keeping it from moving toward you. If you enter a wrong answer or run out of time, the alien will advance closer toward your ship. With each successive level, the time given to answer a problem is reduced. In addition, the problems get progressively more difficult with each higher level and within each round (on the average). If you don't successfully answer enough questions to keep the alien at a distance, your ship will be destroyed and you will be given your final score and the option to play again or quit.

Notice that the values in the program are not selected totally at random. One of the values used is random, and the other is not. This nonrandom value is determined by 2 FOR-NEXT loops. The outer loop counts from 1 to 12, and will be one of the values used. The inner loop will repeat that value 4 times. This sequence of 48 problems makes up one round. This value may appear on the left or the right side of a problem. The larger of the two numbers will always appear on the left side so that the subtraction problems will always have a positive answer.

Division problems are special—it was necessary to eliminate fractional answers. To do this, the computer multiplies the two numbers selected to arrive at the dividend. One of the two selected numbers becomes the divisor; the other the "unknown quotient."

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





In *Laserithmetic*, a problem's difficulty is determined directly by the skill level. In line 440 of the Apple version, the skill level is multiplied by 4 and then multiplied by a random number (between 0 and 1) to determine one of the numbers in the problem. The higher numbers will tend to make the problems more difficult.

Because the game uses the hi-res screen for its display, a special shape table is created in line 820 for all of the digits. They are arranged such that the number of the shape corresponds to the digit shape codes. Thus, if shape 1 is DRAWn, the digit 1 appears. Shape 2 is the digit 2, and so on. The shape table DATA is contained in lines 950-1020. To use this shape table in your programs, simply copy lines 820 and lines 950-1020 into your program. Note that the shape table is placed in memory locations 24576 through 24787. A very large program might interfere with this section and the shape table would have to be relocated.



The Commodore 64 version of *Laserithmetic* uses a sprite for the alien and character graphics to draw the ship. The sprite DATA is located in lines 1480-1610, and the ship is PRINTed to the screen in lines 1130-1190. The location of the alien sprite is updated in the routine in lines 1410-1470. The MA variable contains the horizontal pixel position of the sprite and is originally set to 254. Each time the sprite moves, MA is decremented by 8. Thus, when the sprite move routine is called, the sprite moves 8 pixels to the left.

Meanwhile, the horizontal character column of the sprite is kept in the SP variable and it is decremented by one each time the sprite moves. The spaceship's laser gun is located in column 12, so if SP is less than 12 the alien has reached the vessel, and the player's ship is destroyed.



The IBM computers have many powerful graphic commands for the BASIC program to use. The DRAW command lets you quickly and easily create a graphic shape. The GET and PUT commands, however, actually allow you to place a known shape on the screen more quickly.

*Laserithmetic* uses these commands together to create and move the alien. The DRAW command draws the alien on the instruction screen. Once there, the GET command places the alien's graphic information in an array. From then on, any time the alien is to be placed on the screen, the PUT command allows for lightning-fast access to the alien's shape by PUTting the same information on the screen via the same array. Using these commands together allows you to attain animation effects that many machines can only access in machine language.



The TI-99/4A version of *Laserithmetic* contains a special input routine which checks every character you type against the characters it expects to see for a right answer. For example, if the right answer to a problem is 144, you would register a wrong answer as soon as you typed 13, because the second character typed was a 3, and the program was looking for a 4.

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

Line Nos.	
100-200	Program header.
210-270	Title screen and music routine.
280-360	Get options and display instructions.
370-390	Start a new round.
400-510	Determine problems and answers.
520-560	Display the problem.
570-720	Get input, evaluate, and take action.
730-810	End of game—option to play again.
820-910	Define and display graphics.
920-1020	Music and graphics data.

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

Line Nos.	
100-200	Program header.
210-300	Title screen and initialization.
310-550	Get options and display instructions.
560-690	Determine and display problem.
700-950	Get input, evaluate, and take action.
960-1100	End of game—option to play again.
1110-1190	Game screen and ship.
1200-1280	Subroutines for sound and action.
1290-1390	Music routine.
1400-1470	Move alien.
1480-1610	Sprite data.

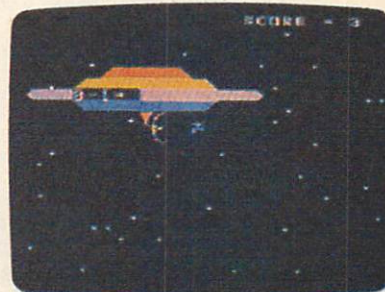
### ***Laserithmetic* (IBM PC & PCjr) Explanation of the Program**

Line Nos.	
100-230	Program header.
240-320	Title, menu, and instructions.
330-400	Begin main game loop; determine and display problem.
410-440	Get input and determine if correct.
450	Do next problem, and display success message if appropriate.
460-500	Right- and wrong-answer subroutines.
510-520	Move alien when time runs out.
530-540	Lose-game routine.
550-590	Display-game-screen subroutine.
600-630	Music and input subroutines.

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

Line Nos.	
100-200	Program header.
210-490	Title screen and initialization.
500-770	Get options and display instructions.
780-940	Start a new round.
950-1130	Determine problems and answers.
1140-1190	Display the problem.
1200-1610	Get input, evaluate, and take action.
1620-1910	End of game—option to play again.
1920-2160	Define and display graphics.
2170-2290	Music and graphics display data.

This task is accomplished by first checking on whether the latest key-input makes the input answer the same number of characters as the actual answer. If so, they are simply compared, and appropriate action is taken. If not, then a FOR-NEXT loop is entered to compare the characters. If any of the characters are not the same, the program immediately reacts with an incorrect-answer response. The alien would advance at this time, and your entry would then be cleared from the screen.



Screen from the TI-99/4A version of *Laserithmetic*



GENERATION 0

GENERATION 1

GENERATION 2

GENERATION 3

GENERATION 4

# THE ORGANIZER REPORTS

In the previous issue of *Home Computer Magazine* (Vol. 5, No. 1) we presented *The Organizer*, an outline-generating program designed to assist you in task solving. We now offer this final addition that allows you to print your organized results.

by William K. Balthrop  
and the HCM Staff

Now that you have had a chance to work with *The Organizer*, wouldn't it be great if you could also get a hardcopy of your file? Well, we promised that we'd show you how, and here's the program to do it. *Organizer Reports* completes the final step in the organizational process—it can give you a listing of your entire outline, just the text, or any number of generations that you specify to be printed out.

*Organizer Reports* has 6 features accessible from the main menu, allowing you to tailor your printout to a variety of specific generations:

- Select Generation (Y/N)?
  - Enter number of generations?
- Indent generations (Y/N)?
  - Indent width (1 to 5)?
- Print Outline (Y/N)?
  - Mark headers?
  - Bold headers?
- Print text (Y/N)?
  - Print Text Headers (Y/N)?
  - Mark headers?
  - Bold headers?
- Format text (Y/N)?
- Printout width (20 to 132)?

## Running The Program

When you run the program, you will be asked to enter the name of an *Organizer* file. After entering the file name, you will be taken to the option-selection screen. There you will be able to input several options to control the output of your report. The first choice for selection is the Maximum Generation option.

SELECT MAXIMUM GENERATION (Y/N)?

When you enter Y for Yes, you will be prompted to enter the number of generations (depth) that you want the program to print out.

ENTER NUMBER OF GENERATIONS?

When you enter a number specifying how many generations you want printed, the program prints all of the generations through the children of the generation specified. If, on the other hand, you entered N for No to the first prompt, the program would not ask for the number of generations and would print the entire outline. (See Figure 1.)

INDENT GENERATIONS (Y/N)?  
INDENT WIDTH (1 TO 5)?

The Indent Generations option will allow you to either indent each generation a specified number of spaces, or left-justify all of your outline. If you answer Yes to the Indent Generations prompt, the Indent Width option will be asked next. You can specify from 1 to 5 spaces of indentation for each generation printed. If, for instance, you wanted your printout to indent each generation 3 spaces, you would enter 3 and the program would automatically indent your outline—how's that for ease of use? (See figure 2.)

There is a limit, however, to the depth of indentations. The program always must have at least 20 character spaces available in which to print. When the program reaches this limit, it will simply stop indenting any farther. All following lines will indent to the same column until a generation that calls for less indentation comes up. For example, if you specify a line width of 80, and an indent width of 5, indentation would stop at column 60, allowing you to indent 12 generations (5 columns per generation).

PRINT OUTLINE (Y/N)?

Entering Y for the Print Outline option will cause all headers in the generations specified to be printed. But first, the next two options will determine how these headers will appear:

MARK HEADERS (Y/N)?  
BOLD HEADERS (Y/N)?





***"In formulating your outline, you could get a printout during the early stages and see at a glance those areas that need more work."***

First, you can direct the program to mark each header by placing a hyphen in front of it, setting it apart from the text. Second, you can have the outline headers printed in bold face. (See Figure 3.) If you don't want the program to print out your entire outline, you can enter N to the Print Outline option and move on to the next option, Print Text.

#### PRINT TEXT (Y/N)?

This option will allow you to print text created with *The Organizer's* text editor. If you select the text option and did not select the above option to Print Outline, then you will be asked whether you want just the text headers to be printed.

#### PRINT TEXT HEADERS (Y/N)?

#### MARK HEADERS (Y/N)?

#### BOLD HEADERS (Y/N)?

If you enter a Y to this question, then only those outline headers which have text will be printed above that text; and, you have the option to set the headers apart from the text with either a hyphen or boldface or both. (See Figure 4.)

#### FORMAT TEXT (Y/N)?

You can also direct the program to format your text to a specified line width. The number you entered for the Indent Width option will be used as the left margin. Your text will be left-justified, although indented. This print format will not change the file itself—only the way that it is presented to the printer. (See figure 5.)

#### PRINTOUT WIDTH (20 TO 132)?

This last option lets you set the column width of the actual printout of your outline. The minimum width you can set it to is 20 characters per line, and the maximum width is 132 characters per line. The number that you enter will be used by the program in formatting your text and for determining how far the report can indent, if the Indent Generations option was selected. (See Figure 6.)

### A Whole New Look

The first time you see a hardcopy of your entire program, you will realize just how useful *The Organizer* really is. We at *Home Computer Magazine* see *The Organizer* as a simple method for keeping track of almost any type of home, business, or school project, and *Organizer Reports* makes the process of getting started easier and more accessible. In formulating your outline, you could get a printout during the early stages and see at a glance those areas that need more work. The Select Generations option could be used to generate a very general report such as a table of contents. You could then print more detailed reports as the need arose. With this multitude of formatting options, *Organizer Reports* is a very flexible tool which will become an indispensable addition to *The Organizer* program family.

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

**Figure 1**

Entire remodel outline indented 3 spaces and including text.

```
Contractor supplies
Labor
All labor will be completed by the
contractor by the contracted date;
but if additional work is necessary,
adjustments to the contract will be
made by agreement of both parties.
New floor
  Replace rotten joists
  Install moisture barrier
  Lay plywood
  Smooth with grout
  Lay tile
  Install cabinets
Plumbing
  Install sink
  Connect drains
  Connect faucets
Wallpaper
  No gap in seams
  No overlap
  Fit edges within 1/8"
Trim work
  Trim floor
  Trim ceiling
  Trim cabinets
Wiring
  Install light above sink
  Replace all switches
Bonding
  To legal minimum
  Ascertain what the local state law
  requires and ask contractor to
  show proof of bond.
Liability for damage
  To existing structure
  To cabinets
  To new materials
  To new fixtures
Late penalty
  After date -----
  Unless revised by owner
Owner supplies
Labor fee
  Total amount $-----
  40% down
  60% down on completion
Cost of materials
  Only on approved list
  Not to exceed estimate
Revisions
  Proposals for change in original
  list must be submitted in writing
  and approved by owner with
  signature.
Blueprints
Materials list
  Nails
  Grout
  Glue
  Wallpaper
  Lumber
  Floor tile
  Plumbing fixtures
  Wire
  Electrical fixtures
Misc.
```



**Figure 2**

*A portion of the outline with no indentation and the same portion with indentation.*

Contractor supplies  
Labor  
New floor  
  Replace rotten joists  
  Install moisture barrier  
  Lay plywood  
  Smooth with grout  
  Lay tile  
  Install cabinets  
Plumbing  
  Install sink  
  Connect drains  
  Connect faucets  
Wallpaper  
  No gap in seams  
  No overlap  
  Fit edges within 1/8"  
Trim work  
  Trim floor  
  Trim ceiling  
  Trim cabinets  
Wiring  
  Install light above sink  
  Replace all switches  
Bonding  
  To legal minimum  
  Liability for damage  
  To existing structure  
  To cabinets  
  To new materials  
  To new fixtures  
Late penalty  
After date \_\_\_\_\_  
Unless revised by owner

Contractor supplies  
Labor  
  New floor  
    Replace rotten joists  
    Install moisture barrier  
    Lay plywood  
    Smooth with grout  
    Lay tile  
    Install cabinets  
  Plumbing  
    Install sink  
    Connect drains  
    Connect faucets  
  Wallpaper  
    No gap in seams  
    No overlap  
    Fit edges within 1/8"  
  Trim work  
    Trim floor  
    Trim ceiling  
    Trim cabinets  
  Wiring  
    Install light above sink  
    Replace all switches  
Bonding  
  To legal minimum  
  Liability for damage  
  To existing structure  
  To cabinets  
  To new materials  
  To new fixtures  
Late penalty  
After date \_\_\_\_\_  
Unless revised by owner

**Figure 3**

*These two examples show marked headers, and bold headers with no text.*

-Contractor supplies  
  -Labor  
    -New floor  
      -Replace rotten joists  
      -Install moisture barrier  
      -Lay plywood  
      -Smooth with grout  
      -Lay tile  
      -Install cabinets  
    -Plumbing  
      -Install sink  
      -Connect drains  
      -Connect faucets  
    -Wallpaper  
      -No gap in seams  
      -No overlap  
      -Fit edges within 1/8"

Contractor supplies  
Labor  
  New floor  
    Replace rotten joists  
    Install moisture barrier  
    Lay plywood  
    Smooth with grout  
    Lay tile  
    Install cabinets  
  Plumbing  
    Install sink  
    Connect drains  
    Connect faucets  
  Wallpaper  
    No gap in seams  
    No overlap  
    Fit edges within 1/8"

**Figure 4**

*Text only printed along with the headers specific to it.*

- Labor  
  All labor will be completed by the contractor by the contracted date; but if additional work is necessary, adjustments to the contract will be made by agreement of both parties.  
- To legal minimum  
  Ascertain what the local state law requires and ask contractor to show proof of bond.

**Figure 5**

*Text only printed along with the specific headers, with a column width of 20 spaces.*

Labor  
All labor will be completed by the contractor by the contracted date; but if additional work is necessary, adjustments to the contract will be made by agreement of both parties.  
To legal minimum  
Ascertain what the local state law requires and ask contractor to show proof of bond.



**Figure 6**

The entire outline formatted  
with a column width of 40 spaces.

**Contractor supplies**

**Labor**

All labor will be completed by the contractor by the contracted date; but if additional work is necessary, adjustments to the contract will be made by agreement of both parties.

**New floor**

- Replace rotten joists
- Install moisture barrier
- Lay plywood
- Smooth with grout
- Lay tile
- Install cabinets

**Plumbing**

- Install sink
- Connect drains
- Connect faucets

**Wallpaper**

- No gap in seams
- No overlap
- Fit edges within 1/8"

**Trim work**

- Trim floor
- Trim ceiling
- Trim cabinets

**Wiring**

- Install light above sink
- Replace all switches

**Bonding**

- To legal minimum
- Ascertain what the local state law requires and ask contractor to show proof of bond.

**Liability for damage**

- To existing structure
- To cabinets
- To new materials
- To new fixtures

**Late penalty**

- After date

-----  
Unless revised by owner

**Owner supplies**

**Labor fee**

- Total amount \$ \_\_\_\_\_
- 40% down
- 60% down on completion

**Cost of materials**

- Only on approved list
- Not to exceed estimate
- Revisions
- Proposals for change in original list must be submitted in writing and approved by owner with signature.

**Blueprints**

**Materials list**

- Nails
- Grout
- Blue
- Wallpaper
- Lumber
- Floor tile
- Plumbing fixtures
- Wire
- Electrical fixtures
- Misc.



This Apple program can be divided into two major sections—option selection and output. The option-selection section prompts the user for a number of formatting options to be used in the final report. The order in which the options are selected is not always the same. Depending on previously selected options, some options no longer apply, and so will not be asked.

The method used to determine which options are asked is quite simple. The description for each option is contained in the string array `PR$( )`. There are ten elements, one for each option.

Similarly, there is another array that contains the responses and defaults for each option. This is the `RAS( )` array. If you simply press `[RETURN]` at each prompt, the array will not be updated, so the last value contained there will be used.

By placing both the options and the responses in an array, you can use any prompt at any time simply by changing the value of the variable used to index into the array. The `SL` variable does this. After each prompt is completed, the value `SL` is updated to the next prompt that should be provided. Tests are often made to determine what the next prompt should be.

Two input routines are used to get the option information. Each routine is designed to accept only one kind of specific information. The routine that starts at line 1220 will input any integer numeric value from 1 to 4 digits. It is used for options requiring a value such as the width at which you want the report formatted. The second routine starts at line 1360 and will only accept a `Y` or `N` response for the Yes/No options.



After using your Commodore *Reports* program, you may notice that the text created with the text editor portion of *The Organizer* has been reformatted. The file you created with *The Organizer* has not changed at all, only the output you see at the printer has been modified. This formatting is done in a string array inside the *Reports* program.

The `PS( )` array will contain one page of formatted text and outline. All information that will go to the printer is first stored in this array, like a large buffer.

As the records are read into memory from the data file, they are manipulated according to the options selected on the option screen. The manipulated text is then placed into the `PS( )` array. This includes special characters to turn the bold typeface off and on, and the spaces used to indent records.

This array makes text formatting simpler. As a text record is read into memory, the program checks the current line that it is working on in the array. The amount of free space on the line is determined by subtracting the length of the line from the maximum allowable length. It then searches through the text just read into memory and locates the first word on the line. If the word will fit onto the line in the array, it is added to the array line and removed from the line that was input from the file. This process is then repeated until no more words will fit on the line in the array. At this time the program simply moves down to the next line in the array and starts over. When the program has finished filling 60 lines in the array, it will branch to an output routine which will print all 60 lines to the printer and eject to the top of the next page. The array is also cleared at this time so that the next page of text can be built there.



This program takes files created with the IBM PC and PCjr version of *The Organizer*, which was published in Vol. 5, No. 1, and reformats them for output to the printer. The file itself is not changed; the program does all of its formatting internally, and then forgets that information when it is done.

One printer page at a time is formatted (60 printer lines). After formatting the page, it will be printed, and the paper will be advanced to the top of the next page. This formatting is accomplished using a string array with 60 elements, `PAGES()`.

As the program reads each record from the file, it builds the `PAGES()` array until it is full. It then prints the entire array, clears the array, and returns for more information.

Each record in the file contains text from either the outline or the text editor. If the text is from the text editor, it will all be "scrunched" together to fill out each line to its maximum length.

After reading in the text from a record in the file, the program checks the current line it's working on in the `PAGES()` array to see how much free space is left at the end of the line. It does this by subtracting the length of the line from the maximum allowable length determined by the options selected on the option page. If there are, say, 22 spaces left on the line, the program then starts at the 22nd character of the line that was input and scans backward until a space is found. The space indicates a break between words. The remainder of the line to the left of the space is then relocated to the `PAGES()` array.



TI Extended BASIC has a few commands built into it which make it a very powerful language.

The `ACCEPT AT` statement allows the program to accept data input from any location on the screen. You can limit the field of input to 1 to 28 characters, and can have the computer give a beep when the statement is executed.

Two features of the `ACCEPT AT` statement used in this program allow you to leave default values in the input field, and validate the type of characters that can be entered.

Default values can be placed into the input field by using the `SIZE` option of the `ACCEPT AT` statement. The size option determines the size of the input window in characters; `SIZE(1)` would allow only one character to be entered. If you specify a negative value as the size parameter, then any characters already on the screen where the input window appears will be kept in the window. This program supplies defaults for all of its inputs in this manner. If you would like to have your program use different defaults, you can change them in the data statements in lines 1150 through 1180.

The second feature used is the ability to limit input to only certain types of characters. In line 1080 the `VALIDATE` option is used to do this. By placing those characters which are legal into a string or directly into the statement, as was done here, you can specify only those characters to be legal. If users try to type anything else, they will be greeted with a loud beep, and the character will be ignored. You can specify groups of characters (as done in line 1090) with parameters like `DIGIT` within the `VALIDATE` option. This parameter will only allow the entry of the numbers 0 through 9.

HCM

### **The Organizer - Reports (Apple II Family) Explanation of the Program**

Line Nos.	
100-240	Program header.
250-370	Main control loop.
380-800	Initialize program.
810-860	Routine to print heading.
870-1440	Display options and get input.
1450-1640	Main control loop for printing the report.
1650-1770	Report abortion messages.
1780-2010	Set up the print buffer.
2020-2130	Output print buffer to the printer.
2140-2550	Format one line of the buffer.
2560-2690	Main control for file name entry.
2700-3000	File-handling routines.
3010-3080	Clear screen.
3090-3140	Get character.
3150-3230	Return to the organizer.
3240-3430	Error routine.
3440-3660	Check limits on file name during entry.

### **The Organizer - Reports (C-64) Explanation of the Program**

Line Nos.	
100-250	Program header.
260-420	Main program-control routine.
430-550	Main control for printing report.
560-650	Main control for building a page of text.
660-690	Print a page to the printer.
700-810	Preliminary text formatter.
820-1020	Formatter.
1030-1690	Get options.
1700-1710	Cursor-control routine.
1720-1750	Initialize the screen.
1760-1800	Get file name.
1810-1860	Initialize variables.
1870-2020	File-handling routines.
2030-2140	Error routine.
2150-2160	Return to main menu.

### **The Organizer - Reports (IBM PC & PCjr) Explanation of the Program**

Line Nos.	
100-240	Program header.
250	Initialize the error routine.
260-290	Display title screen.
300-370	Load file links.
380-430	Main control routine.
440-760	Option input screen.
770-910	Main control for printing the report.
920-990	Main control for building the page buffer.
1000-1170	Format text.
1180-1260	File-handling routines.
1270-1340	Initialize variables.
1350-1380	Exit back to main menu.
1390-1470	Error-handling routine.

### **The Organizer - Reports (TI-99/4A) Explanation of the Program**

Line Nos.	
100-250	Program header.
260-350	Initialize program.
360-430	Load file links.
440-750	Get options for report.
760-890	Main control routine for the report.
900-930	Main control routine for formatting a page.
940-1050	Format text.
1060-1070	Output a page of text to printer.
1080-1090	Input subroutines for the options.
1100	Clear the page of text for the next page.
1110-1140	File-handling routines.
1150-1180	Data for the option screen.
1190	Exit back to main menu.
1200	Error for lack of memory expansion.
1210-1310	Error-handling routine.



# 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

**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•prēs•teh•dī•jeh•tā•shūn) —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 Music of Sound

## A Review of 4 Music/Sound Software Series with Keyboards for the C-64

by Wayne Koberstein

HCM Staff

Suddenly, everyone seems to have discovered the Commodore 64's musical potential—a potential inherent in its Sound Interface Device (SID). This chip gives the C-64 a more powerful sound than anything possible on an unaltered Apple II, IBM, or Texas Instruments home computer, and has the capabilities of a full-range synthesizer costing thousands of dollars. Many C-64 programmers employ SID to enhance their software with some of the richest sounds possible on a home computer. Users have enjoyed these sounds in many games and other programs not specifically focused on sound and music—but their enjoyment has been mostly passive. Without highly-developed programming skills, the ordinary user has had no way to actively tap SID's wealth of synthetic sound. [If you are unfamiliar with SID and synthesizers in general, turn to "Commodore Hornblower" in this issue before reading on—Ed.]

Recently, several companies have each premiered their own series of far-ranging packages designed to turn the C-64 into a relatively inexpensive synthesized sound and musical instrument. These products include piano-like keyboards and software, and their main purpose is to "humanize" the powerful SID chip. Some have gone even farther—setting up a whole system of sound and music composition, from playing, to transposing, to actually printing musical scores digitally recorded on the Commodore machine.

Undoubtedly, other similar products will continue to emerge as time goes on. But for now, I will review the packages available from four different companies: Sight & Sound Music Software, Sequential Circuits, Waveform, and Melodian.

### Getting the Keys

Using a computer as a musical instrument is not easy if all you have to work with is the computer's own typewriter-like keyboard. Thus, all of the companies reviewed here begin by providing piano-like keys to actually play the instrument. These products range from a simple keyboard overlay to separate units with elec-

tric piano-like keys. However, the real value of each company's products does lie in the software; it is also by the total value of this software that we will rate each company's line.

Distributor:	Sight & Sound Music Software, Inc. 3200 S. 166th, P.O. Box 27 New Berlin, WI 53151 1-800-558-0910			
Name:	Incredible Musical Keyboard (\$39.95) Music Processor (\$34.95) Computer Song Albums (\$19.95 ea.) Kawasaki Synthesizer (\$39.95) Kawasaki Rhythm Rocker (\$34.95) 3001: A Sound Odyssey (\$34.95)			
System Requirements:	Disk drive (joystick for Music Processor and 3001.)			
Overall rating:	Poor	Fair	Good	Excellent

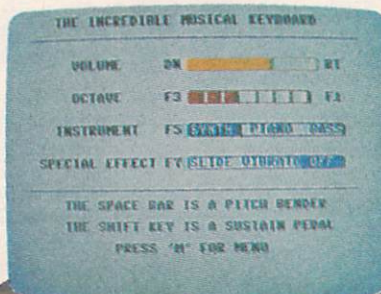
### Sight & Sound Music Software

Sight & Sound offers a fairly flimsy plastic overlay that sits on top of the C-64 console. Called *The Incredible Musical Keyboard*, it is somewhat incredible for its lack of quality—even considering its relatively low price (overlay and basic software for \$49.95). This overlay consists of 24 small black and white keys, each about one-third real piano size. As with all the keyboards reviewed here, you can switch the range of the keys up or down a full 8 octaves. Sight & Sound is correct in warning against picking up the overlay by the keys themselves—they are likely to break right off the mount. It is tempting to tape the whole thing down—but many operations require picking up the overlay in order to press certain letter keys, so taping is really no solution. Despite the ephemeral nature of its hardware, Sight & Sound does provide some of the most innovative software packages, taking some unexpected but creative approaches to music and sound. As listed in the review box, 5 software packages are available from this company.

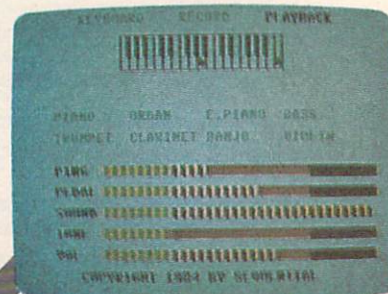
Keyboards clockwise from the top:  
Melodian, MusicMate, Colortone,  
Incredible Musical.



The Incredible Musical Keyboard  
from Sight & Sound.



The MusicMate Keyboard  
from Sequential.





*For the musical beginner and experienced sound buff alike—the hills are alive with the Sound of SID, Commodore's amazing sound chip . . .*

The *Incredible Musical Keyboard's* basic software turns your Commodore into a simple instrument—much like a small home organ—with which you can play up to three notes at a time (using all 3 voices of the SID chip). Its "sound" controls are limited to volume, octave range, a choice of 3 instrument sounds (piano, bass, and "synth"), and two "special effects": vibrato, and sliding pitch between one note and another. (This latter effect has a small bug: If you press one note and then another without releasing the first, the computer seems to briefly stick in a wavering "siren" effect.) The space bar also serves as a "note bender," causing a note played on one of the overlay keys to glide up in pitch.

Another Sight & Sound software package, the *Music Processor*, provides a selection of "computer songs" with which you may play along or manipulate by changing the sound of one "voice." As a song plays, the screen displays a score in traditional musical notation—traditional, except that the notes continuously zip up and down the staff. To affect the voice playing the melody, you can use the joystick to select from 99 preset sounds, ranging from near-normal "instruments" to wild cacophony. As you play along, what you play is automatically recorded so that when you stop, the computer will play back your solo line along with the other two song parts. Sight & Sound markets a line of "computer albums"—such as *On Stage* and *Music Video Hits*—that will load into the *Music Processor*, providing a wider selection of songs.

*Music Processor* is something of a weird hybrid, with one approach for true beginners, and another for semi-skilled musician/programmers. In the second part of the program, you may create original scores and sounds by "programming" line-by-line using the Processor's BASIC-like commands. For example, the command VI 2 imposes a vibrato effect on Voice 2. This is not much of an advance over what is already available—other products, such as *Simon's BASIC* (see "Simon Sez" in Vol. 5, No. 1 of *HCM*), address the SID chip with relatively easy commands. However, a dedicated musical programming language is an interesting idea, and the

*Music Processor* may lure even some beginners into giving it a try.

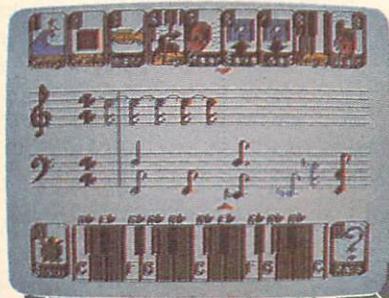
### Ah, Kawasaki!

Coming at SID from a completely different perspective, Ryo Kawasaki livens up the Sight & Sound show with 3 programs—all exhibiting a strongly original and creative approach to music, sound, and graphics as well. His *Kawasaki Synthesizer* is divided into two parts: the *Composer* and the *Performer*. A separate program, *Rhythm Rocker*, is a souped-up version of the *Performer*.

With the *Composer*, you may create sounds using most of SID's capabilities. You may also record original material or form "sequences" using a special notation. Bass and rhythm patterns can also be created with *Composer*, saved to disk, and transferred to the *Rhythm Rocker*. With the *Performer*, you get a playing instrument only—there are no recording features or any way to bring sound patches over from *Composer*. On the other hand, you may select from 21 very interesting preset sounds, play-along with a bass or rhythm track, listen to a selection of songs on disk, or watch the animated *Space/Dance Theater* perform to a tune. To some extent, you can alter the sound settings by changing between SID's four wave forms, adding vibrato, bending notes, or glissandoing rapidly up the scale.

*Composer* is a fairly powerful synthesizer, and *Performer* is a good warm-up, but the *Kawasaki Rhythm Rocker* is the real star of the show. *Rhythm Rocker* is similar to *Performer* in its playing and sound-modification features, although its preset sounds are a bit wilder. *Rocker* is like a personal discotheque with "light show" graphics and a vibrant screen display of the musical keyboard that highlights notes as you play them. Here, musician/programmer Kawasaki's personality really comes through. He believes in *creating* music, rather than *practicing* it, and he has designed this program for the sheer fun of creating sounds and graphics. You can simply play along with contemporary-sounding bass and rhythm tracks, or

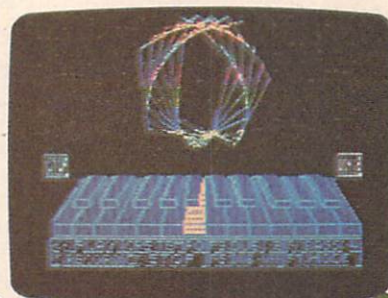
Colortone Keyboard  
from Waveform.



Music Processor  
from Sight & Sound.

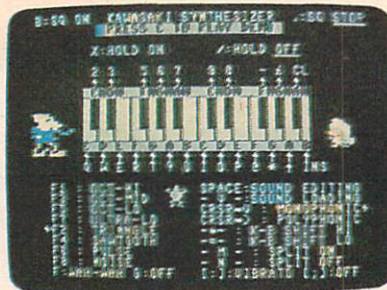


Kawasaki Rhythm Rocker  
from Sight & Sound.





Kawasaki Composer  
(playing screen).



"dub" up to two tracks on your own and then play along with them. If you choose to play a rhythm track, the screen changes, and each percussive sound (one for each musical key) is accompanied by a hi-res graphic effect. A whole set of control keys manipulates a series of intricate and colorful drawings that hover or pan across the screen. One word can describe the *Rhythm Rocker*, and that word is *spectacular*.

It's a tough act to follow, but Sight & Sound has one more addition to its stable of sound programs: *3001: A Sound Odyssey*. This package is a full-range sound synthesizer, with a simple but thorough tutorial that clearly explains the elements of sound. *3001*, discussed in detail later, is one powerful package.

Distributor:	Sequential Circuits, Inc. 3051 North First Street San Jose, CA 95134 (408) 946-5240
Name:	MusicMate Keyboard (\$99) Sound Maker (\$39.95) Song Builder (\$39.95) Song Editor (\$39.95) Song Printer (\$39.95)
System Requirements:	Disk drive
Overall rating:	Poor Fair Good Excellent

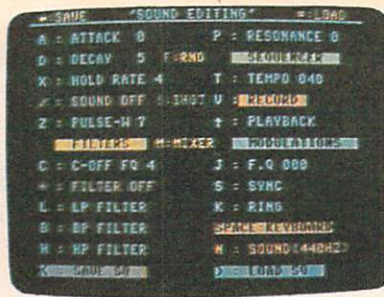
## Sequential Circuits

Already a well-known name among sound synthesizers, Sequential has entered the home market with their *MusicMate* keyboard and *MusicWare* software. Sequential's keyboard is not an overlay, but a separate 2-1/2 octave unit that plugs into the C-64's joystick port. Its piano-like keys are full size with fairly good action, although their springy metal mounts feel a bit flimsy for long-term use. But at twice the price of Sight & Sound's *Incredible Musical Keyboard*, the basic package for *MusicMate* is still more than twice the value.

*MusicMate* itself simulates a small home organ (with a built-in recorder) very successfully, but it's only a warm-up for Sequential's integrated music software series, marketed in 4 separate packages (as listed in the review box). Each program's title states its purpose; we will discuss each in detail a bit later.

*MusicMate*'s basic software allows you to play live (polyphonic), or record and playback your compositions. You can select from 8 preset instrument sounds and can modify these sounds using 5 controls. Besides Volume and Tone, these controls include Ping, Pedal, and Sound. Ping is similar to "attack," Pedal corresponds to "sustain," and, although the manual says that Sound "adds noise," it seems closer to "resonance" in its actual effect.

Most of *MusicMate*'s preset "instruments" sound very little like the real instruments they imitate, because only a few elements of a given instrument's sound—such as waveform and envelope—are actually simulated. SID



Kawasaki Composer  
(sound-editing screen).

when *imitating* real sounds. A synthesizer's real forte is *originating* sounds that no one has heard before. This basic principle holds for all of the programs we are discussing here, no matter how well they may exploit the C-64's amazing sound chip.

is capable of much more, although no synthesizer—even much more complicated ones than this—is at its best

Distributor:	Waveform Corporation 1912 Bonita Way Berkeley, CA 94704 (415) 841-9866
Name:	Colortone Keyboard (\$39.95) MusiCalc 1—Synthesizer & Sound Teacher (\$29.95) MusiCalc 2—ScoreWriter (\$19.95) MusiCalc 3—Keyboard Maker (\$19.95)
System Requirements:	Disk drive
Overall rating:	Poor Fair Good Excellent

## Waveform Corp.

Not everyone wants to learn the piano just to play the computer—so enter the *Colortone*, a "keyboard" that is really a thin touchpad designed to play only the "right" notes. *Colortone*'s design reflects a philosophy echoing Sight & Sound's *Kawasaki*: that it is better to give someone a way to immediately *create* music, than to start off with hours of tedious *practice*. Thus, *Colortone*, with its basic software, is a vehicle for "no-fault music," making it relatively easy for anyone to play the computer, even without any previous musical training.

*Colortone*'s keyboard is designed to work with the *MusiCalc* series of software listed in the review box.

*Colortone*'s touchpad playing surface is divided into three areas: a row of painted black and white keys, a slide bar (called the "Touch Harp"), and a row of special function keys. Its basic software allows you to listen to or play one part of any song prerecorded on disk. If you

play on the Touch Harp, only the notes fitting the scale of sharps and flats for that particular song will be heard. (You can override this, if you wish, by selecting one of the scales from another song or playing the black and white keys.) A musical staff displays notes as they are played, scroll-

ing them across the screen, and a keyboard display places an orange dot over each key as you play its corresponding note.

Whatever you play is recorded automatically along with the other two parts of the song provided by the program. You can then play it back, or save it to disk. If you wish, you can record the song at one tempo and play it back at another.

Unfortunately, *Colortone* is limited by its small selection of songs, which may lead to early boredom. It also has no provision for recording all three parts of a song—which would allow you to store your own original

**"Colortone's design reflects a philosophy echoing Sight & Sound's Kawasaki: that it is better to give someone a way to immediately create music, than to start off with hours of tedious practice."**



ConcertMaster  
from Melodian.



compositions—or for loading more song selections from another disk. Sound modification is limited to a selection of instrument-like sounds that cannot be adjusted. For a serious music student, *Colortone* may not take the place of the piano or any other “non-no-fault” instrument. Yet, if what you want is an immediately enjoyable creative tool, perhaps in the tradition of the old Sears Autoharp, *Colortone* may just be its modern equivalent.

Distributor:	Melodian, Inc. 115 Broadway, Suite 1202 New York, NY 10006 1-800-MELODIA
Name:	Melodian Keyboard & ConcertMaster (\$169.95) RhythmMaster (\$39.95)
System Requirements:	Disk drive
Overall rating:	Poor Fair Good Excellent

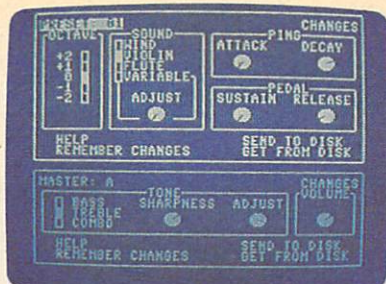
## Melodian

In terms of basic hardware, *Melodian* has the best package of the lot. Its 40-key (over 3-octave) keyboard is solidly constructed, with good key action comparable to a small home organ. At present, only two software packages are available in the *Melodian* series: the *ConcertMaster*, and *RhythmMaster*. Both are straightforward, no nonsense programs that make cautious but good use of Commodore's sound chip.

*ConcertMaster* is *Melodian*'s synthesizer and recorder. It allows you to play along with 35 different prerecorded songs, including several bass lines. You may play and record—either in accompaniment with one of these songs, or up to 3 tracks of your own compositions—using preset “instruments.” You may also modify any of the preset sounds, adjusting a somewhat limited selection of controls affecting the SID parameters. Any adjustments can be saved to or recalled from disk.

“Clean” is perhaps the best word to describe *ConcertMaster*'s arrangement of menus, control fields, and score display. Most operations and adjustments are easily executed with the function keys, which seems to fit *ConcertMaster*'s simple look. Unfortunately, interesting sounds do not seem to pour out of this program. Its preset sounds are without much flavor; even the bass lines are, well . . . wimpy. This conservative sound may be due to the program's conservative design, which does not access some of the more exotic effects possible with SID. As a playing and recording device, however, *ConcertMaster* is on very solid “sound.”

*RhythmMaster* is more of a game than a creative tool. The computer plays musical or rhythmic phrases, and the player is to respond by imitating each phrase. This can be a very entertaining and also an effective way to



Sound Maker  
from Sequential.

build melodic and rhythmic skills. *RhythmMaster* certainly increases the use and the value of the *Melodian* key-

board/software line—and it is an example of the increasing variety of software emerging for the Commodore sound machine.

## Processing Sound

Although all of the companies we've discussed provide basic playing and recording software, they also hold the keys to another level of instrumentation: sound synthesizing. Each company has at least one product specifically billed as a sound synthesizer, and each of these products provide a unique interface with SID. All are varied in their approach and the degree to which they each exploit the sound chip's full potential. To make it all more visible, we have included a chart showing the number of “SID's Tricks” that each program can perform. (See Figure 1.)

To start off, Sight & Sound's *Kawasaki Synthesizer* is a good, basic synthesizer interface with fairly straightforward controls. Most adjustments are made by changing numeric values or selecting options with specific command or function keys. These are easy to learn, and the program soon becomes very fast to operate. Although the *Composer* initially provides 4 preset sound-settings, it does not attempt to imitate “real” instruments. Its selection of basic SID functions is about the same as what the *Melodian* package offers, but it does add a little seasoning with a few built-in effects, such as wah-wah, vibrato, hold, a random sound generator, and a preset “shot” envelope.

Unfortunately, you cannot both play the keyboard and make major sound adjustments on the sound-editing screen at the same time. Thus, one option of sound editing is a 440 Hz “test” note that allows you to hear the effect of your adjustments—which can be saved to or loaded from “Sound Libraries” on disk. (One factor in disk operations, as stated in the manual, is that certain sound settings—such as ring modulation—cannot be saved, and must be reset after loading.)

When playing the keyboard, you have a choice of monophonic (one note at a time) or polyphonic (3 note) modes. We found, however, that in polyphonic mode, some keys sound intermittently when 3 are pressed together. The keys also

tend to sound a second time when released.

Despite these problems, *Composer* is a powerful, inspiring tool for true sound synthesis—and, in conjunction with *Kawasaki*'s other programs, this computer instrument strikes a nearly perfect note.

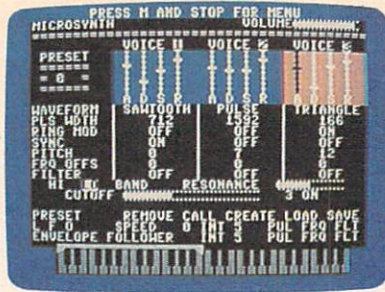
## Sound Maker

Sequential's entry in the home market for sound making on the C-64 is appropriately called *Sound Maker*. This program has a simpler format than *Kawasaki*, but like *Melodian*, it seems to be more limited in the variety of sounds that it can easily produce. Sequential markets some very sophisticated professional synthesizers, but

“Sequential markets some very sophisticated professional synthesizers, but for the home user, the company has apparently chosen a much narrower path.”



3001: A Sound Odyssey  
from Sight & Sound.



for the home user, the company has apparently chosen a much narrower path.

*Sound Maker* and *Song Builder* work together to arrange and record songs. Although sound settings created with *Sound Maker* affect all 3 voices at once, these settings can be loaded into *Song Builder*, where each voice can have a separate sound setting. Sequential's total package is more an ambitious *music* series than a primarily sound-creating program—and that may explain *Sound Maker*'s more limited format.

*Sound Maker* uses unique terms for some of the SID chip functions. For example, instead of calling each waveform by its technical name, the editing screen gives you a choice of "violin" for sawtooth, "flute" for triangle, "wind" for noise, and "variable" for pulse. In some cases, this less-technical nomenclature helps to understand what's happening—as when it divides the envelope settings into "ping," with controls for attack and decay, and "pedal," with controls for sustain and release. In other cases—as when low-pass, high-pass, and band-pass filters are referred to as "bass, treble, and combo"—the terms are unnecessarily vague.

## ConcertMaster Revisited

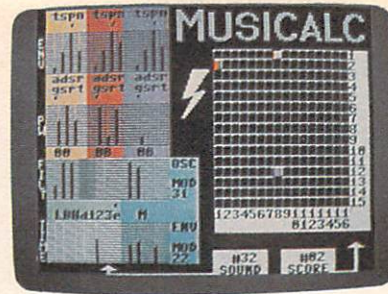
We have already discussed the *Melodian ConcertMaster* to some extent, because this software package is also the basic package for the *Melodian* keyboard. It, like *Sound Maker* and even the *Kawasaki Composer*, is somewhat "flat" sounding, and—though it is a very nice playing and recording tool—cannot compare as a synthesizer to the two programs we have yet to discuss: *3001: A Sound Odyssey* and *MusiCalc I*, *Synthesizer* and *Sound Teacher*. This fact need not discourage someone more interested in *music* than in *sound* making from employing *ConcertMaster*, *Sound Maker*, and *Composer*. In fact, it may help to have some experience with these simpler programs in order to understand what is possible with the more complex sound tools. For real synthetic sound buffs, *3001* and *MusiCalc I* represent the current state-of-the-art in software synthesizers.

Obviously, the major factor in these state-of-the-art packages is that they simply do *more* to exploit the C-64 sound chip. In fact, with these programs, you can make nearly full use of SID's potential as a synthesizer. But what is the significance of the added features these programs provide?

## A Full Bag of Tricks

Probably the most important features contributing to the richer sound of *3001* or *MusiCalc I* are modulation and pitch control. Both programs allow for modulating various sound components by using either the oscillation or the envelope of Voice 3 to affect, for example, the frequency (or the filtering) of another voice. *3001* calls these two types of modulation LFO (Low Frequency Oscillator) and Envelope Following. *MusiCalc* refers to them as simply Oscillator and Envelope Modulation.

*MusiCalc*'s whole system is hard to comprehend until you've banged around in it for awhile. Once you learn your way, you will realize just how much it can do—but it really tries to do so much that it can all be pretty overwhelming at first. This also holds true for the manual, which is a little like a comprehensive almanac



MusiCalc I  
from Waveform

Using one oscillator to modulate another produces a wavering tone, which is especially noticeable when

the frequency of the *modulating* oscillator is very low—hence, the term LFO. Using the *envelope* of one voice to control the frequency of another can produce, as a familiar example, a siren effect. These modulators can also be used to control other elements, such as the cutoff frequency of a filter or the pulse width. This is the extent of modulation possible with *3001*. *MusiCalc* opts for a much more flexible modulation system, in which almost everything can be modulated. It individually addresses all of the SID chip registers, and produces pretty wild, almost random effects. If you want to fully understand how this system works, it helps to first read the explanation of SID registers in the *Commodore 64 Programmer's Manual*.

Pitch control is another way of adding spice to a sound. For every note played, a voice will produce an oscillation at the frequency of that note. This frequency—call it the *center* frequency—can vary up or down depending on the value the program *POKEs* into the proper location in the SID chip. Consequently, it is possible to "temper" a voice by off-tuning its center frequency from a few hertz (vibrations per second) to more than an entire octave in either direction. Two oscillators tuned just a few hertz apart will tend to interact in such a way that they produce a *third* frequency, usually in

the low range. This makes it possible to simulate the kind of harmonics produced by an acoustic instrument, such as a guitar or flute. Slight off-tuning of multiple oscillators increases the chances of creating more such harmonics. This off-tuning can also form whole *chords* by

tuning the voices to the proper intervals. *3001* allows adjustment of the center frequency by pitch in musical half-steps, or by frequency, a much finer scale. *MusiCalc I* allows off-tuning by half-steps only.

Both of these programs come with a selection of preset sounds which can be loaded from disk. (Both will also save your own settings to a work disk.) For each one of its preset sounds, *MusiCalc* provides a prerecorded song. Not so with *3001*—although most of its 100 preset sounds are extremely rich sounding compared to anything I heard in the other programs I reviewed.

For several reasons, *3001* seems to be more immediately satisfying than *MusiCalc I*. Perhaps it is because Sight & Sound has focused their approach on sound synthesizing, without trying to create a much larger integrated package like the *MusiCalc* series. *3001* may be more limited in its ultimate application, but it is clearer in its intent—making it a much more *usable* synthesizing instrument.

*MusiCalc*'s whole system is hard to comprehend until you've banged around in it for awhile. Once you learn your way, you will realize just how much it can do—but it really tries to do so much that it can all be pretty overwhelming at first. This also holds true for the manual, which is a little like a comprehensive almanac



with no index. If you read everything, you stand a chance of finding what you're looking for. And for some information, it helps—as mentioned before—to have yet another source.

## Recording & Sequencing

After sound synthesis, the ability to record or create song sequences may be the greatest contribution these packages make to home computing. Some have elaborate systems to enter sequences of notes and build long songs. Others automatically record whatever you play and then play it back at various tempos.

A computer records by storing digital information representing the recorded sequence. What is interesting is that varying the *tempo* of a digital recorder does not vary the *pitch* of the playback—as it would on an analog recorder, such as a tape deck. Thus, you can record at a very slow speed and play back at a high speed and your song won't sound like the Chipmunks' backup band. This is but one way in which digital recording potentially can be manipulated because, like a computer program, it ultimately consists of nothing but numbers, which do not have to obey the laws of physics.

A *sequencer*—like the one used in the *Kawasaki Synthesizer*—is technically different than a *recorder*. On a sequencer, the notes can be entered carefully one-by-one instead of "played." *MusiCalc I* employs such a system, although its method of building a graph-like musical representation seems overly complicated. *Kawasaki* simply lets you enter a series of notes like building a string of pearls. (*MusiCalc* also lets you record one voice at a time.)

Sequential's *Song Builder* and the *Melodian ConcertMaster* have the best recording features, allowing you to build up three recorded layers or "tracks" by playing one voice at a time while listening to the tracks already recorded. Both programs provide a simple and easy-to-use format. *3001*, not primarily a recording instrument, records all three voices at one time (one key plays all 3 voices). It also has a unique "Player Piano" method of assigning up to 32 preset sounds to one function key. By adjusting the pitch of the 3 voices to different intervals with each preset, and rapidly changing between presets with the function key, you can play and record fairly complex songs—limited though they are in length.

These programs with their ability to record music on a home computer, could be—as they say—the start of something *big*. Your computer may some day have the ability to simulate a small multi-track recording studio, even handling sounds from other outside sources, such as "real" instruments. An emerging system called MIDI (Musical Instrument Digital Interface) could be one vehicle for bringing this about. (See "The Magic of MIDI" sidebar on this topic.) Hopefully, this new use for the home computer will continue to gain momentum—opening up a new frontier for sound machines like the C-64.

## Advanced Music Making

Two software series, one from Sequential and the other from Waveform, take the musical process a couple of extra steps. Both provide a means of playing and recording (or sequencing) an original composition, and turning it into a real musical score which can be edited and even

Figure 1.

SID's Tricks  
Performed by these sound synthesizer programs:

	Kawasaki Composer	Sound Maker	Melodian ConcertMaster	MusiCalc 1	3001: A Sound Odyssey
3 Voices	X	X	X	X	X
Separate Voice Adjust	X	X (in Song Builder)	X	X	X
4 Waveforms	X	X	X	X	X
ADSR Envelope	X	X	X	X	X
Polyphonic Play	X	X	X	X	X (complicated)
Variable Pulse Width	X	X	X	X	X
3 Filters	X	X	X	X	X
Variable Cutoff Frequency	X	X	X	X	X
Resonance	X	X	X	X	X
Ring Modulation	X	X	X	X	X
Synchronization	X		X	X	X
Oscillator Modulation				X	X
Envelope Modulation				X	X
Frequency Offset					X
Pitch Offset				X	X
Preset Sounds	X	X	X	X	X
Special Effects	X				

## The Magic of MIDI

If you want to go beyond the restraints of the packages that we have reviewed here, and even of the Commodore 64 itself, there is an amazingly helpful system for doing so. It's called MIDI, for Musical Instrument Digital Interface. This system is an evolving hardware/software specification for digital data communication between musical instruments, recording and effects devices, and computers. It is, in part, a musical RS-232 interface. Built into many synthesizers, other electronic musical instruments, some separate interface units now available, and even some computers, this system allows one instrument to control another, or to use a computer as a controller to build a more complex sound-producing system. No MIDI device has yet surfaced specifically for the C-64, but some may be on the way. One possible application for it is to use MIDI's 16 "tracks" to turn the computer into a 16-track digital recorder. Look for more information on MIDI in future issues of *Home Computer Magazine*.

printed out on paper. *MusiCalc*'s *ScoreWriter* takes a sequence file from *MusiCalc I* and prints it out in musical notation. In the Sequential series, sounds created on *Sound Maker* can be used to record a song in *Song Builder*. Then *Song Editor* will transform this recording into an editable screen display of notes on a staff, and *Song Printer* will turn it into hard copy.

*MusiCalc* also has a unique program in its series called *Keyboard Maker*, which allows you to build customized musical scales. *MusiCalc I* always loads a particular scale into the keyboard to match the key in which a song is written. A long selection of musical scales from all over the world—some familiar, some exotic—come on the program disk, but with *Keyboard Maker*, you can supply your own. This is one way of transcending the keyboard, whose customary use as the main means of playing the synthesizer is, in the final analysis, merely a historical accident.

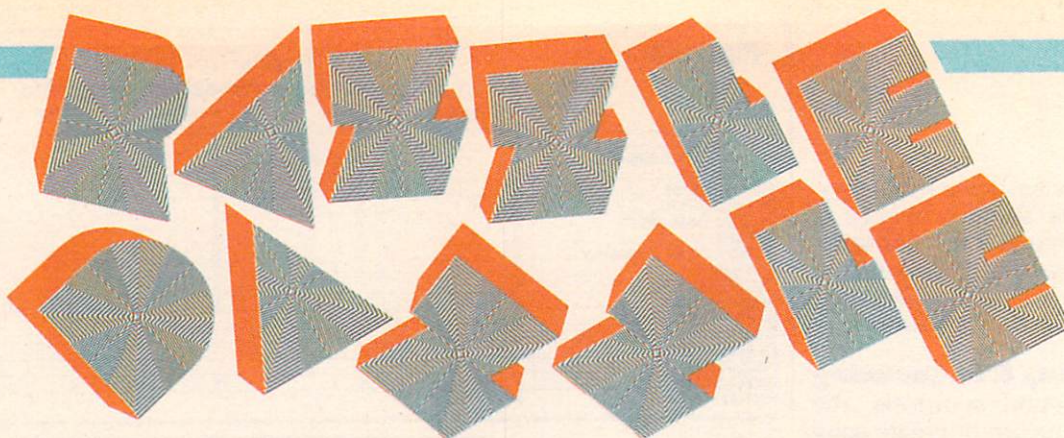
## A Final Word

These musical software packages for the C-64 are primarily creative and educational *tools*. They represent a new era of promise in home computing—and they truly serve both the beginner looking for inspiration and the advanced sound and music enthusiast looking for a relatively inexpensive way to create original works. They also illuminate what may be the C-64's strongest attribute: its amazing sound chip. Now that others in the computer world can see SID in all its home-oriented glory, they will—if we are to believe what we hear from the industry—soon unleash a new generation of incredible music machines.

HCM

[Be sure to follow our "Commodore Hornblower" series, beginning in this issue, as we build a BASIC program to turn your C-64 into a music synthesizer. Also, in the next issue we will compare some of the programs reviewed above to the top-of-the-line Casiotone CT-6000 keyboard—Ed.]





by William K. Balthrop

HCM Staff

## A few quick tricks tune your TI-99/4A into dazzling 3-part harmony.

How much money do you suppose your neighbors paid for that big fancy organ in their living room? I'm willing to bet it was more than you paid for your TI-99/4A. How would you like to aggravate them by turning your computer into an organ just like theirs, at a fraction of the cost? Read on and find out.

The TI-99/4A has a very powerful sound synthesizer built into it. It is capable of producing up to three different tones and one noise at the same time. In addition, you can independently control the volume for any of the three tones or the noise.

This program, *Muskey*, converts your keyboard into a musical organ; each key in your keyboard will be programmed to create a tone. By pressing the keys, you can play your favorite tune, or just have a lot of fun.

### Split Keyboard

The 99/4A has what is known as a split keyboard. This doesn't mean that it's broken. It allows the computer to seemingly read two keys at the same time. When accessing the split keyboard, the parameters passed in the **CALL KEY** statement are slightly different. When the first parameter in the statement is a 0, both halves of the keyboard are read, and normal ASCII values for the key

pressed are returned. If you use a 1 as the first parameter—as in the **CALL KEY(1,K,S)** statement—then only the left side of the keyboard will be read, returning a value from 0 to 19. These are *not* ASCII values. Using a 2 as the first parameter means that the right side of the keyboard will be read, with a value from 0 to 19 again being returned.

If you use two **CALL KEY** statements in a row, reading first one side and then the other, you can detect which two keys are being pressed on either side of the keyboard at practically the same time. This is done in lines 340 and 350 of the program. The left side of the keyboard controls Voice 1. The right side of the keyboard controls Voice 2. Voice 3 is programmable. I will discuss this in more detail later.

This method allows you to press two keys at the same time, thus mixing two different tones together. You can turn these voices off or on at any time. Pressing 8 will turn Voice 1 off or on. Pressing 9 will turn Voice 2 off and on. Pressing 0 will select one of three preprogrammed background rhythms.

### The Beat Goes On

The third voice or tone is preset to generate an accompaniment. You can use one of the three rhythms provided here, or generate your own. The rhythms can be changed by pressing 0. With each keypress, a different rhythm is selected. Currently only three rhythms are set up in the program. If you have a working knowledge of TI BASIC, then you can increase this number or modify the existing three rhythms to suit your needs.

Lines 590 through 610 contain the data for the three rhythms. If you want to change a rhythm, you can do so by altering the data in these lines. Each rhythm is on its own line. The first number in the **DATA** statement represents a frequency, the second number a volume. There are eight combinations of a frequency with a volume on the line. These are the notes which will comprise the

VOICE #1					VOICE #2					
1	2	3	4	5	6	7	8	9	0	=
Q	W	E	R	T	Y	U	I	O	P	/
880	784	698	659	587	880	784	698	659	587	220
A	S	D	F	G	H	J	K	L	;	ENTER
523	494	440	392	349	523	494	440	392	349	
SHIFT	Z	X	C	V	B	N	M	,	.	SHIFT
	330	294	262	247	220	330	294	262	247	

\* Turn voice 1 off and on  
 \*\* Turn voice 2 off and on  
 \*\*\* Select background beat

Key ——— M D ——— Note  
 294 ——— Frequency





rhythm. If you change the data in these lines, you will need to keep the same number of items on each line—unless you wish to change other parts of the program. A frequency can be any number from 110 to 44733, or a negative number from -1 to -8. The positive values are in Hertz or cycles per second. The negative values produce 8 different types of noise. The volume can be any number from 0 to 30. A value of 30 turns off the sound. The notes you set up in these lines will repeat themselves over and over again while you play notes on the rest of the keyboard.

If you want to make the background tune longer, you can do so by making minor changes to the program. In line 220 you will need to change the center subscript of the BN( ) array to the number of notes that you want to use. It is currently set to 8. You will then need to change the 8 in the FOR-NEXT loop in line 250 to the number of notes used. Line 540 checks for the note that is to be played and restarts the sequence when it reaches the end. You will also need to change the value that Z2 is tested against to one more than the number of notes you are using. Finally, you will need to provide enough data in lines 590-610 to supply the array with notes and volume settings.

***“The TI-99/4A has a very powerful sound synthesizer . . . capable of producing up to three different tones and one noise at the same time.”***

If you want to add more choices to the background rhythms, you can make a few more minor changes. To start, you will need to increase the size of the first subscript of the BN( ) array in line 220, which is the rhythm index. It is currently set to 3 because there are 3 rhythms. Increase the size to the number of different rhythms you would like to use. Next, change the 3 in line 240 to the number of choices that you are going to have. Each time you select another rhythm, Line 500 increments Z1. Line 510 then checks to see if the rhythm index is beyond the number of rhythms available. If so, line 520 resets the rhythm index to 1. You need to change the value against which Z1 is checked. The value needs to be one more than the number of rhythms being used. Currently the value is set to 4, because there are 3 rhythms.

You can experiment with other parts of the program. In line 560, the tones are actually played. The volume is set to either 0 or 30, depending on the status of the flags in V1 and V2. If you would like to simply have a lower volume instead of turning off the voice, you can lower the number 30 to a value from 0 to 30.

In line 580 there are a number of preset values for the program when it starts. The first value is the duration. It is currently set to -600. The minus sign tells the computer to discontinue the tones if another CALL SOUND is encountered. Try changing this to 600 and see what happens. What happens when you try some of these values in place of the -600 duration: -50, -200, -300, 50, 250, 400, 1000?

HCM

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

Line Nos.	Explanation
100-190	Program header.
200-330	Initialize program variables.
340-350	Scan both halves of the keyboard.
360-430	Assign frequency to voice 1 and 2 from N( ) array.
440-490	Turn Voice 1 and 2 off or on.
500-520	Choose which background rhythm is playing.
530-550	Choose which note to play in the background. Indexes into the BN( ) array.
560-570	Play the tone and return for more.
580	Data for the initial condition of the tones when the program starts. The first value is the duration, which will remain the same throughout the program.
590-610	Data for the background tune. One data statement exists for each beat.
620	Data which maps notes to the keyboard.

```

100 REM *****
110 REM * MUSIKEY *
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 5.2.1
180 REM TI BASIC
190 REM OR EXTENDED BASIC
200 CALL CLEAR
210 PRINT TAB(10); "MUSIKEY" : : : : :
220 DIM N(19), BN(3,8,2)
230 READ D,F1,V1,F2,V2
240 FOR Z1=1 TO 3
250 FOR Z2=1 TO 8
260 READ BN(Z1,Z2,1), BN(Z1,Z2,2)
270 NEXT Z2
280 NEXT Z1
290 Z1=1
300 Z2=1
310 FOR Z=0 TO 19
320 READ N(Z)
330 NEXT Z
340 CALL KEY(1,K1,S)
350 CALL KEY(2,K2,S)
360 IF (K1=-1)*(K2=-1) THEN 530
370 IF K1=-1 THEN 400
380 IF N(K1)<110 THEN 400
390 F1=N(K1)
400 IF K2=-1 THEN 530
410 IF N(K2)<110 THEN 440
420 F2=N(K2)
430 GOTO 530
440 IF N(K2)<3 THEN 530
450 ON N(K2)-2 GOTO 460,480,500
460 V1=ABS(V1-1)
470 GOTO 530
480 V2=ABS(V2-1)
490 GOTO 530
500 Z1=Z1+1
510 IF Z1<4 THEN 530
520 Z1=1
530 Z2=Z2+1
540 IF Z2<9 THEN 560
550 Z2=1
560 CALL SOUND(D,F1,V1*30,F2,V2*30,BN(Z1,Z2,1),BN(Z1,Z2,2))
570 GOTO 340
580 DATA -600,220,0,220,0,1250,20,-7,0
590 DATA 262,0,330,0,220,0,1250,20,-7,0
600 DATA 440,0,494,0,523,0,494,0,523,0,
610 DATA 220,0,220,15,247,0,220,15,330,
620 DATA 294,523,494,440,784,698,659,2,
3,4,5,587,392,247,262,330,220,349,8
80,1

```



# Lost in CP/M Land:

A Review of the Microsoft Premium Softcard IIe  
by Patricia Swift

*An impressive-looking package turns in  
a less-than-impressive performance. Is Microsoft's implementation  
merely a **Confusing Package for Microcomputers?***

**W**hen is an Apple not an Apple? When it's running CP/M? You might think so—but in the application of this CP/M package from Microsoft, we have found the above answer to be only partly true.

The *Microsoft Premium Softcard IIe* does allow you to run the popular CP/M operating system on the Apple IIe with at least one floppy-disk drive. The *Softcard* hardware is an add-on board which you install in the auxiliary slot of the Apple IIe. This board has a Z80 processor with its own 64K of Random Access Memory. It also doubles as an extended 80-column card when the CP/M system is not running. In contrast to previous CP/M add-ons from other companies—designed for the II+ and with fewer features—the Microsoft package is tailored specifically for the IIe. If you're content to run CP/M programs specifically formatted for the IIe, this product has much to offer.

The *Softcard* software includes Digital Research's CP/M operating system, some of the standard CP/M utility programs, some CP/M utilities by Microsoft which are unique to the Apple, and a Microsoft BASIC interpreter. The *Softcard* package also comes with a manual which covers installation and operation of CP/M and the BASIC interpreter. When purchasers return their registration card, Microsoft will send them an additional programmer's reference manual.

According to the *Softcard* manual, this is CP/M 2.0—which might be a problem because most of the programs out there run under CP/M 2.2. I asked the Microsoft support line about this and was told that the

manual probably meant CP/M 2.x, which would include 2.2. I later found out that this omission is quite significant, as I will explain later.

## Installation

Installing the *Softcard* itself in the Apple was easy, and clear instructions are given in the manual. The instructions also list the preferred slot positions of other boards you may be using, so that you can reposition the boards while you have the cover off.

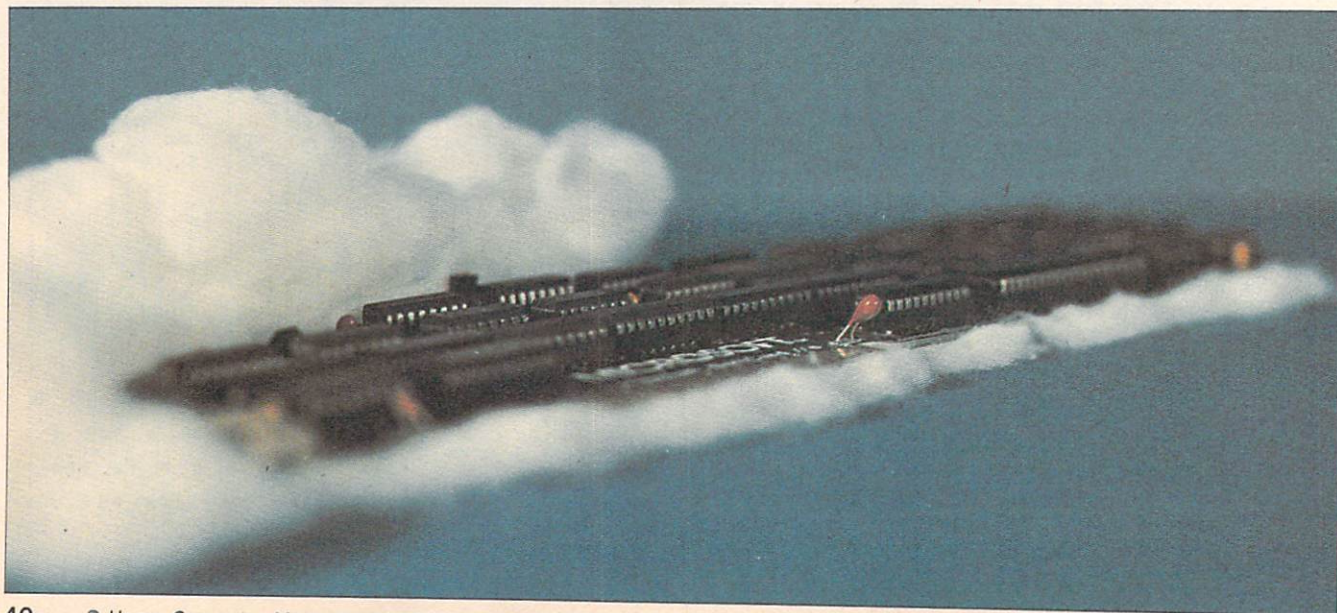
A separate section in the manual called "Getting Started" describes how to boot the system and make a backup copy of the master disk. Again, these instructions are easy to follow.

I used an Apple IIe with a monitor, two disk drives, a *Super Serial Card*, and the *Softcard* package for this review.

## CP/M Commands

The *Softcard* implements most of the commands and utilities you'd expect to see in CP/M. With the one exception discussed below, these commands are used in the usual way. For readers who are not familiar with CP/M, here's a list of the standard CP/M command files included with the package:

ASM	an assembler
DDT	a debugger
DIR	display the directory of a disk
DUMP	display a file in hexadecimal notation





ED a line editor  
 ERA erase file(s) from disk  
 LOAD turn assembler output into an executable program  
 PIP copy disk files  
 REN rename disk files  
 SAVE copy a portion of memory to a disk file  
 STAT display the status of the system, disks, and files  
 SUBMIT have CP/M execute a series of instructions  
 TYPE display a text file  
 USER select a user area  
 XSUB extended SUBMIT with more power

The STAT command is also used to associate logical and physical devices. The traditional CP/M commands FORMAT, SYSGEN, and MOVCPM are not present. MOVCPM is unnecessary on a 64K machine. FORMAT, which formats disks, and SYSGEN, which copies the CP/M bootstrap to disk, are replaced by the Microsoft utility program COPY.

### Microsoft Extras

Microsoft includes several additional command files with the Softcard package. These are summarized below:

APDOS convert files from Apple DOS to CP/M  
 AUTORUN lets you designate a command to be executed automatically whenever CP/M is booted  
 BOOT reboot the system (cold boot)  
 CAT list the directory of a disk in alphabetical order  
 COPY copy and/or format disks, with or without the bootstrap  
 GBASIC the Microsoft BASIC interpreter  
 MFT copy disks on a single-drive system  
 PATCH install program modifications

### Trying To Run Some CP/M Applications

The main reason for adding the Softcard to your Apple is to allow you to make use of thousands of existing CP/M programs that the Apple 6502 processor cannot run. This is not only my opinion—Microsoft says the same thing in its introduction in the Softcard manual.

I have a large library of CP/M programs which run on other computers. In evaluating the Softcard, I had planned to transfer a few of these programs to the Apple so that I could report on how they ran. But after hours of frustration trying to transfer CP/M programs from a Kaypro computer to the Apple, I am forced to admit defeat.

My first stumbling block was the disk format used by the Apple with the Softcard. Although I can produce more than 40 CP/M disk formats, I was not able to produce one that the Apple could read. Microsoft confirmed that the Softcard uses a disk format unique to the Apple/Softcard combination.

This is not good news for potential users of the Softcard. It means that you will have to order CP/M software in "Apple with Softcard" format. Several major software distributors supply some titles in this format, but you might have problems trying to order this format from smaller distributors and software manufacturers.

Name: Microsoft Premium Softcard IIe  
 Product Type: Add-on board with Z80 processor, 64K memory, and 80-column display hardware, plus CP/M operating system software and Microsoft BASIC interpreter  
 Machine: Apple IIe  
 Distributor: Microsoft Corporation  
 10700 Northrup Way  
 Bellevue, WA 98004  
 1-800-426-9400  
 Price: \$395  
 System Requirements: One disk drive required, second disk drive recommended.

	Poor	Fair	Good	Excellent
Ease of Set-up				
Ease of Use				
Documentation				

So my first strategy of transferring CP/M programs and copying them onto Apple disks was out. But there is another standard way to transfer files from one computer to another: the RS-232 interface. Because my partner and I have transferred files between computers many times, we were prepared for a few difficulties. The *Super Serial Card's* manual had sufficient information on its RS-232 interface for us to easily construct a cable, although we noticed that there were no "handshaking" lines, which proved to be a problem later. (Handshaking refers to the acknowledgements which pass back and forth between two computers.)

We hooked the computers together and were ready to send a file to the Apple. CP/M uses two utilities to

***"After hours of frustration, I am forced to admit defeat. I was not able to transfer any usable programs to the Apple."***

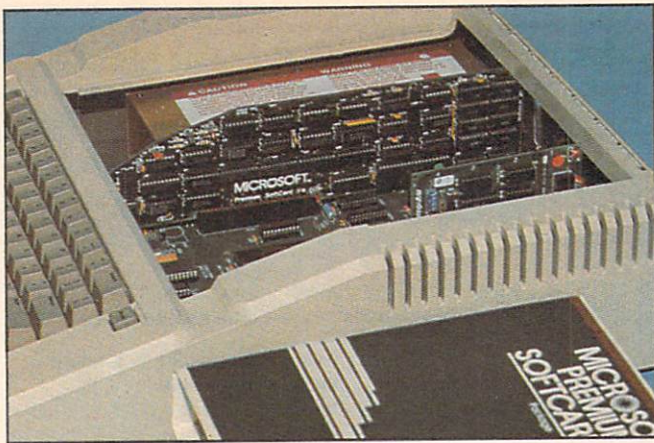
transfer CP/M files; PIP and STAT. PIP do most of the work, but STAT must be used first to associate a logical name with the physical RS-232 device. On every other CP/M computer of my experience, the physical device name for the RS-232 interface is TTY; but on the Apple it is PTR; I had to call the Microsoft support line for this insight, and also for the information that the *Super Serial Card* should be installed in slot 2, as this was not clear in the documentation. I also found out that the version of CP/M which had been shipped with the Softcard (V2.25) didn't work with the *Super Serial Card*.

Several days elapsed while we waited for version 2.26 to arrive from Microsoft. Then we had our first success: we were able to load three short CP/M files to the Apple by running PIP on the Apple. The files we loaded were from a favorite communications program which we had hoped to use for the rest of the file transfers. Alas, the Softcard uses the 6502's Basic Input Output System (BIOS) for all Input and Output (I/O), and our standard CP/M communications program is not equipped to issue 6502 BIOS calls.

### Ups and Downs

Next we decided to transfer a word processor utility to the Apple using the PIP technique that worked for the other files. The technique is laborious because executable programs have to be processed with the Load utility after being transferred to the Apple—but at least it appeared to work. We soon found out that our longer





files were not being transferred properly, due to the lack of "handshaking" which I mentioned before. With our simple transfer, the Kaypro was sending out each file in a burst. This worked fine as long as the Apple could also receive it in a burst—that is, hold the file in its memory. But for longer length files, the Apple had to pause to write part of the file to disk when its memory got full. Since there was no handshaking to tell the Kaypro to stop transmitting while the Apple was writing to disk, the Apple missed a section of the file for each disk-write. This problem effectively ruled out our PIP transfer strategy because files as short as 16K were being received incorrectly.

Now we had exhausted our "usual" approaches to transferring programs between CP/M computers. Undaunted, I turned next to the *Softcard Programmer's Manual* to follow Microsoft's procedure for transferring files. My thought after a first reading was: Good luck if you don't know CP/M and 8080 assembly language! I do, and I still had problems. The manual provides listings of two programs: *Download*, which runs on the Apple, and *Upload*, which runs on the other computer (Kaypro in my case). On a previous call to Microsoft's support line, I had been told that the *Download* listing was not correct and I received the correct listing in the mail a few days later. I keyed the new *Download*

listing into the Apple, assembled it, and loaded it with no problems. In the process I came to understand that Microsoft's strategy involves sending and receiving a file block-by-block—a very simple process using software handshaking, so I figured that the lack of hardware handshaking should cause no problems.

With the Apple half of the link squared away, I turned to the Kaypro half. The *Upload* program in the manual isn't all there; you have to write your own routines to initialize the serial port, input a byte from the port, and output a byte to the port. Microsoft provides only a verbal description of the routines, and no examples. And you can't refer to the *Download* program for hints, because *Download* uses 6502 BIOS calls while you'll be using the 8080 opcodes IN and OUT. Because I knew my Kaypro, I managed to write the missing sections and assemble the program.

After all this, I was actually able to transfer some files using *Upload* and *Download*. The transferred files were even the correct lengths, showing that no sections of the files had been lost. My elation soon turned to dismay, however, when I tried to run the transferred programs. INVALID OPCODE was the typical response from the Ap-

ple. When I examined the transferred programs, I saw that the eighth bit had been set to 0 in every byte—in effect, the Apple was receiving only 7 bits of each byte. This would be fine for text files which use only ASCII codes, but this doesn't cut it for program files.

The Microsoft support line insisted that CP/M must be responsible for stripping off the eighth bit, even after I explained that I had transferred 8-bit bytes using PIP and the same cabling setup. I suggested that the sections of code I wrote for *Download* might be at fault, but I was unable to obtain any help at all with those routines. The Microsoft support person said he was unable to provide details because he was not familiar with the Kaypro, which is understandable. But he wouldn't (or couldn't) give even general information on the structure of these routines. At this point I finally gave up trying to transfer programs to the Apple.

### The Written Word

My confusion over the correct slot for the *Super Serial Card* was caused by the manuals. In different places I was advised that the card should be in slots 1, 2 or 3 of the Apple. And then there was the listing of the *Download* program, which was incorrect in the manual.

The Microsoft manuals look nice, are well-

organized and contain a

substantial amount of information, but none of this matters if correct factual content is missing. I spent too much time in confusion to dismiss these things lightly.

### In Summary

In the process of trying to transfer programs, I had the opportunity to run most of the standard CP/M utilities that come with the *Softcard*. The editor, assembler, and debugger ran exactly as expected, and except for the device names and port handling, the *Softcard* provides a solid CP/M operating system. Experienced CP/M users will have to learn a few new commands, but that's easy enough to do.

I can't say whether off-the-shelf CP/M 2.2 programs will run on the Apple with *Softcard*. The experiences I outlined above should make potential users aware of two things: First, be sure that the CP/M software you plan to run is available in "Apple with *Softcard*" disk format—unless you're a communications wizard who knows CP/M. Second, don't count on much help from Microsoft's support line for this product.

---

*"Although I can produce over  
40 CP/M disk formats,  
I was not able to produce one  
that the Apple could read."*

---



# What is CP/M?

CP/M (Control Program for Microcomputers) was first developed for the Intel 8080 microprocessor by Gary Kildall, founder of Digital Research Inc. Since that time, CP/M versions have been created that will work with microcomputers based, not only on the Intel 8080 microprocessor, but also on the Zilog Z80 and the Intel 8085 microprocessors. All three are eight-bit microprocessors, and the CP/M versions that work with these integrated circuits are generically known as CP/M-80.

CP/M itself is a program commonly referred to as an operating system. Operating system software, in general, is designed to act as both isolation and interface between computer hardware

and applications software (such as the BASIC and Pascal languages, business software, or even video games). Operating system software performs all of the work in accessing or cataloging the files on a diskette, routing information from the memory to the screen, controlling the system printer interface, and other mundane tasks.

One of the biggest advantages of CP/M-80 is that it provides a common environment for application programs across several different manufacturers' computers. The differences between CP/M-80 machine environments primarily occur in two areas—the disk format, and the kernel of routines known as the Basic Input/Output System (BIOS).

## IBM and CP/M

Digital Research has also developed a CP/M version that works with the Intel 8086 and 8088 16-bit microprocessors. This version of CP/M is known as CP/M-86. This operating system works on 16-bit computers such as the IBM PC Family, Texas Instruments Professional Computer, and several other PC-type machines. Software cannot be directly transferred from machines using CP/M-80. In many cases, however, a program's "source code" can be transferred from a CP/M-80 machine into a CP/M-86 machine and recompiled.

In the IBM PC/compatibles world, Microsoft's MS-DOS (IBM PC-DOS) is by far the dominant operating system. CP/M-86 finds favor largely with former CP/M-80 users who have gone to a PC-machine environment. **HCM**

## CP/M in the Commodore World

At one time, Commodore sold a plug-in module for the C-64 that supported CP/M-80. For reliability and performance reasons that module is no longer available. To our knowledge, no viable CP/M conversion exists for the C-64. But—CP/M fans, take heart!

The new Commodore 128 announced at the Consumer Electronics Show in Las Vegas this past January boasts an 80-column-display CP/M mode. This machine is due to be released in the second quarter of 1985, and the cost of the C-128 console and companion disk drive is projected to have a \$500 suggested retail price.

In order to run the new C-128 in CP/M mode, the also new "smart" Commodore 1571 disk drive is required. This drive will store 410K-bytes in a format that is compatible with the Kaypro and Osborne CP/M computers. The Commodore 128 will allow CP/M to access all available memory in the machine (up to its maximum of 512K).

Commodore is presently de-emphasizing the fact that the C-128 runs CP/M because of the reported high cost of CP/M applications programs compared to the low cost of the Commodore 128 system. This is understandable when considering consumers who are looking for a low-cost computer system. But for those consumers who are looking for a CP/M applications system with a low total cost—i.e., hardware plus particular applications software unique to the CP/M world—the C-128 may be a good machine to consider. If the

marketplace consumes a large quantity of C-128's, the cost of CP/M software will most likely take a tumble

as smart software houses attempt to capitalize on the newly created market for their wares. **HCM**

The New Commodore 128 Home Computer



### Specifications

<b>Size:</b>	The light beige Commodore 128 is 12.96 inches deep, 17.28 inches wide, and 2.24 inches high. The 1571 disk drive is 13.84 inches deep, 8.64 inches wide, and 3.04 high.
<b>Weight:</b>	The 128 weighs 5 pounds, 6 ounces. The disk drive weighs 6 pounds, 12 ounces.
<b>Power Source:</b>	Both the drive and the computer require 117 Volts, 60 Hz, 15 Watts.
<b>Sound:</b>	The Commodore 128 includes a 6581 Sound Interface Chip.
<b>Keyboard:</b>	A 92-key standard typewriter style is used, with a 14-key numeric keypad, 8 programmable function keys, and 6 cursor keys.
<b>Video:</b>	In 64 mode: 40 x 25 lines (320 x 200 resolution), 16 colors and 8 sprites. In 128 mode and CP/M mode: 40 x 25 lines (320 x 200), 80 x 25 lines (640 x 200 resolution), 16 colors and 8 sprites in 40 column 128 mode only.
<b>Memory:</b>	In 64 mode: 64K RAM and 16K ROM. In 128 mode: 128K RAM expandable to 512K, 48K ROM, and 16K ROM for DOS support. In CP/M mode: 128K RAM expandable to 512K. The 1571 drive has 2K RAM and 32K ROM.
<b>Processor:</b>	The 64 mode has a 6510 compatible microprocessor. The 128 mode has an 8502 microprocessor and is 6502-compatible. CP/M mode has a Z80A microprocessor. The 1571 drive, which is required for CP/M mode, has a 6502 microprocessor.
<b>Disk Drive:</b>	The 1571 drive's data transfer rates are 300 cps under 64 mode, 1500 cps under 128 mode, and 3500 cps under CP/M mode.
<b>Operating System:</b>	Built-in DOS support is provided with the 128, as is BASIC 7.0 and CP/M Plus Version 3.0. The drive also has built-in DOS.
<b>Input/Output Ports:</b>	The 128 has a user port, cassette port, RF/TV port, audio input port, composite video port, serial port, 2 game ports, cartridge port, audio output port, and digital RGBI video port. The drive has 2 serial ports for daisy-chaining additional peripherals.



# A CP/M Dawn

## for the TI-99/4A



A Review of Morning Star's CP/M Package  
by Patricia Swift

*Thousands of programs await the TI-99/4A—if  
only it ran Adam Osborne's brand of CP/M. With this new package,  
it can do just that and more . . .*

**A**re you looking for a way to expand the range of software available for your TI-99/4A? How about the wealth of programs that run on the "transcomputer" CP/M operating system? Ordinarily you would not be able to run these programs on the TI-99/4A because the TI machine uses the TMS 9900 microprocessor, which is not compatible with CP/M. But Morning Star Software's CP/M product contains the hardware and software necessary to run this popular system on the TI machine, finally opening up the wide, business-oriented world of CP/M to TI-99/4A users.

Morning Star's package consists of hardware, software and documentation. The hardware is a sturdy card which plugs into the Peripheral Expansion Box. This card contains a Z80 processor; 64K of random access memory (RAM); and an erasable, programmable, read-only memory (EPROM). The CP/M software comes on one 5-1/4 inch floppy disk. This disk contains most of the standard CP/M programs plus a few extras. The documentation consists of two manuals: Digital Research's general CP/M manual, plus a smaller manual from Morning Star describing CP/M on the TI-99/4A.

For this review I used a 99/4A with a TI color monitor, the peripheral expansion box, one disk drive, the memory expansion, the RS-232 interface card, an Okidata 82A printer, and the Morning Star CP/M package.

Installing CP/M on the 99/4A is a breeze. Complete installation instructions are given in the manual. After installation, you turn on the 99/4A as usual. You will see that a small menu has been added to the TI "color bars" screen, simply telling you to press 1 for the TI disk system or 2 for CP/M. Next you mount the Morning Star CP/M disk in the main (or only) disk drive, and press 2 for CP/M.

### The Screen

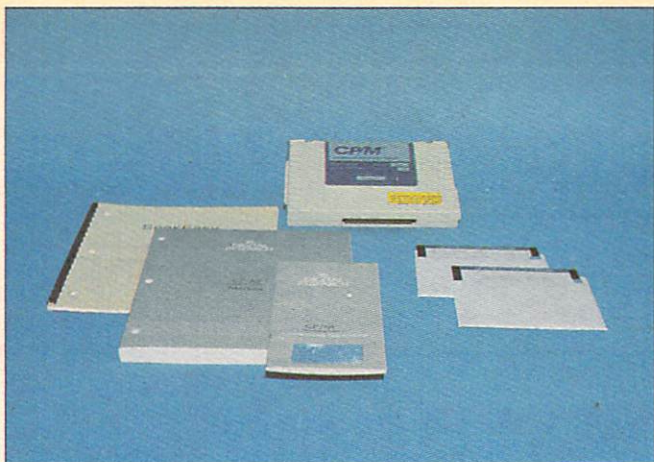
Morning Star CP/M uses the screen in Color Graphics mode, which gives you 40 columns by 24 lines on the screen at any one time. This 40-column limitation is built into the TI hardware. Most of the CP/M software I've seen uses 80 columns by 24 lines. To get around this potential conflict, Morning Star thinks of the TI screen as a "window" on a larger "writing pad." This writing pad is 80 columns wide by 72 lines long. You can move the screen window around to view different portions of the writing pad by means of special key combinations. For example, (FCTN) 4 rolls the screen down 24 lines to show the next 24 lines of the writing pad. Nine different combinations exist for moving the window, and I found myself referring to the manual for all except one: (FCTN) (SPACE) gives you the other half of the screen (that is, the 40 columns to the left or right of where you are). This combination was the only one I really needed.

By using color graphics mode, Morning Star squeezes the most out of the existing TI-99/4A hardware (remember, TI BASIC gives you 32 columns, and only 28 using the PRINT statement), but there's a price: the screen is slow to draw and scroll. The company plans to offer the option of using 40-column Text mode in the near future, and this should speed things up for text applications like word processing, where you don't need color. To someone who has used CP/M on an 80-column screen, it is difficult to adjust to seeing only 40 columns at a time. If you are primarily a TI-99/4A user, then you are probably accustomed to 32 or 40 columns, so this limitation won't bother you.

### The Tutorials

The Morning Star manual includes several tutorials which cover the basics. They are simple and clear. The





most important points—such as backing up your CP/M disk—are covered, and there is even an explanation on how to copy a file from one disk to another on a single-disk system. This useful procedure might take you months to figure out on your own. It's just the kind of thing you always hope to find in a tutorial but often don't.

### CP/M Commands

Most of the "traditional" CP/M commands are present in the Morning Star system, and there is nothing weird about their usage. If you're already familiar with CP/M, then you should be able to step right up and start working. For readers who aren't familiar with CP/M, these commands and programs are present:

ASM	—an assembler
DDT	—a debugger
DIR	—display the directory of a disk
DUMP	—display a file in hexadecimal notation
ED	—a line editor
ERA	—erase a file from disk
LOAD	—turn assembler output into an executable program
PIP	—copy disk files
REN	—rename disk files
SAVE	—copy a portion of memory to a disk file
STAT	—display the status of the system, disks, and files
SUBMIT	—have CP/M execute a series of instructions
TYPE	—display a text file
USER	—select a user area
XSUB	—extended SUBMIT with more power

You can do quite a lot with these utilities. For example, you can input an 8080 assembly language program (with ED), assemble it into machine code (with ASM), debug it (with DDT), and store it on disk in executable form (with SAVE).

Experienced CP/Mer's might notice that three utilities are missing, namely MOVCPM, SYSGEN, and FORMAT. The MOVCPM utility, which lets you reconfigure CP/M for different memory sizes, would serve no purpose because the Morning Star system already has 64K of memory. SYSGEN, which copies the CP/M operating system bootstrap to disk, and FORMAT, which formats disks for CP/M, are replaced by the two special utilities described in the next section.

### Morning Star Extras

Morning Star provides two nontraditional utilities with its CP/M. The first is called INIT. It formats disks for CP/M and puts a copy of the system bootstrap onto



Name: Morning Star Software  
 Program Type: CP/M  
 Machine: Operating System  
 Distributor: TI-99/4A  
 Morning Star Software  
 4325 S.W. 109th Ave.  
 Beaverton, OR 97005  
 1-800-824-2412  
 Price: \$595  
 System Requirements: Peripheral expansion box, disk drive. Optional: Expansion memory, 2 additional disk drives and RS-232 interface.

	Poor	Fair	Good	Excellent
Performance:	████████████████████			
Ease of Use:	████████████████			
Ease of Set-up:	████████████████████			
Documentation:	████████████████			

*"It's obvious  
 that Morning Star Software designed  
 their product with the goal  
 of allowing TI-99/4A users access to most  
 of the CP/M programs available."*

the first three tracks of the disk. INIT is a nice addition to CP/M because it's a one-step process that you will use a lot.

If you're wondering what a bootstrap is, let me explain. The bootstrap is the set of instructions that brings CP/M to life. When you turn on the 99/4A and select CP/M, you must have a disk with the bootstrap mounted in the main disk drive or the system won't know what to do. The bootstrap may also be needed at other times while you're running CP/M. For this reason you'll want to have bootstraps on almost all of the disks that you work with in the main disk drive. Morning Star makes it easy by including the copy procedure of the bootstrap in its INIT and BACKUP programs.

BACKUP is the other extra Morning Star utility. It makes copies of whole disks. It is like INIT in that it formats the new disk and copies the bootstrap. It is especially useful in making backup copies on one-disk systems.

Morning Star CP/M adds another useful feature to the 99/4A: a keyboard buffer. This means that you can type faster than the letters appear on the screen. Once you get used to this you'll have a hard time using TIBASIC, where your keystrokes are lost if you type too fast.

### Running Some CP/M Applications

As I said earlier, the real reason to use CP/M is the wealth of applications programs which run under it. So it seemed reasonable to run some of my favorite CP/M programs on the Morning Star system as a test of the system's capabilities.

You'll notice that many ads for CP/M software include a list of available disk formats. Although CP/M is known as a standard, the truth is that there are variations from computer to computer. One of these variations is the disk format. You usually cannot take a disk from one CP/M computer and read it on a different brand of CP/M computer. I was relieved to find that Morning Star CP/M uses the common Osborne-1 single-sided, single-density disk format. In other words, you can put an Osborne-1 disk into your 99/4A disk drive and have Morning Star CP/M read it. This means that you can order software in a readily-available disk format. One of my other CP/M computers writes Osborne-1 disks, so I was able to get



programs across to the TI machine very easily. I created Osborne disks from scratch on the other computer and used them on the 99-4/A. I also took CP/M disks from the 99/4A and used them in the other computer.

Another variation among CP/M programs involves the type of terminal or screen being used. Screen manufacturers use different codes to manipulate the information on the screen—for example to clear the screen. Many CP/M applications include a procedure for setting up the appropriate screen codes for your computer (be careful though, many programs do not).

### T/Maker II

The first program I ran on the 99/4A was *T/Maker II* by Peter Roizen. *T/Maker* was one of the first "integrated" programs—combining word processing, calculations, and primitive graphing. Running *T/Maker* itself went normally except for the smaller screen. I did regret that Morning Star replaces the normal (FCTN) S and (FCTN) D for cursor movement with (CTRL) S and (CTRL) D. This was a little hard to grow accustomed to, but once I adjusted to the new keystrokes I was able to whip out sensible text at close to my usual speed.

When it came time to print out my document, I had to do some experimenting. Morning Star will have a *SETUP* program to let you configure CP/M for various printer requirements such as baud rate, but it was not being supplied yet when *Home Computer Magazine* received the system for review. I was able to print on the Okidata printer without too much trouble by setting up the printer to conform with the RS-232 interface's default serial settings: 300 baud with parity and 7 bits of data. On the Okidata 82A, these settings are made with pencil switches. On the front bank of switches, I set switches 5 and 8 to on and the rest to off. On the rear bank of switches, I set switches 4 and 5 to off and the rest to on. Printing at 300 baud was not speedy, but it was adequate for testing purposes.

### Fortran-80 And Link-80

The next test involved running *Fortran-80*, a CP/M Fortran compiler, and *Link-80*, its linker, both by Microsoft. This was easier than running *T/Maker*, because these programs do no fancy screen manipulations. I used CP/M's editor *ED* to write a simple program and a couple of very simple subroutines on the 99/4A. Then I compiled them with *Fortran-80* and linked them with *Link-80*. Both the compiler and linker ran normally, and the resulting program ran just as expected.

There was one nice surprise, though: the disk speed was faster than usual. Morning Star uses a disk-buffering technique which speeds up reading from and writing to a disk.

### The Third Test

For my final test, I decided to run one of my own programs, which does maintenance on disk files containing business statistics. This program displays a main menu and several submenus. The user selects items from the menus and ultimately arrives at screens showing information about such things as year-to-date sales and profits. The user can then change the numbers selectively, and the changed information is stored on disk.

To prepare for the test, I copied the program and its associated data files onto a Morning Star (Osborne) disk. I mounted the disk in the 99/4A and invoked the program. It ran flawlessly.

This test pointed up another feature of Morning Star CP/M. My program is on the large side; in fact, it is too large to run on some CP/M computers. Morning Star gives you a 60K TPA (Transient Program Area), which means that you should have no memory-size problem in running off-the-shelf programs.

---

*"There was one nice surprise . . .  
the disk speed was faster than  
usual."*

---

### Things That Could Be Better

Aside from the small screen size, I found very few things about Morning Star's CP/M that I didn't like. The complexities of moving the window might be simplified by some kind of crib-strip for the TI-99/4A console, although the (FCTN) (SHIFT) combinations would be hard to show that way.

The *Digital Research CP/M Manual* is not as clear as it might be. I've used this standard manual many times before and always found it to be somewhat limited and confusing. Plenty of better books about CP/M are available in bookstores. I have at least two such books, and I think you might want to invest in one, especially if you are new to CP/M.

The Morning Star manual says that you can exit from CP/M and get back to the "color bars" screen by pressing (CTRL) (FCTN) (SPACE BAR). I found that this produced a garbled screen. I was able to reselect CP/M with no problems, but if I tried to select the TI disk system from the garbled screen, I continued to see garbage. The simple remedy is to

turn the console off and back on (after removing your disk).

### Conclusion

It's obvious that Morning Star Software designed their product with the goal of allowing TI-99/4A users access to most of the CP/M programs available. Its choice of a common disk format and screen codes, and the generous size of the TPA illustrate this intention. Within the limitations of the TI-99/4A hardware, I would say that Morning Star was very successful.

HCM



**MOVING?**  
**Don't Miss Out On**  
**Any Issues Of**  
**HOME COMPUTER**  
magazine

Send us a Change-of-Address Card  
(available at any Post Office)  
6-8 weeks prior to the move.

Be sure to include both the old & new address, plus the alphanumeric code above your name on the mailing label.

Please send this  
information to:

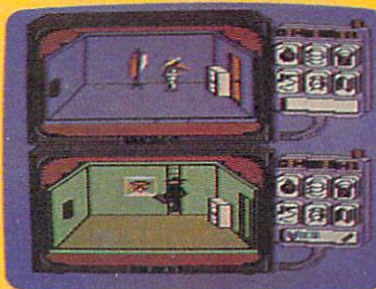
**Home Computer Magazine**  
P.O. Box 70288  
Eugene, OR 97401



# Spy vs Spy

A review by Steve Nelson  
HCM Staff

## HCM Review



Name: Spy vs Spy  
Program Type: Arcade game  
Machines: C-64, Apple II Family  
Distributor: First Star Software, Inc.  
18 East 41st St.  
New York, N.Y. 10017  
Price: \$34.95 Apple version  
\$29.95 C-64 version  
System Requirements: Apple: 48K, disk drive  
C-64: joystick, disk drive  
Performance: 

Poor	Fair	Good	Excellent
[Bar chart showing performance level]			

  
Engrossment: 

[Bar chart showing engrossment level]			
---------------------------------------	--	--	--

  
Documentation: 

[Bar chart showing documentation level]			
---	--	--	--

**Bombs, booby traps, and black spies reveal that  
there's nothing funny about counterespionage.**

As one who grew up with *MAD* magazine's ironic humor and satiric, madcap takeoffs on all sorts of things, it was with a sense of anticipation that I loaded its "official" *Spy vs Spy* game. As I played the game however, I got the feeling that something was missing. I found myself waiting for something funny to happen, like a booby-trapped booby trap—one that explodes in your spy's face as he is planting it. I began to ask myself "what's wrong here?" Then I realized, "this isn't very funny"—it's just another shoot-em-up, in spy drag. None of the humor of the comic strip ever shines through.

The object of *Spy vs Spy* is to move your spy through several rooms in an embassy while searching for your briefcase, a passport, traveling money, a key, and some secret plans. Once you find, fill, and keep the briefcase, you must locate the only exit and carry the briefcase to a waiting airplane in which you can make your escape.

In its split-screen, two-player format, *Spy vs Spy* introduces two new concepts: Simulvision and Simulplay, allowing you to see what the computer (or player 2) is doing while you are playing the game. That's right folks, you don't have to take turns with this game. The screen is divided in two, each half displaying various rooms through which the spies madly search for the hidden objects. The objects may be found under furniture, inside drawers, even behind pictures on the wall, but be careful—you may end up finding a booby trap.

### Boobytraps

To the right of each player's screen is the "trapulator." This is a machine that keeps track of the number of hidden objects that you have found, and the amount of time you have left. It is also used to choose and set booby traps. There are only 5 different booby traps to choose from: a bomb, a bucket of water used to electrocute the enemy spy, a gun with a string, a time bomb, and a spring—all of which can kill your spy quite effectively. One other item on the trapulator is the embassy map. You can access this map at any time to check your location in the building, and to see in which rooms objects are hidden—but there is a point penalty each time you use it.

*Spy vs Spy* is available for the Apple II Family, and also for the Commodore 64. Graphically, the two versions are nearly identical, with the C-64 version

generating a much sharper image on the monitor. The main difference between the two lies in their sound effects: the Apples' sound effects are minimal, almost nonexistent, while the C-64 version has a variety of different sounds. In order to move about, a joystick is required for the C-64, and is recommended for the Apple IIe and IIc. Movement is easy, and the spies are very quick to respond. In the Single Player mode, you can set the intelligence quotient of the computer. I found level 1 to be challenging enough, and level 5 (the highest) to be extremely difficult.

### Hand-to-Hand Combat

When a spy enters a room already occupied by the other spy, the screen of the room he left goes blank, and the two spies share the same screen until one is killed or escapes. When thrown together, the spies go into

Combat mode, where they attack each other with billy clubs. A few blows are sufficient to weaken the opposing spy, but you can kill him with about 7 to 10 solid blows. (Every time you kill the other spy, his screen goes blank and he loses 7

seconds of play time.) In hand-to-hand combat, position is the most important element. If you can't get close enough, the enemy spy will pummel you until you break and run, or die. Getting position on the spy when playing the computer is overly difficult—meaning you will lose a lot. This tends to get rather frustrating, to say the least.

### Something Missing

One can argue that, in a video adaptation of a magazine cartoon, the new medium itself demands action. So in *Spy vs Spy*, instead of humorously outwitting your opponent, you outrun him. Instead of carefully planting a diabolically clever booby trap, your spy races through the embassy scattering stock booby traps like mad hoping that the other spy, in his haste to discover the hidden objects, will just bumble into one.

Adapting *Spy vs Spy* from a very familiar comic strip to a computer game must have been a challenging task—and, as with all adaptations, you certainly can't please everybody. But in this instance, the most important part of the comic strip was left out, and without it, the game comes off hurried, and humorless.

HCM





## Sort Your ProDOS Catalogs

by Hendrik Broekhoff

*Tired of the disorganized order of your diskette files  
when you display the catalog?  
Here's an easy-to-use utility to "have it your way . . ."*

**H**ave you ever spent ten minutes looking through a giant directory on a disk, just trying to locate one file? Do you want to see your *AppleWorks* files in chronological order for a change? Well, here is the solution to your problems! The *Alpha Cat* program lets you re-arrange your files in any order you like, according to your needs. You can alphabetize the files on those giant directories, or put data base files in chronological order, making it easy to find any desired version. If you just want to move one or two files, *Alpha Cat* provides a means for doing this, too. This program never actually moves the files themselves, it simply re-arranges the order of the file information in the catalog. *Alpha Cat* is thus an invaluable aid in organizing your diskette volumes.

### Using The Program

When you first start up, *Alpha Cat* finds the current value of **HIMEM** and resets it so that the program will not write over the date buffer when it reads and/or writes to the disk. The title screen asks you to insert the ProDOS disk that you wish to work with in drive 1 and press (RETURN). *Alpha Cat* will take a few seconds to read in the file names. After all of the file names have been read in, the main menu will appear.

### The Main Menu

From the main menu you may choose the function that you want by scrolling to it through the given list with the arrow keys. To select a function, simply highlight it and press (RETURN). The functions are as follows:

- 1) SORT FILES
- 2) SEE CURRENT SETUP OF FILES (CATALOG)
- 3) MOVE FILE TO A NEW POSITION
- 4) WRITE MODIFIED CATALOG TO DISK
- 5) READ FILES FROM ANOTHER DISK
- 6) QUIT

Here's a description of each function:

1) **SORT FILES** - This option will organize the files in memory. After choosing this function, you will be asked one or two questions:

First you must decide how you want your files sorted. Choose **A** if you want them in alphabetical order, **B** for chronological. If you choose chronological order, then you must indicate which dates you want to use when sorting the files. Choose **A** for their modification dates (the ones you see in a normal, 40-column-display catalog), or **B** for their creation dates.

*Alpha Cat* then sorts the files in memory. In about seven seconds, 25 files can be sorted.

2) **SEE CURRENT SETUP OF FILES (CATALOG)** - This function lets you view the current order of the files. For convenience, the files are numbered. You refer to these numbers when using the Move File function described below.

3) **MOVE FILE TO A NEW POSITION** - To move a single file to a different position, you choose this function, and enter the file name when prompted. *Alpha Cat* will tell you the current position of the file, and ask you where you want to move it. You then enter a number for the new position. Use the Catalog function (above) to find the current numbers.

4) **WRITE MODIFIED CATALOG TO DISK** - None of the changes you make are permanent until you use this function. Before choosing it, make sure the drive contains the disk that was originally read in.

5) **READ FILES FROM ANOTHER DISK** - As its name suggests, this function allows you to work on another disk. It does this quite simply by restarting the program from the beginning.

6) **QUIT** - *Alpha Cat* resets **HIMEM** to its original value, catalogs the disk in drive one, and then exits to BASIC.

### Alpha Cat (Apple II Family) Explanation of the Program.

Line Nos.	
100-220	Program header and initialization.
230-480	Read in file names and main menu.
490-560	Catalog routine.
570-910	Routines to sort, move, and write file names.
920-940	Re-RUN program and Quit.
950-1010	Read/write-block routine.
1020-1070	File-purging routine.
1080-1100	Error-trapping routine.



```

100 REM *****
110 REM ** ALPHA CAT **
120 REM *****
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO
150 REM BY HENDRIK BROEKHOFF
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 5.2.1
180 REM APPLE II FAMILY APPLESOFT
190 REM PRODOS ONLY
200 HI = PEEK (115) + PEEK (116) * 25
6: HIMEM = 16384: ONERR GOTO 1080
210 TEXT: HOME: DIM FIS(51),EFS(51),O
PS(6):O = 1: GOSUB 960: INVERSE
220 HTAB 15: VTAB 7: PRINT " "
HTAB 15: PRINT "ALPHA": HTAB 1
5: PRINT "CAT": HTAB 15: PRIN
T: "NORMAL
230 IF PEEK (-637) < 223 THEN VT
AB 17: HTAB 6: PRINT "BE SURE <CAPS
LOCK> IS DOWN"
240 VTAB 22: PRINT "INSERT DISK TO BE R
EAD INTO DRIVE 1 AND PRESS <RETURN>
": CALL - 678
250 RW = 0: BL = 2: N = 0: FF = 1: PRINT
CHR$(4): "PREFIX,D1"
260 PRINT CHR$(4): "PREFIX": INPUT PFS
270 HOME: VTAB 4: PRINT "READING DATA.
": GOSUB 1000
280 VTAB 4: PRINT "PROCESSING DATA..."
290 IF BL < 2 THEN 340
300 NF = PEEK (24613): IF NF > 1 THEN
340
310 HOME: VTAB 10: HTAB 10: PRINT "ONLY
ONE FILE ON DISK": CHR$(7): PRIN
T: PRINT "WANT TO TRY A DIFFERENT
DISK? (Y/N)"
320 GET KS: IF KS < > "Y" THEN HOME:
END
330 HOME: GOTO 240
340 PN = 24576 + (4 + (39 * N)): IF BL
= 2 THEN PN = PN + 39
350 IF PEEK (PN) = 0 THEN 380
360 FOR I = PN TO PN + 38: FIS(FF) = FIS
(FF) + CHR$(PEEK (I)): NEXT
370 FF = FF + 1: IF FF > NF THEN 400
380 N = N + 1: IF N = 13 - (BL = 2) THE
N N = 0: BL = BL + 1: GOTO 270
390 GOTO 340
400 POKE - 16368,0: HOME: PRINT "MAIN
MENU -- USE ARROW KEYS TO SELECT"
410 FOR I = 1 TO 6: VTAB I + 3: HTAB 2:
PRINT I: "": OPS(I): NEXT
420 VTAB 3 + O: HTAB 2: INVERSE: PRINT
O: "": OPS(O): NORMAL
430 KB = PEEK (-16384): IF KB < 127
THEN 430
440 POKE - 16368,0: VTAB 3 + O: HTAB 2
: PRINT O: "": OPS(O): IF KB = 136
OR KB = 139 THEN O = O - 1: IF O =
0 THEN O = 6
450 IF KB = 149 OR KB = 138 THEN O = O
+ 1: IF O = 7 THEN O = 1
460 IF KB = 141 THEN 480
470 GOTO 420
480 ON O GOTO 540,490,710,820,920,930
490 HOME: INVERSE: PRINT "CATALOG": N
ORMAL: PRINT: PRINT "FILES ON": P
FS: "": PRINT: POKE 34,4
500 FOR I = 1 TO NF: PRINT I: "": IF I
< 10 THEN PRINT " "
510 GOSUB 950: PRINT MIDS (FIS(I),2,LN)
520 IF I / 15 = INT (I / 15) THEN PRI
NT: PRINT "PRESS A KEY TO CONTINUE
": CALL - 756: HOME
530 NEXT: PRINT: PRINT "PRESS A KEY F
OR MAIN MENU": CALL - 756: TEXT:
GOTO 400
540 HOME: INVERSE: PRINT "SORT FILES"
: NORMAL: PRINT: PRINT "CHOOSE TH
E ORDER THAT YOU WANT: "
550 PRINT: PRINT "<A>. ALPHABETICAL":
PRINT "<B>. CHRONOLOGICAL": PRINT:
PRINT "PRESS <ESC> TO CANCEL SORT"
560 KB = PEEK (-16384): IF KB = 155
OR KB = 193 OR KB = 194 THEN POKE
- 16368,0: KB = KB - 192: VTAB 2: C
ALL - 958: IF KB = - 37 THEN 350
570 ON KB GOTO 650,590
580 GOTO 560
590 D1 = 35: D2 = 34: VTAB 3: PRINT "BY
WHICH DATES?": PRINT: PRINT "<A>.
MODIFICATION DATES": PRINT "<B>. CR
EATION DATES": PRINT: PRINT "PRESS
<ESC> TO CANCEL SORT"
600 KB = PEEK (-16384): IF KB = 155
OR KB = 193 OR KB = 194 THEN POKE
- 16368,0: IF KB <
610 IF KB = 155 THEN 350
620 GOTO 600
630 IF KB = 194 THEN D1 = 26: D2 = 25
640 VTAB 2: CALL - 958: VTAB 3: PRINT
"-sorting DATES...": FOR I = 1 TO NF
: EFS(I) = MIDS (FIS(I),D1,1) + MI
DS (FIS(I),D2,1): NEXT: GOTO 660
650 VTAB 3: PRINT "SORTING FILENAMES...
": FOR I = 1 TO NF: EFS(I) = MIDS (
FIS(I),2,15): NEXT

```

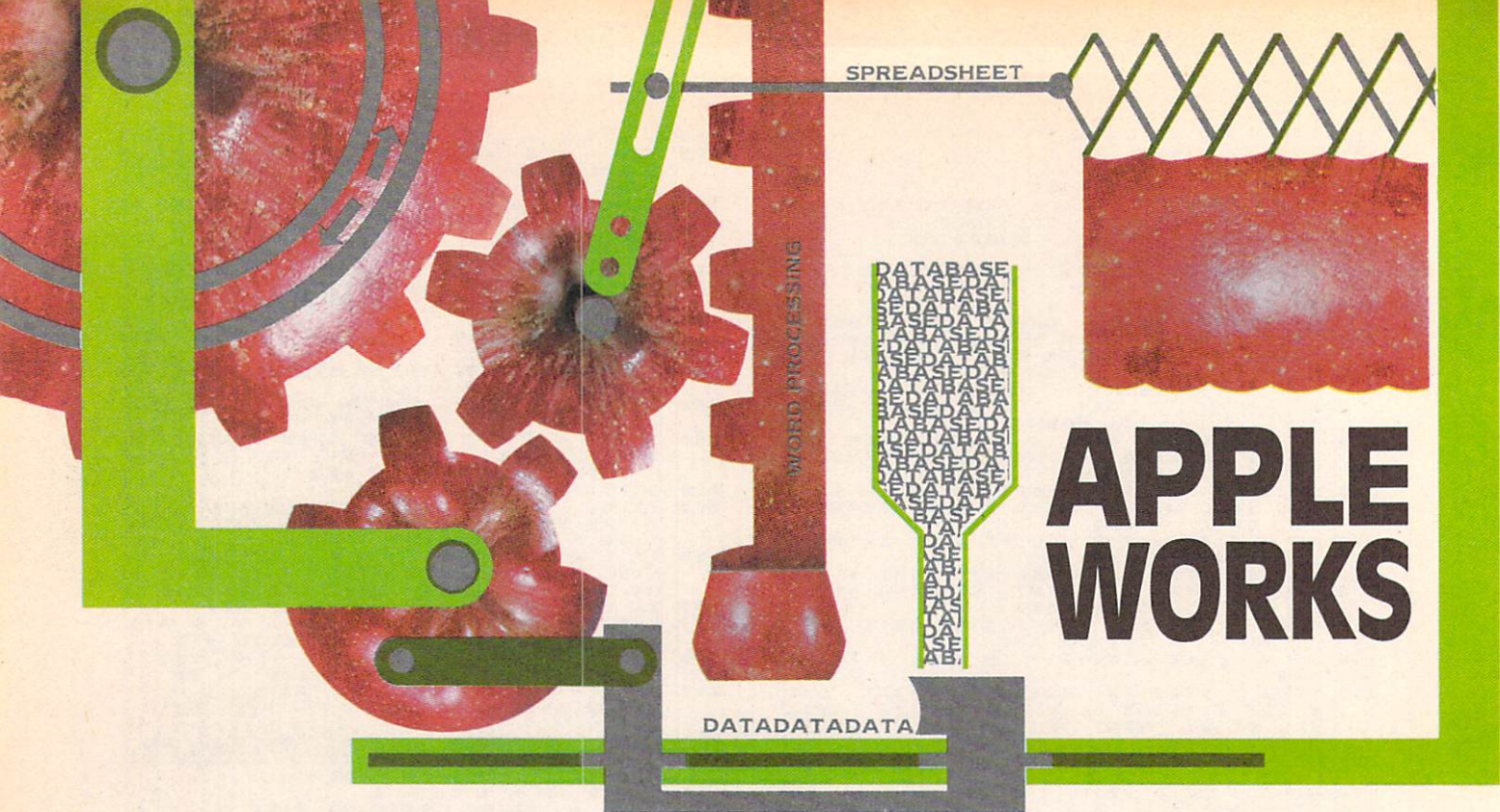
```

660 A = 1: L = 0: Z = 1
670 FOR I = A TO NF: FOR J = A TO NF: I
F EFS(I) > EFS(J) THEN A = J: GOTO I
670
680 NEXT J: L = L + Z: X1$ = FIS(I): X2$ =
EFS(I): FIS(I) = FIS(L): EFS(I) = EF
S(L): FIS(L) = X1$: EFS(L) = X2$: A =
L + Z: IF A < = NF THEN 670
690 NEXT I
700 PRINT: PRINT: PRINT "PRESS A KEY
FOR MAIN MENU": CALL - 756: GOTO
400
710 HOME: INVERSE: PRINT "MOVE A FILE
": NORMAL
720 PRINT: INPUT "ENTER NAME OF FILE>
": FS: I = 1
730 GOSUB 950: CFS = MIDS (FIS(I),2,LN)
: IF FS = CFS THEN 760
740 I = I + 1: IF I > NF THEN PRINT "N
O SUCH FILE": CHR$(7): FOR T = 1 T
O 2000: NEXT: GOTO 400
750 GOTO 730
760 PRINT: PRINT "CURRENT POSITION: "
: I: PRINT: INPUT "MOVE TO POSITION>
": PS: P = VAL (PS)
770 IF P = 0 OR P = I OR P > NF THEN P
RINT "ILLEGAL POSITION": CHR$(7): F
OR T = 1 TO 2000: NEXT: GOTO 400
780 X$ = FIS(I): IF P < I THEN 800
790 FOR J = I + 1 TO P: FIS(J - 1) = FIS
(J): NEXT: FIS(P) = X$: GOTO 810
800 FOR J = I - 1 TO P STEP - 1: FIS(J
+ 1) = FIS(J): NEXT: FIS(P) = X$
810 PRINT: PRINT "MOVED. PRESS A KEY
FOR MAIN MENU": CALL - 756: GOTO
400
820 HOME: INVERSE: PRINT "WRITING TO
DISK": NORMAL: PRINT "PLEASE ST
AND BY...": GOSUB 1020
830 N = 0: FW = 1: BL = 2
840 VTAB 3: PRINT "READING DATA..."
850 VTAB 3: PRINT "CHANGING DATA..."
860 PN = 24576 + (4 + (39 * N)): IF BL
= 2 THEN PN = PN + 39
870 FOR I = 1 TO LEN (FIS(FW)): POKE P
N + (I - 1), ASC (MIDS (FIS(FW),I,
1)): NEXT
880 FW = FW + 1: IF FW > NF THEN VTAB
3: PRINT "WRITING BACK NEW DATA..."
: RW = 1: GOSUB 1000: GOTO 910
890 N = N + 1: IF N = 13 - (BL = 2) THE
N VTAB 3: PRINT "WRITING BACK NEW
DATA...": RW = 1: GOSUB 1000: BL = BL
+ 1: N = 0: GOTO 840
900 GOTO 860
910 PRINT: PRINT "DONE. PRESS A KEY F
OR MAIN MENU": CALL - 756: GOTO 4
00
920 RUN
930 POKE 115,HI - INT (HI / 256) * 256
: POKE 116,INT (HI / 256): PRINT
CHR$(4): "PREFIX,D1"
940 HOME: PRINT "THANK YOU.": PRINT C
HR$(4): "CAT": END
950 LN = ASC (FIS(I)): CL = LN: LN = LN
- (16 * (CL > 16)) - (16 * (CL > 32
)) - (16 * (CL > 48)) - (160 * (CL
> 208)): RETURN
960 FOR I = 1 TO 6: READ OPS(I): NEXT
970 FOR I = 768 TO 779: READ V: POKE I,
V: NEXT: RETURN
980 DATA SORT FILES,SEE CURRENT SETUP
OF FILES (CATALOG),MOVE FILE TO A N
EW POSITION,WRITE MODIFIED CATALOG
TO DISK,READ ANOTHER DISK,QUIT
990 DATA 32,0,191,128,7,3,96,3,96,0,96
,2
1000 POKE 771,RW + 128: POKE 779,BL: CAL
L 768
1010 RETURN
1020 BL = 2: VTAB 3: PRINT "ERASING FORM
ER FILE POSITIONS...": PRINT CHR$(
4): "PREFIX,D1"
1030 PRINT CHR$(4): "PREFIX": INPUT CPS
: IF PFS < > CPS THEN HOME: PRIN
T "THIS DISK IS NOT ": PFS: PRINT:
PRINT "PLEASE CORRECT AND RETRY": F
OR T = 1 TO 5000: NEXT: GOTO 400
1040 RW = 0: GOSUB 1000: ST = 4: IF BL =
2 THEN ST = ST + 39
1050 FOR I = 24576 + ST TO 25088: POKE I
,0: NEXT
1060 RW = 1: GOSUB 1000: BL = BL + 1: IF
BL = 6 THEN VTAB 2: CALL - 958: R
ETURN
1070 GOTO 1040
1080 IF PEEK (222) < 16 AND PEEK (222)
> 0 THEN TEXT: HOME: PRINT "DOS
ERROR": CHR$(7): END
1090 IF PEEK (222) = 255 THEN TEXT: H
OME: PRINT "BREAK IN LINE ": PEEK
(218) + PEEK (219) * 256: END
1100 PRINT "PROGRAM ERROR IN LINE ": PEE
K (218) + PEEK (219) * 256

```

HCM





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

*They say the whole is greater than the sum of its parts,  
 but with this integrated package for home and small business,  
 the 3 parts could almost stand alone as wholes themselves.*

**W**hile heavyweight integrated programs like Lotus' 1-2-3 and Symphony have topped the bestseller lists in the productivity category for the past year, scaled-down integrated programs for the home and small-business user have been appearing more frequently. And as the need for programs that can quickly and easily manipulate financial data, records, and words *between* each application grows for this new segment, one question must not fail to be addressed by both manufacturer and user: Does each individual component of the program perform well enough to stand alone? In a limited way, the AppleWorks integrated program by Apple Computer, Inc. passes this test.

The AppleWorks program, which runs only on the IIe and IIfx, combines the 3 elements most commonly used in integrated programs—a data base, a spreadsheet, and a word processor. It requires an 80-column card and disk drive on the IIe, but for optimal use you really need a second disk drive, an extended 80-column card (bringing the memory up to 128K on the IIe), and a printer. The package includes a tutorial disk and manual, a program disk, a sample files disk, a reference manual, and a plastic-coated quick-reference card of commands.

The program's speed, friendliness, and flexibility make it a good carry-over package from the home to the small business, and in fact, it may best be used by those who operate small businesses or contract computer-

related services out of their home. Concepts and commands sidestep computerese and are instead presented in everyday terms, both in the tutorial and in the well-written 320-page manual. Using familiar concepts like a "desktop," a "clipboard," and "files" keep the program inside the realm of the average workstation, and also serve as a bridge between the Macintosh and the more abstract-command oriented programs. And what a relief it is to find that each command performs basically the same function in all three applications, and that the control keys mnemonically match their functions (C for Copy, D for Delete, W for Window).

---

*"... familiar concepts like a  
 'desktop,' a 'clipboard,' and 'files'  
 keep the program inside the realm  
 of the average workstation ..."*

---

AppleWorks features a strong visual display, with the main menu and any successive menus selected appearing on screen as index cards that overlap to show you at a glance how deep you are into the program.

It is difficult to get lost, but if you do, the (Esc) key returns you to the previous screen, and directions to get to the Help screen usually remain on screen. With the Ruler option, (OPEN APPLE) 1 quickly returns you to the top of a file, (OPEN APPLE) 9 to the bottom of a file, and (OPEN APPLE) 2-8 takes you to proportionate places in between.

### Getting Around

The Clipboard is your gateway to integrating the three applications. By cutting and pasting (Moving or Copying) to and from the Clipboard, an interim storage area,





Name:	AppleWorks
Program Type:	Integrated word processor, spreadsheet, data base.
Machines:	Apple IIe, IIc
Distributor:	Apple Computer 20525 Mariani Ave. Cupertino, CA 95014
Price:	\$250
System Requirements:	IIe: 80-column card, disk drive
	Poor Fair Good Excellent
Performance:	██████████
Ease of Use:	██████████
Documentation:	██████████

you can manipulate text within the same file, between different files of the same application, or from the Data Base and Spreadsheet to the Word Processor. The beauty of this program is the speed and ease with which you can move between several files and applications at one time. This is accomplished with the help of the Desktop, which accommodates 12 files from the disk for immediate use. It takes just a few quick keypresses to add or remove files from the Desktop. A file may contain up to 140K total characters, and a disk will hold about 51 *AppleWorks* files, according to the manual.

You may also move text between the Spreadsheet and the Data Base, but you must first print to a DIF (Data Interchange Format) file and then read from it. DIF files save data in a format that allows for the easy interchange of data between programs. This procedure means performing an extra step, but it's a handy feature nonetheless. And while the Spreadsheet and Data Base integrate in both directions, moving data to the Word Processor is a one-way street. Information in the Word Processor cannot be moved to the other two applications.

## Processing Words

Like the Data Base and Spreadsheet, the Word Processor can be broken down into two basic activities: creating documents, and formatting them for various purposes.

When creating a file, you can either toggle between the insert cursor, which pushes over everything to the right of it, or the overstrike cursor, which types new information on top of old text. Though you can move across lines by characters, words, or tabs, it would be nice if you could quickly move to the front or end of a line with one key.

Wordwrap is present, as is a handy tool called Sticky Spaces, which allows you to keep together characters that you don't want wrapping to the next line—such as a full name. Those useful standbys Find and Replace are also here, but unlike many word processors, they can distinguish between upper- and lower-case letter combinations or ignore them altogether, depending on your whim. However, they only work from the cursor location on down.

In the Word Processor, the Zoom function lets you view all of your carriage returns and other format settings. In the Data Base and Spreadsheet it displays other things that will be discussed later.

I also appreciated Markers, which allow you to "mark" places in your document so that you can quickly jump to that spot later. This is handy when you want to insert something from the Spreadsheet or Data Base later, for copying or moving, or just quickly moving your cursor. You can set up to 254 numbered markers.

*AppleWorks'* writing screen isn't cluttered looking, and it's simple to start writing right away, unless you decide to change the default settings of the document's parameters. You can do this before, during, or after you write. The default values for the main printer options—like margins, spacing, lines and characters per inch, etc.—all seem reasonable and are easy to change, but the default platen width is 8.0 inches, and most standard paper is 8.5 inches. So you must change that setting every time you set up a document, and that seems unnecessary.

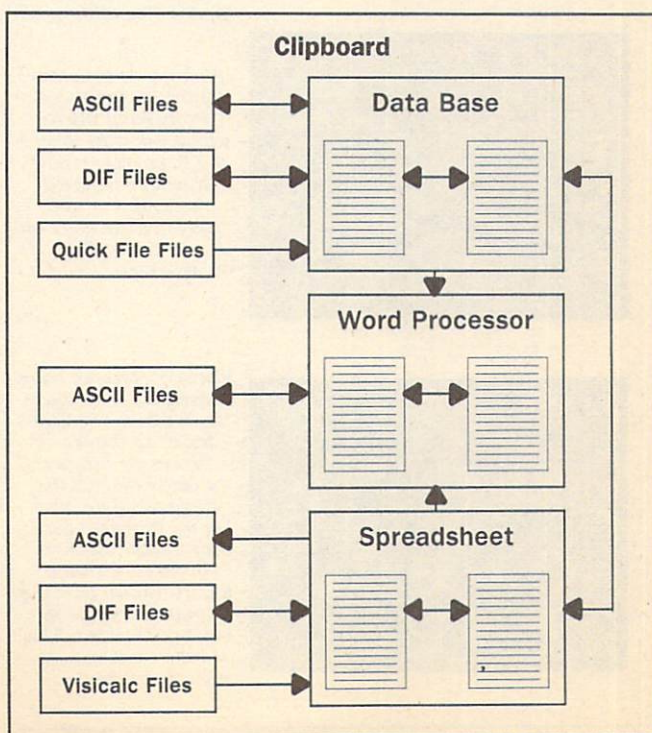
**"While the Spreadsheet and Data Base integrate in both directions, moving data to the Word Processor is a one-way street."**

Other special formatting features that allow you to control the layout of your document include options for hanging paragraphs and bullets, justification of the sides of your text, and page headers and footers. You can set up your own pages, or let *AppleWorks* do it for you. When *AppleWorks* breaks a page, it attempts to keep paragraphs intact. This is a nice feature. Unfortunately, you can't set up several files to automatically print out sequentially. Perhaps to counterbalance this, the program does let you pause during a printout to type in information, and then it continues printing.

Because the Word Processor keeps all of its information in memory, it works extremely fast—it doesn't have to keep stopping to access the disk.

However, this limits your documents to about 26 pages of 54-line, single-spaced text with 128K, and only 8 pages when using just 64K. Also, don't look for a spelling checker or mail-merge capabilities. They weren't included. [*MegaWorks*, a \$125 package from Megahaus Corp., provides the spelling checker and mail-merge capabilities "missing" from *AppleWorks*.—Ed.]

Interestingly, *AppleWorks* can also convert and use ASCII-format text files from outside programs. The use





## INTEGRATING TWO APPLICATIONS

Here is a display of the steps involved in putting some *AppleWorks* Data Base information into a *Word Processing* memo (to the director of the pie company used in examples in the tutorial manual).

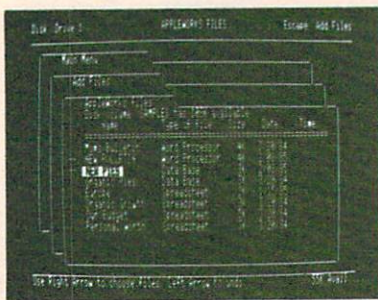


Photo 1. Here we've selected the files that we wish to add to the Desktop for current use—New Pies (a Data Base file), and New Bulletin (a Word Processor file).

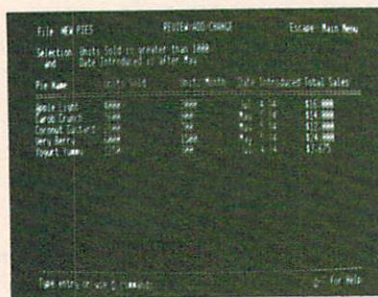


Photo 2. Using the Record Selection Rules option on the New Pies file, we've extracted all of the pie brands with sales of more than 1000, and that were introduced into the product line after May. This is the information that we want to insert in our memo.



Photo 3. Before we can Print this file to the Clipboard and then to the Word Processor, we must format it to fit. The Report Format screen displays all of the options available to do this. We deleted some categories so that the report length is only 60 characters wide, which will allow it to fit into the memo. We will also have a total calculated for the Total Sales column.

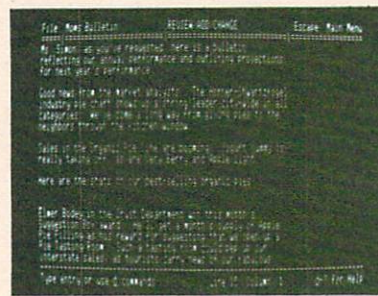


Photo 4. This is New Bulletin, the Word Processing file to which we want to add the Data Base report. It will be inserted beneath the line "Here are the stats on our best-selling organic pies."

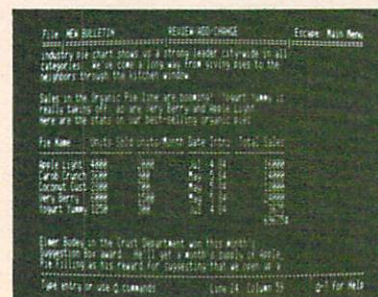


Photo 5. Here we have Moved our Data Base report from the Clipboard to the Word Processor file. We already deleted the filename, date, and other formatting information that was inserted with the report. Notice that the Total Sales column has been totaled. This is the final appearance of the memo.

of the powerful text editor could be a nifty feature for assembly language or Pascal programmers. When you finish editing such a file, you could save it as ASCII to return it back to its original source for compiling. This would take experimentation and is not something for beginners. However, it is worth investigating.

## Summing Cells

The key to making a good, useful spreadsheet is to know before you begin what answers you want calculated, and what kind of format or report you would like to end up with. But the hardest part is actually creating your spreadsheet; the complicated process is inherently migraine-breeding. *AppleWorks*, however, does a commendable job of trying to make it easy on you, and once you've done one or two, you probably won't even need to consult the manual.

Like the Word Processor screen, the Spreadsheet screen is pretty bare, leaving you plenty of room to view your work. Across the top are letters representing the default column widths, and down the left side are row numbers. However, you are not restricted to using the typical column and formula type Spreadsheet—you can format the Spreadsheet to accommodate financial statements, amortization schedules, or a broad range of other numeric reports.

**"You are not allowed to Copy or Move part of a record—you've got to take the whole thing... or nothing at all."**

Once you've decided how you want your spreadsheet to look, you can set your Standard Values for labels, values, column widths, and recalculations. Labels may be left- or right-justified, or centered, while values may be displayed with or without commas, dollar signs, fixed decimals, or as percents. Column widths may vary from 1 to 75 characters and can change throughout the spreadsheet if desired. When you enter a new value into the spreadsheet, you can either have the program automatically recalculate the sheet, or you can do it manually. Because the entire spreadsheet disappears and reprints on the screen every time you change a value or use a command (which is rather annoying), you may want to set this standard on Manual if you have to make a lot of changes—so that it recalculates only once. For flexibility's sake you may override the Standard Values for any cells you specify so that you can tinker with the format. To find out the current settings of the Standard Values at any time, simply call up the Help screen.

Editing the Spreadsheet is very simple and uses the same editing commands of the other two applications. Here the Zoom command serves to display the formulas for every cell, and you can print this out while you are zoomed in.

Two exceptions to the simple commands are Move and Copy. The program will not allow you to identify blocks for these functions unless you specifically Print to the Clipboard for a printout or placement in a Word Processing file. You may Move to the Clipboard by rows only, and within the spreadsheet by entire rows or columns. You may Copy to the Clipboard by rows only, and Copy within the spreadsheet in one direction—either up or down in a column or left or right in a row—



---

***"Another limitation occurs in keeping  
a whole file in memory at once . . .  
What you gain in speed,  
you lose in capacity."***

---

depending on where the cursor is when you choose to Copy. Moving and Copying become even trickier

when formulas refer to other cells in the worksheet. However, this can be handled by specifying whether the formulas should be altered relative to their new position, or should remain unchanged.

Calculating formulas may be difficult for business users, who will be disappointed to find that there is only one financial function—Net Present Value—and one logical function—IF—provided in the program.

For frequently-used spreadsheets, you can protect against accidental changing of labels, values, or all cells with the Protection option. And to see part of a sheet that may be out of view, you can work with a spreadsheet split horizontally or vertically—changing your screen from one window into two that scroll in synchronized fashion, or separately.

According to the manual, Spreadsheet files can contain 127 columns and 999 rows for a maximum of 126,873 empty cells. But if you want to discuss filled cells, that is another matter entirely, with 64K RAM allowing about 1000 filled cells, and 128K RAM providing for about 6000 filled cells. That is an enormous discrepancy in figures.

### **Dealing With Data**

The *AppleWorks* Data Base does not look much different from the Spreadsheet—it presents numbers and words in cells, with the rows comprised of records and the columns made up of the categories, or "fields" within the records. The only difference is that these cells aren't calculated together. (However, you can get group or report totals for specified numeric categories as well as 3 calculated fields.)

The menu-driven nature of the program makes it easy to set up a new file or edit an existing one in the other two applications, but in the Data Base some of the screen prompts are vague, and the manual lapses from its normally clear presentation. For example, after you indicate that you want to start a file from scratch and enter a title for it, a screen pops up that is divided in two: on one side it says **Category names** and the cursor is blinking on **Category 1**. On the other side are some options to use in entering category names. Apparently what you are supposed to do is delete **Category 1**, type in your own category names with a (RETURN) following each one, and when you are done press (Esc) to go to the next task. I discovered this mainly through trial and error.

Records are displayed in either single- or multiple-record layouts. The single layout lists one record's categories and its contents vertically, and the multiple layout lists all the records, with the categories stretching out horizontally. Here the Zoom feature toggles you between the two. Although you can choose the categories that you want to display on screen if they stretch beyond the screen width, oddly enough you cannot scroll over to view the missing categories.

Also odd is the fact that *AppleWorks* has included a Find command for the Data Base, but left out its complement the Replace command. The other editing commands are consistent with the rest of the program, except that the Move command will only place records below the cursor's current position, which prevents you from moving anything down to the bottom line of the report. You are also not allowed to Copy or Move part of a record—you've got to take the whole thing. . . or nothing at all.

Another limitation occurs again in keeping a whole file in memory at once. With an average record size of 75

characters, an Apple equipped with only 64K can hold only about 140 records; an Apple with 128K, 750—surely not enough for many business applications. And, even if you have only one byte per record, you can only use a maximum of 1350 records. What you gain in speed, you lose in capacity.

One good feature of the Data Base is its date and time formats for categories. By including the words "date" or "time" in some form as part of a category name—such as "Birthdate" or "StartTime"—*AppleWorks* will convert your entries in that category to the proper date or time format and add a.m. or p.m. designations. This is especially handy if you later want to sort your records by date or time. The Arrange option will sort one field at a time in numerical or alphabetical order, from highest to lowest or vice versa. If you want more than one field per file sorted, you have to keep repeating the Arrange procedure.

The Data Base is quite flexible in the ways it allows you to extract one or several records from a file with its Record Selection Rules option. In addition to searching for records with certain character combinations, the program provides a variety of terms that you can use to set up one or more conditions for selecting records. An example might be to highlight one of your categories, say, Monthly Sales: if you choose the option is greater than and type in \$10,000, the program will give you a list of all months in the file when sales exceeded \$10,000.

For screen displays, print-outs, or insertion into Word Processing files, you can choose between two basic report styles: a tables format, with yet more rows and columns; and a labels format, with a list look. You can choose the categories and records to be included, adjust their lengths and justify their sides, set margins and spacing, and make other decisions with the help of a menu, and save these formats for repeated use. But beware: one little change in the file's structure, such as deleting or moving categories, and you lose all formats based on that file—a bit of over-reaction, I'd say.

Compared to the wide variety of print options offered in the Word Processor, the Data Base's print options are pretty slim pickings—no underlines, boldface, or other "special" features are available here. Of course, you could first paste the report into a Word Processor file, and then use its features to reformat your report.

### **Wrap-up**

The Spreadsheet and Word Processor have a few minor drawbacks, but on the whole they work quite well by themselves and together. The Data Base is adequate, but a bit weaker and more limited than the other two applications. They are all fast and easy to use. Because of *AppleWork's* price, some people may at first be put off by this package, figuring that they could adequately fill their needs by buying solo programs that cost less. Others who could easily afford this program might be hampered by some of its business-related limitations, and would perhaps opt for something more extensive. But for students, small-business owners and workers, and others who find that they have a need for a good integrated program that won't overburden them with more commands and options than their needs dictate, this is a fair price, and a solid productivity package for the Apple.

HCM





# Commodore Hornblower

## Inside The SID Chip

by Roger Wood

HCM Staff

*Learn about "SID's Tricks" and start building your own BASIC Synthesizer on the sound-rich C-64 . . .*

Elsewhere in this magazine is a review of a number of exciting hardware and software products that make using the SID chip literally child's play (see *The Music of Sound*, pages 32-37). For those who want to use a "do-it-yourself" approach, this article is the first in a series that will help you build your own BASIC electronic synthesizer. We will begin by explaining the SID chip and the "tricks" that it can perform.

Commodore boasts that the C-64 "is equipped with one of the most sophisticated electronic music synthesizers available on any home computer." Anyone familiar with sound synthesis who looks at the Sound Interface Device (SID) chip's capabilities has to agree.

To start with, the chip is "polyphonic"—that is, its 3 oscillators (sound generators) can create 3 distinct tones or "voices." Each voice is completely programmable with 4 distinctive waveforms and its own dedicated Envelope Generator. In addition, each voice can be run through a programmable filter to further enhance its sound. Finally, the 3 voices can be made to work together through synchronization and various forms of modulation, where one voice controls or "modulates" another.

### Four Waveforms

The SID chip can create the triangle, sawtooth, pulse, and noise waveforms (see Figure 1). The triangle wave is an even, regular wave that creates a mellow, flute-like tone. Of all the waves available in the SID chip, this one is closest to the basic wave of all sound—the sine wave. The sawtooth wave, also called a ramp, has a more brazen tonal quality—like a saxophone. Due to the added harmonics present in this waveform, it tends to be the loudest and have the most presence of all the waves available in the C-64.

Where the triangle and sawtooth waves tend to vary in an analog or fluid way, the pulse wave is more digital in nature, being either "on" or "off." It can produce a wide variety of tonal qualities by altering the "width" of the pulse, or the amount of time the pulse is "on" or "off." Each oscillator in the SID chip has a pair of registers which determine the pulse width. The more

regular, square pulse wave ("on" time equal to "off" time) creates a hollow and bright tone, and the more uneven pulse waves ("on" time not equal to "off" time) possess a more nasal, reedy quality. Figure 1 shows 2 different pulse waves: one square wave and one with very narrow pulse width.

The fourth waveform, called noise, is irregular in nature, and, depending upon the frequency selected, creates sounds ranging from a low grumbling to a high hissing noise. It is most useful in creating the numerous types of sound effects heard in the arcade-type games for the C-64.

### Envelope Control

Sound is a time-related phenomenon. No sound exists outside of time—rather, it is the relationship between a particular waveform's amplitude (or loudness) to time that gives a sound its unique character. The SID chip contains a master volume control for all 3 oscillators; that is, the maximum volume of any sound the SID chip creates. Each of the oscillators becomes active when a "gate" signal is sent to it, and the oscillator remains active until the "gate" is closed.

The loudness of a particular tone at any point in time is controlled by two registers for each voice, known as the ADSR. This name is an acronym for 4 parameters: Attack, Decay, Sustain, and Release. Figure 2 provides a graph showing how these parameters relate to the duration and amplitude of a note.

The Attack is the amount of time it takes a particular note or sound to go from zero to the maximum volume set by the master volume. The Decay is the time for the note to decrease from maximum volume to what is called the Sustain level. This is the volume that the sound will settle at until the gate signal is shut off.



Figure 1

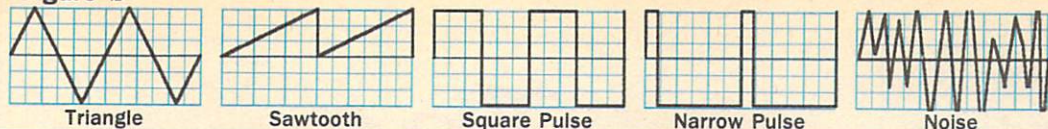
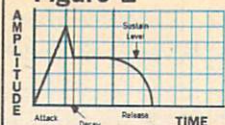


Figure 2



Release is the time it takes for the tone to go from the Sustain level to zero volume once the gate signal is removed. The amount of time for Attack and Decay can vary from 2 milliseconds to 8 seconds. Release time can be varied between 6 milliseconds and 24 seconds.

### Filter, Synchronization, and Modulation

All output from the 3 oscillators can be selectively modified by a programmable filter. The filter becomes active around a particular frequency called the "cutoff" frequency. The filter has 3 different modes: high pass, low pass, and band pass. The high-pass filter "attenuates" (turns down the volume on) tones below the cutoff frequency, and "passes" (does not turn down the volume on) tones above the cutoff frequency. Similarly, the low-pass filter attenuates frequencies above the cutoff frequency and passes those below it. Finally, the band-pass filter passes only those frequencies around the cutoff frequency.

Not only do the separate oscillators function independently, but they also can be programmed to work together. They can be synchronized so that their fundamental frequencies are actually in sync, which allows for some exciting harmonic effects. And, a Ring Modulation capability (the mathematical combination of the outputs from 2 oscillators) makes all sorts of gong and bell effects possible. In addition, Oscillator number 3 can be set up to frequency modulate the other 2 oscillators—giving the C-64 some amazingly sophisticated electronic sound abilities.

### Getting SID To Play

The only real difficulty associated with this marvelous synthesizer is in making it play. Can't people who want to use Commodore BASIC write their own music and sound programs? This article is the first in a series that will help you do just that. In the next few issues of HCM, we will build a BASIC electronic synthesizer using the Commodore 64's own BASIC language to tap the SID chip's sound capabilities.

### Translating Frequency To SID POKES

In SID, the exact frequency of the sound is determined for each of the 3 oscillators by 2 frequency-control registers. But the frequency of the note you wish to play must first be converted by a mathematical formula to obtain the proper values. The *Commodore 64 User's Guide* contains a list for converting any note into the proper POKE values. However, this means that in order to play each note, you, the programmer, have to look up the note and include its POKE value in the program. This is usually done in many DATA statements. In order to include all 8 octaves (96 notes), 192 different numbers must be placed into variables to allow a BASIC program to have all of the notes at its beck and call.

The accompanying program is a far more memory-efficient method of arriving at these POKE values. It mathematically calculates (through the use of 4 functions in lines 450 and 460) the values of all the POKE values for the entire 8 octaves, based upon the value of the lowest note—a C note of frequency 16.3515977 Hertz (cycles per second). It stores the values in a 3-dimensional, integer array NT%( ). The first subscript is a number from 1 to 12, which identifies the note: 1 is C, 2 is C#, 3 is D, etc. The second subscript is a number from 0 to 7, which designates the octave of the note.

The third subscript is either 0 or 1. The zero element will give the low-byte POKE value, and the one element will give the high-byte POKE value. Thus, to have oscillator 1 play an A-440, you would POKE its frequency control registers with NT%(10,4,0) and NT%(10,4,1).

If you wish to include this feature in one of your programs, you only need to copy the subroutine from lines 450-520. This routine takes approximately 15 seconds to run; when the array is full, the entire cost in memory is only 959 bytes.

The program also includes a very simple keyboard routine that allows you to play any of the "natural" notes (white keys on the piano) by pressing the corresponding letter key. To access notes in a higher octave, press + and a lower octave by pressing -. This program uses only oscillator 1 and has "hard-coded" a sawtooth wave and ADSR settings in lines 280-290. In future installments, we will be expanding this program to give you more insight into how to program the SID chip from BASIC, and build a complete music synthesizer.

HCM

```

100 REM *****
110 REM * CREATE FREQUENCY DEMO *
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 5.2.1
180 REM COMMODORE 64 BASIC
190 REM
200 PRINT "SHIFT CLR 2 CRSRDOWN PREP
   ARING SID POKE VALUES":PRINT "CRSR
   DOWN PLEASE WAIT":GOSUB430
210 PRINT "SHIFT CLR 2 CRSRDOWN READ
   Y"
220 PRINT "2 CRSRDOWN PRESS LETTER TO
   PLAY THAT NOTE"
230 PRINT "2 CRSRDOWN PRESS + TO GO U
   P AN OCTAVE"
240 PRINT "2 CRSRDOWN PRESS - TO GO D
   OWN AN OCTAVE"
250 PRINT "2 CRSRDOWN PRESS - TO QUIT
   "
260 FOR I=S TO VL(0):POKE I,0:NEXT
270 VL(1)=15:POKE VL(0),VL(1):OC=4
280 AD(1)=21:SR(1)=165:CR(1)=32
290 POKE AD(0),AD(1):POKE SR(0),SR(1):P
   OKE CR(0),CR(1)
300 K=PEEK(197):IF K=64 THEN300
310 IF K=40 THEN OC=OC+1:IF OC>7 THEN O
   C=7
320 IF K=43 THEN OC=OC-1:IF OC<0 THEN O
   C=0
330 IF K=40 OR K=43 THEN FOR DE=1 TO 10
   0:NEXT:GOTO300
340 NT=(-1)*((K=20)+(3*(K=18)))+(5*(K=14
   ))+(6*(K=21)):IF NT THEN380
350 NT=(-1)*((8*(K=26)))+(10*(K=10))+(12
   *(K=28))
360 IF K=57 THEN POKE VL(0),0:POKE 198
   ,0:END
370 IF NT=0 THEN300
380 POKE FL,NT%(NT,OC,0):POKE FH,NT%(NT
   ,OC,1)
390 POKE CR(0),CR(1)+1
400 K1=PEEK(197):IF K1=K THEN400
410 POKE CR(0),CR(1)
420 GOTO300
430 DIM CR(1):DIM AD(1):DIM SR(1):DIM V
   L(1)
440 S=54272:FL=S:FH=S+1:CR(0)=S+4:AD(0)
   =S+5:SR(0)=S+6:VL(0)=S+24
450 DEF FN PF(FR)=FR/.06096:DEF FN FR(B
   A)=BA*(1.05946309)
460 DEF FN PH(FR)=INT(FR/256):DEF FN PL
   (FR)=INT(FR-(256*(INT(FR/256))))
470 DIM NT%(12,7,2)
480 FQ=16.3515977:FOR OC=0 TO 7:FOR NT=
   1 TO 12
490 NT%(NT,OC,0)=FN PL(FN PF(FQ))
500 NT%(NT,OC,1)=FN PH(FN PF(FQ))
510 FQ=FN FR(FQ):NEXT NT:NEXT OC
520 RETURN

```



*"To underestimate your enemy's strengths is to lose the battle"*

—Sun Tzu.

With these words of advice, we begin *The Ancient Art of War*. This is a game of strategy, with two generals: you, and your opponent—a famous figure out of the bloody pages of history, placed in a battlefield in a fight to the finish.

This game is based on principles from the book *The Art of War* written by Chinese philosopher/general Sun Tzu in 400 AD. (He is also the toughest general you will have to face in the game.) This book is still considered one of the most important studies of strategy and tactics ever written, and was studied by Ghengis Khan, Napoleon, and Mao Tse Tung.

In the book and in this game there are four basic types of war: attrition, guerilla, mass and maneuver, and siege. Each type requires its own strategies and tactics if you are going to survive and capture your opponent's flag. As you play the game, your strategies may need to change as your enemy's strategy unfolds.

### Pick a Fight, Any Fight

You begin the game by reading the ancient scrolls, which tell you the story of the battle you are about to fight and name the general whom you will face. You also have an option to change the rules, and can even decide which general you will be facing. (If you don't feel up to tackling Napoleon, you can start out with Crazy Ivan.) This means that the 11 preset campaigns can be changed by you before play, actually creating many different battle scenarios.

*The Ancient Art of War* recreates 8 legendary generals as your opponents and endows them with the capabilities, and in some cases the quirks that each one had when they lived and fought. Each general is accompanied by a different number of squads—sometimes you are outnumbered, sometimes not. Each squad is represented by a small

# The Ancient Art of War

A review by Steve Nelson

HCM Staff

*Heed the advice of Sun Tzu, and who knows . . . maybe you will be the next general to conquer the world.*

figure somewhere on the battlefield. Each figure represents up to 14 knights, archers, barbarians, or a combination of the above.

The game can be played in 4 different speeds, which can be changed at any time during play with one keypress.

Every campaign has its own set of obstacles: rugged mountains, dense forests, and treacherous water crossings. Each of these can cause you to lose a squad, so keep this in mind as you play. The type of terrain where the battle is fought has a definite influence on the battle's outcome. The more rugged the playing field, the greater the chance that your men will be killed in transit, or will arrive tired and hungry. The more fatigued your men are, the less effective they will be in battle. You can replenish your food supplies along the way by traveling through villages.

### Building a Battlefield

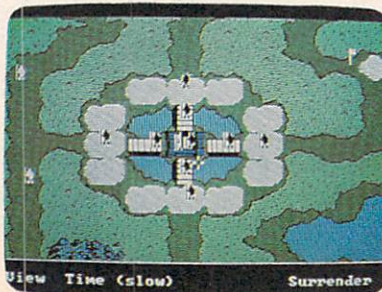
The most exciting part of this game, for me, was creating my own

battles. From the main menu, you can access the game generator. There you start with a blank screen which represents a section of the battlefield and a grid showing you different types of terrain—forests, rivers, ponds, hills, etc. At the beginning of the battle you select obstacles and terrain from this grid and design the battlefield, setting up the actual battle by planting the flags, creating the armies, and deciding where they will be deployed. At the beginning of each battle, the ancient scrolls open to let you give a description of the field of battle and the reason why the battle is being fought. I dreamt up all kinds of crazy reasons for people to fight. Once the game is under way, the computer takes control of the black army.

Directing your squads into battle is accomplished by moving the cursor over each squad and pressing M, then moving the cursor to a point on the battlefield where you wish to move the squad and pressing M again. Although this may seem like

*"This is a game of strategy, with two generals: you, and your opponent—a famous figure out of the bloody pages of history, placed in a battlefield in a fight to the finish."*

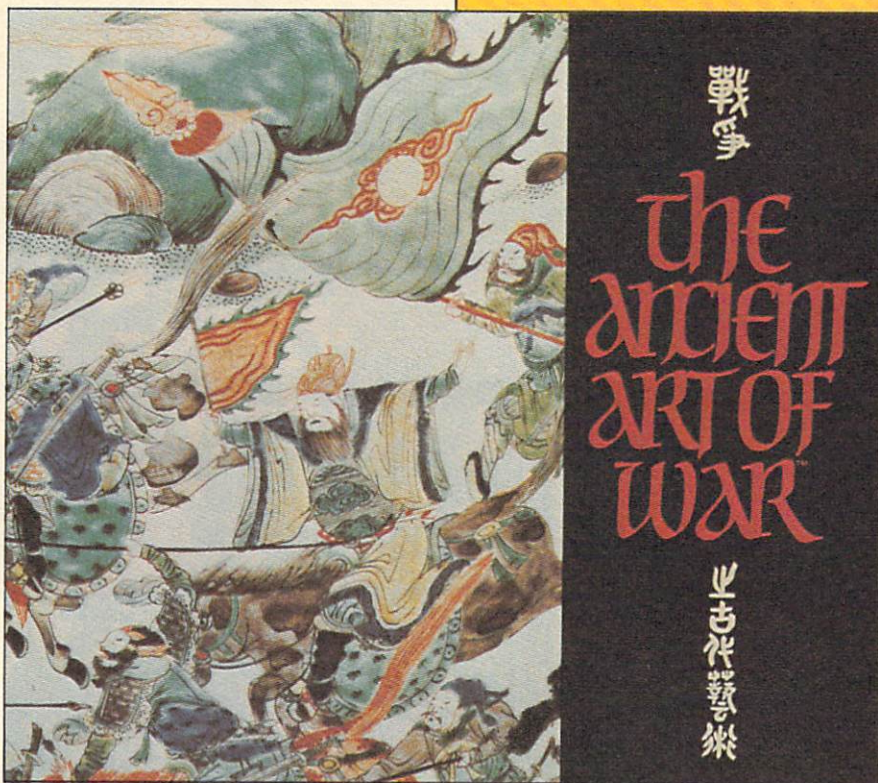
This photo shows troop emplacements in an actual battle scene around a fort.



The zoom feature puts you right in the thick of battle.







Name: The Ancient Art of War  
 Program Type: Adventure game  
 Machines: IBM PC, PCjr  
 Distributor: Broderbund Software  
 17 Paul Drive  
 San Rafael, CA. 94903  
 Price: \$44.95  
 System Requirements:  
 IBM PC: 128K, color/adaptor card, RGB monitor  
 IBM PCjr: 128K, color monitor  
 Performance: Poor Fair Good Excellent  
 PC: \_\_\_\_\_  
 PCjr: \_\_\_\_\_  
 Engrossment: \_\_\_\_\_  
 Documentation: \_\_\_\_\_

*"The 11 preset campaigns can be changed by you before play, actually creating many different battle scenarios."*

edge—although, as they say, all is fair . . . And, although the graphics are impressive, the game could be spiced up with some sound effects during the battles.

It took a few battles before I began to plan a real strategy, rather than engaging the enemy whenever and wherever I encountered him. Because the emphasis of this game is on developing strategies and waging campaigns, my initial approach led to many lost battles and a lot of frustration.

Documentation provided with the game is excellent. It is creatively packaged, is very easy to understand, and comes with a quick-reference card for use while playing.

Overall, I was quite impressed with this game. The only real complaint I have is that the keyboard response on the PCjr could be better—other than that, it is a fine game. *The Ancient Art of War* has all the action of a video game and the interest of a text-oriented adventure, and it combines the two into a rather pleasing package.

HCM

an inefficient way to move, it works rather well. The pace of the game in its slowest mode is slow enough so that you can make all of your moves without losing much ground to the enemy. Once under way, you can speed up the time and begin the battle.

As you are playing, you can move the cursor freely about the screen and gather information on the strength of the enemy. You will receive news from the front notifying you when the enemy is sighted (spies can help you see the enemy faster), and when there is an encounter between two squads. Once two squads meet, you must move the cursor over to them and press the Z key, activating the Zoom mode. You are then transported to a close-up view of the battle scene, with your squad on one side of the screen and the enemy's on the other. From there

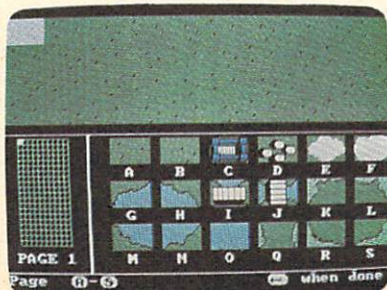
you can direct the battle, attacking, feinting, sending in only a portion of your squad while others lay back in reserve, or even retreating. Your strategy here is crucial to the success of your battle.

The graphics are first rate, especially when in the Zoom mode. Close-ups of villages, forts, and bridges are quite detailed, and the animation of the battles is very good.

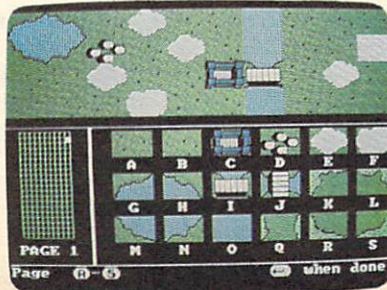
## War is Imperfect

*The Ancient Art of War* is a fine game, but it does have a few problems. The keyboard response on the PCjr is slow compared to the PC. However, this is probably not a fault of the program; rather it has to do with the way the two machines handle video access. It also seemed to me that the enemy's squads almost always moved a lot faster than mine, giving the computer an unfair

This photo shows the map board feature of the game generator.



You can assemble the pieces of your battlefield into any configuration.



This photo shows the placing of the squads after the battlefield has been made.





# LEGACY II for the PCjr

A Review  
by David G. Brader  
HCM Staff



*More than just flashing red lights, this little beauty adds a second disk drive and expansion slots to the humble Junior.*

What can you do with a PCjr that has one disk drive and 128K of memory? You can do a lot of things, but not very conveniently without a second disk drive. One way you could add a second disk drive is to actually try installing it yourself. [See the article entitled, "One for the Money, Two for the Slow: Adding a Second Drive to the PCjr" in *Home Computer Magazine* Vol. 4, No. 4—Ed.] Or, you could find a third-party manufacturer who will supply a second disk drive. But one of the main problems with adding a second disk drive to the PCjr is that the IBM-supplied, floppy-disk controller card is designed specifically to address only one floppy disk drive.

Three ways to get around this problem exist: The first option is to modify the IBM-supplied disk controller card. The second is to throw it away and replace it with a totally new disk drive controller card that will address more than one floppy disk drive. The last method available is to *not* buy the IBM PCjr with the disk drive controller, and instead start with a third-party controller and drive already inside the Junior.

Legacy Technologies, Ltd. supplies hardware and software to satisfy both options two and three, above. Because most of our readers who currently own PCjrs purchased them with the IBM disk drive, controller, and 128K of memory, we will cover the Legacy Technologies expansion system that was engineered for that configuration—i.e., the *Legacy II*.

## Legacy II—The Inside Story

When getting started with the expanded PCjr, it is really annoying to just throw away an expensive electronic device such as the IBM disk drive controller, so we won't. We'll keep it as a spare in case the *Legacy II* disk drive controller board ever fails. (Of course, we will be reduced to single-drive operation again during any repair time, but that is better than having the system down totally.)

The disk drive controller board isn't the only thing that comes in the *Legacy II*—the whole package consists of a case (the same size and shape as the PCjr) containing the second disk drive, an interesting front-panel display, power supply, and expansion slots. The front-panel display is mostly a row of flashing red lights. These lights, or LED's (light emitting diodes), are driven by the various internal signals of the computer "bus," and in this day and age are as useful as the twinkling lights on your Christmas tree—they're pretty to look at, but they serve no important purpose. I realize that a few diehard assembly-language programmers will use those lights for debugging, but for the majority of people they represent nothing more than a pretty light show.

Behind this pretty facade reside 4 expansion card slots. These slots will accept Legacy Technologies, Ltd. expansion cards. Legacy has published the card-connector signal definitions, which they are calling the *L-bus*. Two expansion cards are currently available for the *L-bus* from the company: One is a memory card that holds up to 256K of dynamic RAM chips (a fully populated board has a suggested retail price of \$395). The second card available is a combination clock/calendar chronograph and parallel printer port unit that comes with a print spooler program (suggested retail price: \$159).

## A Tight Squeeze on the Memory Card

The *Legacy II* unit, as supplied to us, included one of the fully populated memory cards. The major complaint that I have about this product is the difficulty involved in installing the expansion cards. The memory card supplied with the unit had to fit into the bottom slot in the case. It so happens that the position of the power supply in the back of the case and the actual depth of the case preclude inserting the card straight into the *L-bus* connector. I had to start the card into the slot at an angle and then force it in, slightly bending





**PHOTO 1:** Everything shown here comes in the Legacy II system kit, with the exception of the fully populated memory card. The memory card is shown in the left foreground above.

the printed circuit card itself, until it snapped into the socket—it's a most uncomfortable sound and one that causes sweaty palms from fear of damaging the memory board, the case, or the disk drive. Any operation that requires undue force is one that should have been engineered out of the product. I did try to pull the card out and insert it a couple of times to see if there was any damage. I was successful in completing the "hair-raising" operation twice and the system still worked, but a quarter of an inch more in depth, or a skinnier power supply certainly would be greatly appreciated. [When asked about this, a company source informed us that the LED display will be removed from a future version of the Legacy expansion system—increasing the space for card insertion—Ed.]

### Adding a Second Story to the Junior

Once the memory board is installed into its uncomfortable position, the procedure for installing the *Legacy II* is clearly described with text and illustrations in the rather terse manual.

The Legacy unit sits directly on top of the PCjr as a free-standing separate piece. The two flat-ribbon cables (one for the L-bus and one for the disk drive) are the only ties between the two units. These cables extend from the bottom-right of the Legacy chassis (which is about an inch wider than the PCjr case). The Legacy-supplied dummy cover is installed to protect the cables that are connecting the Legacy unit to the Junior. Aligning the *Legacy II* so it sits squarely on top of the Junior produces a very handsome computer system—unless you have an IBM side-expansion module (such as the parallel printer port) already installed. In this case the Legacy unit is shorter than the PCjr and must be offset to the right. If more than one side-expansion module is attached, the spread between the L-bus and disk drive cables tend to become a problem. This is because the disk drive cable must connect inside the PCjr and the L-bus cable must plug into the outside of the last side-expansion module.

Plugging the power cable into the back of the Legacy unit (the other end into a power outlet) and switching it on prepares the system for operation. Now, when you

switch on the PCjr power button you can watch the *Legacy II*'s lights flicker and blink as the unit goes through its self-test. By watching the screen in the lower right-hand corner, you will notice that the memory test cycle will go right through the normal 128K all the way up through 256K and finally end at 384K of memory—pretty impressive. But once you boot up DOS, you'll discover that you won't be able to access the second disk drive or the additional memory, so it's back to reading the *Legacy II* instruction manual which, by the way, consists of 13 loose-leaf pages—not even bound together. Fortunately, it does (barely) have sufficient instructions to get the system operational. I was told by Legacy that they have a new manual with more pages and more information that covers usage of the system, but it was not provided in time for this review.

When checking through the packing material, you will find a diskette called the LEGACY SYSFIX DISKETTE. This, along with some instructions in the manual, will allow you to generate a special PC-DOS disk that will recognize the additional memory and the second disk drive. This operation really is quite simple and takes about five minutes. From that point on you always use this specially-prepared "Legacy PC-DOS boot disk" to bring up your system. I tried several different DOS commands to ensure that the system worked properly.

The only DOS command that I had any problem with was the DISKCOPY command. For some reason, during the actual operation of the DISKCOPY, the PCjr's video memory got overwritten and the screen image showed nothing but scrambled characters. The first time this happened, I thought that the entire system had gone bananas and I shut it down. The second time it happened, I just waited until the disk drives stopped running. Then I assumed that the DISKCOPY had been completed, even though it wasn't showing on the screen, and rebooted the system. After the reboot, I checked the new copy of the diskette (produced under DISKCOPY before the power was shut down) to see whether it was in fact a complete copy. It was. Apparently, the DISKCOPY function does perform properly, but the usage to which Legacy puts the video memory gets totally wiped out

Name:	Legacy II
Description:	Peripheral expansion system with disk drive and expansion card slots. Also included: a replacement disk drive controller (that can handle the extra disk drive), cables, and software to modify the PC-DOS boot disk to allow the PCjr to recognize the Legacy unit.
Machine:	IBM PCjr
Distributor:	Legacy Technologies, Ltd. 4817 North 56th Street Lincoln, Nebraska 68504 (402) 466-8108.
Price:	\$795 without expansion cards.
System Requirements:	IBM PCjr with 128K memory and the IBM internal disk drive.
	Poor Fair Good Excellent
Performance:	████████████████████
Ease of Set-up:	████████████████
Documentation:	████████████████
Cost/Benefit:	████████████████

*"... a few diehard assembly-language programmers will use the front-panel lights for debugging—but for the majority, they represent nothing more than a pretty light show."*



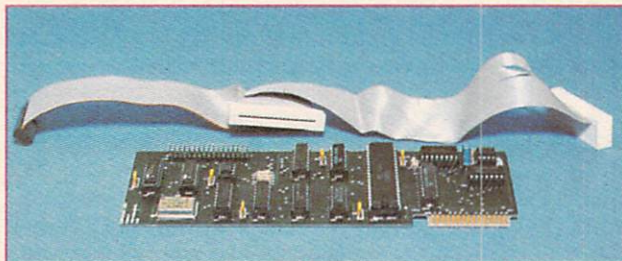
by the *DISKCOPY* command. By reading further in the manual, I found this statement about the *LEGACY.COM* file. "This program rearranges some of the PCjr memory so you will be able to run more of the PC programs." Sure enough, as soon as that file executed and I retried the *DISKCOPY* command, the apparent flaw disappeared.

---

**"... it is really annoying to just throw away an expensive electronic device such as the IBM disk drive controller ..."**

---

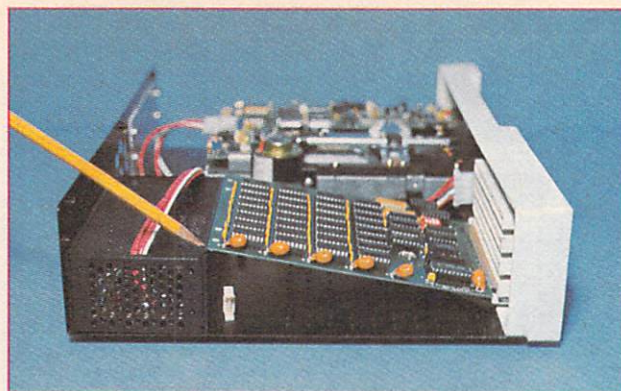
Well, there isn't much else to say about the *Legacy II* disk drive. It works just like you would expect a second disk drive to work and has caused no trouble during the period of time that the staff here has been using it. It works as well as the internal disk drive of the PCjr. The memory expansion card in the *Legacy II* unit has also been working very well. Of course, there is still the problem of the PCjr architecture (lack of Direct Memory Access and the PCjr ID code in ROM) which may preclude some of the larger, standard IBM PC software (available from IBM and also some third-party vendors) from functioning on the PCjr. You would be wise to try any software package on your *Legacy*-equipped system before buying it.



**PHOTO 2:** Here is the *Legacy II* disk drive controller (which replaces the IBM disk drive controller) and the cable to connect the two drives to the controller.

Any PCjr software will still operate fine and apparently recognizes the added memory, except of course, programs written in Cartridge BASIC, because it only recognizes 64K of memory. We tried Microsoft's *Flight Simulator* and it recognizes the additional memory. Perhaps the most impressive piece of software that we have run on this expanded Junior system is Andrew Tobias' *Managing Your Money*, a financial data-base package that eats up a lot of memory and requires two disks. It will run on a single drive system, but it requires disk swapping at certain points in its operation. [Watch for a comparison of several of the PC software programs' performance with different expansion systems in a future issue.—Ed.]

The question then, comes down to this: Is the *Legacy II* unit worth its cost? Considering the recent sale price for the PCjr (around \$999 with 128K of memory, disk drive, and color monitor), the *Legacy II* price of \$795 seems to be too high. (Don't forget, the 256K memory



**PHOTO 3:** The pencil tip illustrates the point of interference between the power supply and the memory card during installation of the card (see text for details).



**PHOTO 4:** *Legacy II* with the memory card fully installed and operating with the PCjr system.

board is an additional \$395 . . . ) The \$795 price point would certainly seem more palatable if the extra memory card were included.

But once you get over the phobia of inserting the memory card and having to discard the IBM disk controller card, the *Legacy II* system is as reliable as if it were built by Big Blue itself. That's right folks, I like this product—I too am easily pleased by pretty blinking lights.

HCM

**Thinking of Subscribing?**  
Remember these time-worn truths:



"A watched pot never boils."

"Patience is a virtue."

"Good things come to those who wait."

"Allow 6-8 weeks for delivery of your first issue."

**HOME COMPUTER™**  
magazine



# The Factory

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

## HCM Review



Name:	The Factory
Program Type:	Educational
Machines:	Apple II Family
Distributor:	Sunburst Communications 39 Washington Ave Pleasantville, NY 10570
Price:	\$55
System Requirements:	48K, disk drive
	Poor Fair Good Excellent
Performance:	██████████
Engrossment:	██████████
Documentation:	██████████

*Learn how to operate machines, make your own products, even set up assembly lines in your very own factory.*

One of the best uses for home computers is in educating the young. The type of teaching that requires repetition and visual stimulation is ideally suited to these learning machines. Unfortunately, the number of quality educational packages produced by software manufacturers is limited. More are being written every day, but most of the early programs were half-hearted attempts to convert video games into educational games. While these types of games can work well with children, it is better to present challenges in a nonviolent and positive environment.

Sunburst Communications is marketing an educational game called *The Factory* which lets children actually create products and assembly lines on screen. Three jobs in the factory need to be done: The first is learning about the three machines and what they can do. The second job is building your own factory to create products on an assembly line. The third job is to test your new-found skill at making products by duplicating the same product that the computer makes.

### Learning Three New Jobs

As mentioned, job number one concerns learning about how the factory works and about the machines that are in it. You will learn how to operate 3 machines—the punch, the stripe machine, and the rotator. Each one performs several functions. The punch makes either square or circular holes in your raw material. It can make just one hole, or it can make up to three holes. The stripe machine paints stripes across the middle of your raw material. It paints in three widths: thin, medium, and thick. The rotator takes your raw material and turns it counter-clockwise 45, 90, 135, and 180 degrees.

Job number two teaches you how to take the machines and put them in an assembly line to manufacture your own products. You start out with a piece of raw material—a blank square. It's up to you to turn it into any product you wish, using any or all of the three machines; you can even use up to eight machines at once to make your product. The actual end product that you make is simply the same piece of raw material that has been punched, painted, or rotated (kind of like being folded, spindled, and mutilated, only it's not illegal)—but there are many different patterns that you can make. Once you decide on the sequence that the raw material is going to go through in the assembly line,

the factory will make it for you. You can watch as the machines punch, paint, and rotate the raw material, finally creating the pattern you designed. Once you get used to making different patterns, you can try to design them as complex as you can and have your friends try to copy them.

Job number three is the most challenging job in the factory. The computer will make a product and it's up to you to make one identical to it. There are three levels of difficulty: easy, medium, and hard. If you make the product and it doesn't come out just right (it's flawed), don't worry—you can try again until you get it right. Even if you get it right on the first try, you may want to try again because there may be more than one way to make the product.

### Problem-Solving Skills

*The Factory* teaches children how to look at a problem and break it down into steps in order to solve it. In duplicating one of the products that the computer has manufactured, the child has to try and figure out the order in which the pattern was made, or plan his or her own assembly line to recreate the product.

*The Factory* comes with excellent documentation in a simple, easy-to-read and understand manual. The only real complaint I have about this package is its lack of sound effects. I see a factory as a big, noisy building with rows of machines grinding and squealing as they crank out product after product. Granted there are small, clean, quiet factories out there, but that image is boring and fairly uninteresting.

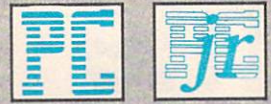
The sound effects in *The Factory* are feeble at best and could be improved. The graphics are simply drawn and are easy to see on the green Apple monitor, but I recommend using a color monitor if you have access to one.

With *The Factory*, Sunburst Communications has created a simple, yet challenging educational game for young children. The concept of a factory assembly line is novel and interesting, and is presented in a straightforward and easy-to-learn format. Young children—boys and girls alike—will find *The Factory* to be a challenging and helpful game that encourages creative thinking through the requisite preproduction planning. And it all takes place in an environment that is not destructive nor counterproductive to the emotional well-being of a child.

HCM

*"You can watch as the machines punch, paint, and rotate the raw material, finally creating the pattern you designed."*





# IBMPRESSIONS

## Create 3-D Surface Drawings With BASIC

by William K. Balthrop  
HCM Staff

At the heart of the IBM PC and PCjr is an extremely flexible graphics system. This, coupled with an extensive BASIC language such as BASICA or Cartridge BASIC, make writing graphics programs a breeze. The program listed here, *Ripples*, illustrates this point. A very minimal amount of BASIC code produced the spectacular effects that you see in the picture below.

### Drawing in 3-D

The ability to represent a three-dimensional shape on a two-dimensional screen is tricky at best. Many programmers have spent many arduous hours over their terminals trying to come up with better 3-D algorithms. One of the oldest problems encountered in drawing a three-dimensional object on the screen is: *What do you do with the hidden lines?* Because the computer screen is two-dimensional, you can't physically place a line or object behind another. You must somehow determine which lines or objects will be obscured by something in front of them. This is no easy task.

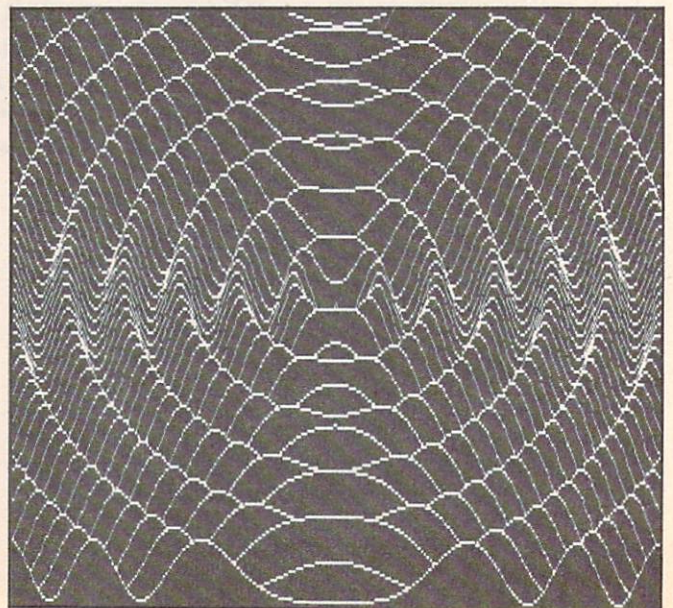
One method might involve drawing those objects that are farthest away first, and then drawing over them with closer objects. This method is probably the easiest to implement, but it has many drawbacks. For instance, the closer object must be solid in all parts, or the object behind it might show through.

The next best method would be to first draw those objects that are closest to you. You would then draw those objects that are farther away. As each pixel of an object is drawn, the screen would check to see whether the pixel already had been turned on by a previous object. If it had, then the object would discontinue drawing. This method, though cleaner than the previous, still has its drawbacks—like when you need to have the obscured object reappear on the other side of the object in front of it.

In the program *Ripples*, however, lies a fairly easy solution to the problem of the reappearing line. The only object used in this program is a line, so we don't need to worry about filling in solid objects. However, this presents a new problem. We can't simply check the screen for pixels that are already turned on, because the new line may traverse behind several lines before emerging again.

The solution here involved keeping track of the uppermost pixel painted on the screen for each pixel column. Because the object we are displaying is below the horizon, it will appear that we are looking down on it. This means that as you move higher up the screen, you move farther back on the object. Because of this, if we start drawing the object from the bottom up, any pixels plotted must never appear below the highest pixel previously plotted on that pixel column. This only works when you start drawing from the bottom of the screen first (the closest location to you).

The MAP() array is used to keep track of each pixel column. Every time the program prepares to plot a pixel, it first checks the array to see whether the new pixel





# Creating 3-D graphic illusions on a 2-D screen is tricky at best, but here is a program that does it with a clever "cover-up."

is above the previously drawn pixel in this column (the screen coordinate is less than the array value). If it is, then the new pixel is drawn, and the array is updated with the new position of the pixel. You can see this happen at the tail end of line 240.

## Ripples On The Screen

The algorithm for producing the ripple effect you see in the picture is really quite simple, and lends itself perfectly to the chosen method of drawing 3-D graphics. The program starts at the bottom of the screen, drawing each line from left to right until it works its way to the top. At each location along the way, it calculates the distance from that point to the center of the screen. With the center coordinates measured as 0,0 the distance to any point on the screen can be measured like this:

$$Z = \text{SQR}((X * X) + (Y * Y))$$

To find the exact location of the next point in the line we are drawing, we take the cosine of the distance and multiply it by an offset to give us an adjustment from the current vertical position:

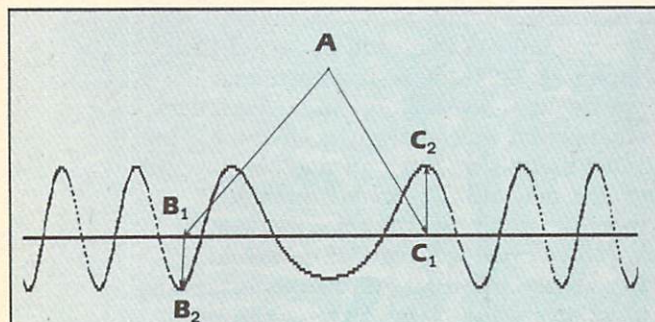
$$ZA = \text{COS}(Z) * 10$$

With the current vertical position contained in the variable Y, we can calculate the vertical screen coordinate with the equation below. The variable Y will have a value ranging from -11 to 12, with 0 at the center of the screen. This equation calculates the vertical screen coordinate:

$$YP = 191 - (Y + 12) * 8 + ZA$$

This new point is then compared to the MAP() array as described above to see whether it should be plotted. The variable XP keeps track of the horizontal column, while YP marks the vertical position:

IF YP < MAP(XP) THEN PSET (XP,YP):MAP(XP) = YP



The above chart illustrates the algorithm used in drawing the screen. A line is plotted from left to right. At each point along the line the distance from the line to A is determined. The cosine of that value is used to determine an offset from that line. The distance from A to B<sub>1</sub> produced a positive offset which placed the point to be plotted at B<sub>2</sub>. A to C<sub>1</sub> produced a negative offset with a point plotted a C<sub>2</sub>.

One quick word about the speed of this program. Because of the large number of calculations being done, and the number of points to be plotted, you have time to get a cup of coffee while waiting for the display to be completed. The result, though, is well worth the wait.

## Add Color with the PCjr

If you like the graphics created with *Ripples*, you're going to love an enhanced version for the PCjr. The PCjr has a screen mode that the PC does not. Screen mode 6 will allow you to draw high resolution graphics (620 x 200), and will also give you three colors to use, plus the ability to change the background color.

A few PCjr owners may encounter one minor hitch: You will need to have the 64K memory expansion card installed, giving your system a total of 128K memory.

(Most owners had this card installed when they bought their computer.) If you are unsure of how much memory you have, watch the screen when you first power up your system. Numbers will appear in the lower right hand corner of the screen. If they increase up to 128, then you have 128K of memory. If the numbers go above 128, don't worry, it just means you have more than enough memory.

Change line 230 to read as follows:

230 CLS: CLEAR ..., 32768: SCREEN 6: COLOR 2,1: PALETTE 2,15

Next, you will need to change the PSET command at the end of line 240 to read as follows:

PSET (XP,YP),2

Now have some fun creating your own 3-dimensional graphics on your IBM home system.

HCM

```

100  '*****
110  ' * R I P P L E S *
120  '*****
130  ' COPYRIGHT 1985
140  ' EMERALD VALLEY PUBLISHING CO.
150  ' BY WILLIAM K. BALTHROP
160  ' HOME COMPUTER MAGAZINE
170  ' VERSION 5.2.1
180  ' IBM PCjr: W/CARTRIDGE BASIC or
190  ' IBM PC W/BASICA and
200  ' COLOR/GRAPHICS ADAPTER and
210  ' COLOR MONITOR
220
230  CLS: SCREEN 2
240  KEY OFF: DIM MAP(639): FOR Z=0 TO 639
      :MAP(Z)=199:NEXT: PQ=0: FOR Y=-11 TO
12 STEP .5: FOR X=-12 TO 12 STEP .04
      :Z=SQR((Y*3)^2+(X*3)^2): ZA=COS(Z)*1
0: YP=191-(Y+12)*8+ZA: XP=(X+12)*25: I
F YP<MAP(XP) THEN PSET (XP,YP):MAP(X
XP)=YP
250  NEXT: NEXT
260  GOTO 260

```



# INDUSTRY WATCH

## **SPECIALIZED PRODUCTS CONTINUE TO FLOURISH FOR ORPHANED 99/4A**

Even though production of the TI-99/4A computer has long been discontinued, product development for it continues to flourish. A new CP/M-compatible card for the 99/4A's peripheral expansion box (PEB) could itself function as a stand-alone computer, says a representative of Foundation Computing, its creator. The card includes 64K RAM, a proprietary operating system, 2 RS-232 ports, and a built-in Western Digital disk controller that operates with double-density drives. The \$350 card's CPU is a Zilog Z80A that operates at 4 megahertz. It can be operated with Foundation's soon-to-be-released 80-column card. Speaking of cards, Myarc Inc. is expected to introduce a flexible expansion memory card, also for the TI-99/4A PEB. It will come with as little as 32K RAM, with additional chips available to upgrade it to 128K. In addition, Myarc is releasing a chip that will provide owners of Myarc's disk controller card with a resident disk directory available for use at any time. It takes up 1K of the card's memory and can be used without losing the contents of the computer's memory.

## **APPLE PRODUCTS SHOW EFFORT AT COEXISTENCE WITH IBM**

In its continuing effort to wedge its way into the business environment, Apple Computer Inc. recently introduced its new product line (the Macintosh Office), reduced prices, and renamed the Lisa as expected. Making their first official appearances at Apple's annual stockholders' meeting were Laserwriter, Apple's \$7,000 laser printer, and AppleTalk Personal Network, Apple's local area network that consists of a cable kit, software for the Mac and for MS-DOS, and a card for the IBM PC, PC XT, and PC AT—all at \$50 per unit. AppleTalk will support 32 devices. (In addition, a third-party company, Dayna Communications, has announced a \$995 MacCharlie, a coprocessor that allows Mac users to run IBM PC software.) Price reductions included the 512K Fat Mac, down to \$2,795 from \$3,195; the 512K memory expansion kit, cut \$300; and the Lisa 2/10, now renamed the Macintosh XL, down to \$3,995 from \$5,495. What wasn't expected, at least to outsiders, was the departure of Apple Co-Founder Steve Wozniak, who reportedly disagrees bitterly with the company's direction and is leaving to develop home-video equipment (and maybe even other computers . . .). He will stay on Apple's payroll as an engineering consultant. With the departure of Wozniak and at least 2 other engineers, development of Apple's IIx—its next generation II Family machine—may be suspended.

## **UNLIKE PREDICTIONS, MSX INTRODUCTION A NONEVENT**

The much-touted MSX world finally made a presence at the Winter Consumer Electronics Show, but it was one of yawning excitement rather than something of real impact. The arrival here of MSX computers—which all run Microsoft Extended BASIC and are supported by the biggest names in Japanese electronics—has been long awaited, but the wait may have been too long. Dealers, distributors, and competitors gave the technically-similar machines a less-than-warm welcome, saying that the 8-bit machines are now technically outdated, and that there is no room for another low-end machine standard such as the MSX. However, the new MSX computers include displays of 256 colors instead of 16, disk drives, and computer graphics that can be combined with images stored on a videodisk. MSX's price will place it between low-enders Atari and Commodore and high-enders Apple and IBM, and its biggest selling point may be its unique ability to interface to video systems, laser disks, and other home appliances originating from Japan Inc. Kazuhiko "Kay" Nishi, who devised the MSX standard, promised that the computers would be in the U.S. in time for Christmas 1985, even though the 12 MSX manufacturers at CES stated that they were in Las Vegas only to show, and not sell.



## **ATARI & COMMODORE DECIDE: OUT WITH THE OLD, IN WITH THE NEW**

Swelling inventories due to sluggish Christmas sales has prompted Commodore International Ltd. officials to cut both its production workers and its prices. Commodore has "furloughed" approximately 540 workers at its chip-assembly and computer plants, and cut the price of its Commodore 64 computer and disk drive units about 25 percent. Thousands of the systems are being sold at unheard-of low prices through a national liquidator, which said that Commodore is phasing out the C-64. Competitor Atari Corp. immediately followed by reducing the price of its 800 XL to \$99, which, some analysts speculate, would mean that Atari is selling the machine for less than it costs to make it. Both companies are trying to clear their decks of these old machines in favor of obtaining cash for their new models, the Commodore 128 and the Atari ST. Interestingly, both of these rivals have finally seen the need to implement upward software compatibility, but neither firms' machines have built-in disk drives, nor is there mention of intended cassette software support. Although there are only price rumors available, it seems that buyers will be expected to spring for the optional disk drives and monitors at the time of machine purchase, pushing the price of the typically low-end system up into the \$500 to \$1000 range.

## **WHAT NEXT FOR THE PCjr?**

Now that the Christmas buying rush is long past, the big question on the minds of PCjr dealers is whether/when IBM will renew its promotional pricing for the machine. Reducing its tag to \$999 (almost a 60-percent reduction from its price a year ago) while throwing in a color monitor and software, jolted it out of ho-hum sales into a starring role this past Christmas. Many retailers have insisted that the \$999 price set a precedent for the machine, and to discontinue the promotions would once again have them singing the "blues." Even though IBM has not yet announced any new marketing schemes for Junior, it is reportedly producing PCjrs at the same rate as it did in its successful fourth quarter.

## **THINGS TO WATCH FOR THIS YEAR:**

Turbo Pascal for the TI-99/4A . . . A Kodak personal computer . . . Hitachi's and IBM's CD ROM compact disk drives, which store up to 300,000 text pages . . . A 40-megabyte optical drive for IBM's portable . . . IBM's Personal Computer IX, similar to the PCjr but with 3-1/2 inch floppy-disk drives. It may in fact turn out to be the long-awaited PC2 . . . Rumored for the more distant future are a 3-1/2 inch, hard-disk-based Atari computer for less than \$1,000, and a Mitsubishi-built Atari PC-compatible . . . Things Not To Watch For: a Cabbage Patch computer from Coleco—the company which gave us the Adam, another famous orphan.

## **GEM: DIAMOND OR RHINESTONE?**

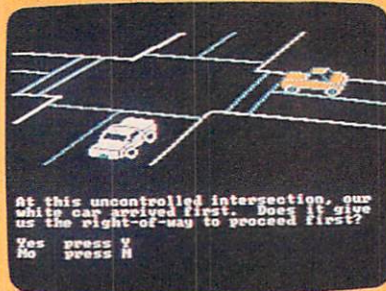
It's been compared to Apple's Macintosh. It will run on the IBM PC and has been described as a better alternative than IBM's Top View windows program. It is going to be bundled with Atari's new ST computers, and it has received enthusiastic response from industry critics. Hot out of the starting blocks before it has even been released for public consumption is GEM, Digital Research Inc.'s Graphics Environment Manager. It functions as a transparent layer between the operating system and an application, using a mouse to point to icons representing common computing functions much like the Macintosh does. In addition to the bundling endorsement from Atari and Britain's Applied Computer Technology, both Texas Instruments (for its Pro line) and Tandy/Radio Shack are expected to follow suit. So far at least 7 major software publishers are adapting old and developing new programs to run with the window manager. The GEM Desktop application will reportedly be available in late spring as a separate retail product of about \$50.



# Keys to Responsible Driving

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

## HCM Review



Name:	Keys to Responsible Driving
Program Type:	Educational
Machines:	Apple II Family, IBM PC, PCjr
Distributor:	CBS Software One Fawcett Place Greenwich, CT 06836
Price:	\$79.95
System Requirements:	Apple IIe, IIc: 48k, disk drive. IBM PC: 128K. PCjr: 128K, BASIC cartridge.
Performance:	Poor Fair Good Excellent
Apple	_____
IBM, IBM PCjr	_____
Cost/Benefit:	_____
Documentation:	_____

*Take the simple test, and find out whether or not you are really as safe a driver as you think . . .*

**K**ey to Responsible Driving by CBS software is a computerized study guide designed to help all drivers, new and experienced, become safer and more responsible when at the wheel of an automobile—a worthwhile goal if ever there was one. In the past few years, a new attitude toward driver responsibility has been developing, especially as it relates to drunk driving or driving while impaired. Driving is no longer just a faster way to get from point A to point B.

From the main menu you can elect to take a quick test to give you an idea of your general knowledge about driving safety and the rules of the road. This was the first thing I did, and I found that I didn't know as much as I thought I did. With that sobering fact in mind, I began to check out the tutorial.

### Questions & Answers

There are nine sections in this program, covering everything from drunk driving to warning signs and regulations. Each section is presented in a combination study-guide/question-and-answer format. As you work through a chapter, you are asked questions, presented with possible situations that could arise during driving, and shown correct methods of operating motor vehicles. A typical section consists of several screens of text explaining the proper procedures for handling various driving situations. These are combined with short question and answer (true or false, and multiple choice) problems, along with graphic representations of the problem. For instance, a picture puts you behind a slow-moving truck, and you are asked to answer a question about the situation. Then you are given the correct answer, and an explanation as to why your answer is correct or incorrect.

The program's documentation has a very informative and even chilling section detailing the cost of unsafe driving. It is primarily composed of statistics about highway fatalities and accidents, along with a very hard look at drunk driving and its consequences. (Of all drivers involved in fatal crashes, 36 percent were known to be intoxicated, and the true figure, including unproven cases, is probably quite higher.) The rest of the manual contains more safety tips, like driving in winter weather, and what to do at the scene of an accident—and it even has a set of sample questions taken from several states' written driving test.

Unfortunately, the documentation doesn't mention anything about one section of the menu—records. You can access a short explanation screen at the beginning

of the program, but it should be covered in the manual as well. In the records section, the program keeps track of your scores on the pretest and the post-test, along with the letter of the last completed chapter.

What little animation there is tends to be slow and uninteresting. If the program showed you, as in a video game, more of the consequences of your driving decisions, it would be a much better use of the computer medium to "drive a point home." As it is, Keys well prepares you for the multiple-choice license exam—but it may be no better than a textbook at preparing you

for the open road, and considering this product's high price, you may wonder if it is worth the added expense.

Keys to Responsible Driving is available for the IBM PC, PCjr and the Apple II Family of computers. Both versions are nearly identical in content, but I found myself getting bored

waiting for the IBM programs to draw the graphics, especially the PCjr version. The Apple versions were no problem here. This is surprising considering the basic speed and graphics capabilities of the PCjr. Perhaps both versions were written by programmers most familiar with the Apple machines.

### A Hard Look

I've always considered myself to be a safe and defensive driver, and, for the most part, this program confirmed that. But it also showed me that I could improve my skills in ways in which I wasn't aware. This was the program's main impact—awareness. Just the quick pretest showed me that I wasn't as sure of the rules of the road as I thought I was, and working my way through the first few sections indicated to me that even though I haven't had a ticket or been involved in an accident in years, doesn't mean that I can safely assume that I don't need to work at driving safely.

Once you have completed all of the chapters, the final check is to take the post-test to see whether you have learned anything. I noticed that I had improved since taking the pretest, but even more importantly, I found that as I worked through the chapters, there was a subtle improvement in my responses to each chapter's questions as well. All this leads me to believe that this is a worthwhile package—one that will help make safe drivers safer, and give other drivers a realistic gauge to assess their driving skills and find out where improvement is needed.

HCM



# HOME COMPUTER<sup>TM</sup>

## 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:

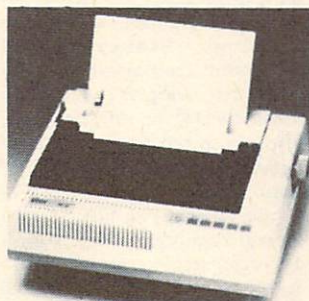
Product News Editor  
Home Computer Magazine  
1500 Valley River Drive., Suite 250  
Eugene, OR 97401



### The Printed Word Becomes Starstruck

New Printer Line Offers Flexibility

Star Micronics Inc. has introduced its new line of printers, the SG, SD, and SR series, each available in 10- or 15-inch versions. The 15-inch machines have a standard 16K buffer. All of the printers offer Near Letter Quality or Draft modes, friction and tractor standards, hex dump, and downloadable characters. In addition, the SR printers have automatic single-sheet feed, pause and feed buttons, and short form tear-off. The SG-10 (\$299) and SG-15 (\$499) print at 120 cps, the SD-10 (\$449) and SD-15 (\$599) print at 160 cps, and the SR-10 (\$649)



and SR-15 (\$799) print at 200 cps. The printer series combines the Star standard and PC printer lines into one line that is switch-selectable for the IBM PC, Apple II Family, and Commodore computers.

Star Micronics Inc.  
200 Park Ave.  
New York, NY 10166



### TI Users Get An Extension

Extended BASIC Available Again

Under a direct license from Texas Instruments Inc., Microsphere has begun reshipping their MicroPal Extended BASIC cartridges for the TI-99/4A. Microsphere guarantees that MicroPal is

100-percent compatible with all commercial and user-written programs requiring the original TI Extended BASIC. It carries a suggested retail price of \$89.95.

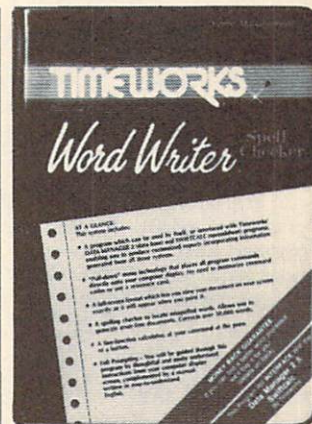
Microsphere, Inc.  
14009 E. Jefferson Blvd.  
Mishawaka, IN 46545  
1-800-348-2778



### Integration Is The Key Word

Double Package for PC, PCjr

The Word Writer word processing program and Data Manager 2 data base from Timeworks, Inc. are now available for the IBM PC and PCjr. They join Timeworks' SwiftCalc spreadsheet, making up an integrated, three-program set. Word Writer includes a spell checker of more than 40,000 words, as well as a built-in calculator. Along with normal editing features, it provides for document chaining, page separations, horizontal and vertical scrolling, and form-letter printout options. Data Manager 2 is a general information-storage and retrieval system with report-writing, graphics, and label-making capabilities. It contains Timeworks'



X-Search, X-Sort, and X-Chart features. For a limited time, the IBM versions of Word Writer and Data Manager 2 can both be purchased for \$129.95 total.

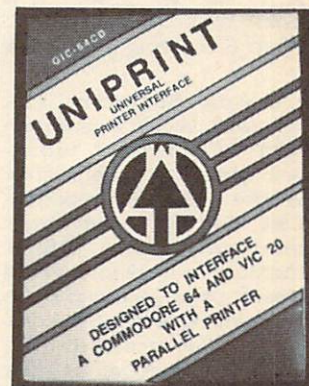
Timeworks  
444 Lake Cook Rd.  
Deerfield, IL 60015  
1-800-323-9755



### The Commodore Connection

An Easy-to-Use Printer Interface

Uniprint, by Giga International Corp., interfaces the Commodore 64 and VIC-20 with any parallel printer, including daisy wheels. It has no dip switches, and no chip changes are required. Uniprint converts Commodore ASCII into Standard ASCII through its Translate mode, which accesses special printer features like underlining, boldface, and italics when set in Transparent mode. It permits most printers to emulate the Commodore VIC 1525 Printer and allows dot-matrix printers with programmable graphic capabilities to



print all the letters and graphic characters found in Commodore ASCII. The Uniprint interface costs \$99.

Giga International Corp.  
312A Auburn St.  
San Rafael, CA 94901  
(415) 258-0901





# HOME COMPUTER

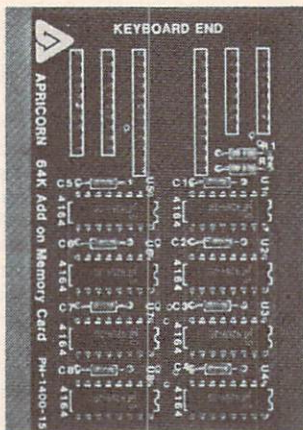
product news

## Peripheral Power

*New Add-Ons for Apple*

Two new peripherals for Apple II Family computers have been released by Apricorn: Extend-It, and Super Serial Imager. Extend-It is a 64K memory module that doubles system memory to 128K bytes for owners of Apple IIe computers equipped with Apple's 80 Column Text Card. Super Serial Imager can transfer high-resolution graphics from screen to printer, and it will also support the new 300/1200 baud intelligent modems.

Apricorn  
7050 Convoy Court  
San Diego, CA 92111  
(619) 569-9483



## Great Graphics With Zoom

*Graphics Design Tool for the 99/4A*

Graphx is a graphics design program for the TI-99/4A that can be used alone or as a tool with assembly language programs. Users can try free-hand drawing, zooming in on or moving sections of pic-

tures, repainting parts of pictures to new colors, a typewriter mode to add text, line and circle creation, and animation, among other things. Graphx requires 32K, a disk drive, and a joystick, and it costs \$50.

Graphx  
P.O. Box C568  
Sydney, NSW 2000 Australia

## Only The Final Result Counts

*Word Processing & Spreadsheet Results*

Handic Software has released Word Result, an IBM PC or PCjr word-processing program that works with the company's Calc Result spreadsheet program when 256K memory is available. Word Result uses verb and noun combinations for the commands. Full-screen formatting displays what the printed document will look like, including headers, page numbers, margins, indents, and footnotes. Other features include mail merge, discretionary word division, abbreviation registers, and



"total printer compatibility." Word Result and Calc Result are \$195 each, or \$345 for the set when purchased together.

Handic Software Inc.  
520 Fellowship Rd. Suite B206  
Mount Laurel, NJ 08054  
(609) 866-1001

## Update Of A Classic

*It's Super Zaxxon!*

Human Engineered Software has introduced Super Zaxxon for the Commodore 64, with fast action, more ground targets, and advanced three-dimensional graphics. A tunnel

where planes zoom in to attack and flying saucers drop bombs on from above has been added, as have floating fortresses and a fireball-spitting dragon. Super Zaxxon retails for \$29.95.

Human Engineered Software  
150 North Hill Dr.  
Brisbane, CA 94005  
(415) 468-4111

## Computer-Assisted Weight Reduction

*Ways to Keep Fit & Lose Weight*

The Original Boston Computer Diet, a new program by Scarborough Systems, Inc., emphasizes change—in eating habits, exercise routines, lifestyle, and behavior. Designed for people with weight-loss problems of 10 to 40 pounds, the program acts as a personal fitness and weight-loss counselor, using dialogue to analyze your nutritional requirements, monitor your eating and exercise habits, and suggest meal plans. The Original Boston Computer Diet is available for Apple II Family and IBM



PC/PCjr systems for \$79.95, and for the Commodore 64 for \$49.95.

Scarborough Systems, Inc.  
55 S. Broadway  
Tarrytown, NY 10591  
1-800-882-8222

## Role-Playing On Apples

*Magic Abounds in Fantasy Lands*

Two fantasy role-playing games for the Apple II Family, IBM PC/PCjr, and Commodore 64 will be released this Spring by Origin Systems, Inc. Ultima IV—Quest of the Avatar fully occupies both sides of two disks and allows the player to converse with characters in the game on hundreds of topics. Opportunities for an "infinite variety of combat situations in a multitude of terrains and scenarios" abound

in the land of Brittania, as does "a unified system of magic." In Moebius I—The Orb of Celestial Harmony, players must recover the stolen Orb before it disrupts the universe and destroys the island kingdom of Khan-tun. Travels take players through the elemental planes of Earth, Water, Air, and Fire and their hostile residents, with a sword and knowledge of martial arts the only defenses. Both games retail for \$59.95.

Origin Systems, Inc.  
1545 Osgood St. #7  
North Andover, MA 01845  
(617) 681-0609



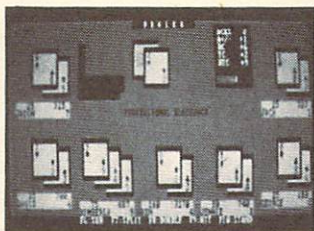
# HOME COMPUTER<sup>TM</sup>

## product news

### Turn From Luck To Skill

*A Guide to Casino Gaming*

Caesars World Productions, Inc. and Screenplay, Inc. have released Black-jack, the first game in their series "Caesar's Guide to Gaming." The series is designed to help players improve their skills through the simulation of casino games as played at Caesars Palace in Las Vegas, Nevada; Caesars Tahoe at Stateline, Nevada; and Caesars Atlantic City in New Jersey. Other games to be released later include Roulette, Craps, Poker,



Baccarat, and Slots. They will all be available for the Commodore 64, IBM PC and PCjr, and the Apple II Family computers. The retail price of Black-jack is \$69.95.

Screenplay, Inc.  
1095 Airport Rd.  
Minden, NV 89423  
(702) 782-9731



### Let The Computer Deal With The IRS

*A New Tax Calculation Program*

Texas Taxes is a tax calculation program for the TI-99/4A that includes forms 1040 and 2441, schedules A, B, G, and W, and the tax tables (including the state sales tax for your state). The 16K program

requires only a cassette player or a disk drive, and it is available for \$10.95 plus \$2 postage and handling. If you send it back every year with \$5, you will receive an updated version that includes all form changes.

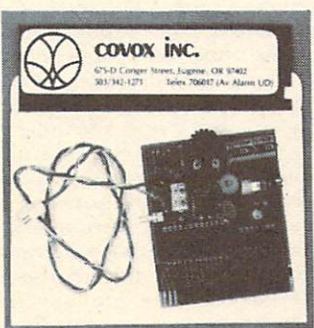
Steven Karasek  
855 Diversey Dr.  
St. Louis, Mo. 63126



### Let The Apple Sound Off

*Music & Sound Capabilities Arrive*

Polyphonic music, sound effects, and speech response for the Apple II+ and IIe are now possible with the Sound Master plug-in printed circuit board from Covox, Inc. It can produce 3 independently programmable tones and other noises, and a demonstration disk includes a numbers vocabulary for creating talking calculators and games. An on-board power amplifier directly drives either the internal or an external loudspeaker. Also, prerecorded synthetic or digitized speech may be



reproduced. The Sound Master is \$39.95, which includes the board, a 32-page manual, and the demo disk.

Covox, Inc.  
675-D Conger St.  
Eugene, OR 97402  
(503) 342-1271



### Commodore Rolls Out New Systems

*A PC, An LCD, and Peripherals*

Commodore International Ltd. has introduced its new Commodore 128 Personal Computer, its Commodore LCD, and a line of peripherals which include a disk drive, monochrome and color monitors, modems, and a mouse. The Commodore 128 features 128K RAM expandable to 512K, 40/80-column full-color display, compatibility with all Commodore 64 peripherals and software, a 92-key keyboard including a numeric keypad, and the ability to run CP/M programs.

The LCD (liquid crystal display) is a three-pound, briefcase-size computer with built-in Commodore 3.6 BASIC, 32K RAM, and an 80-column-by-16-line screen. It can run on batteries or external power supplies, and word processing, file management, spreadsheet, communications, and other software are built-in.

The Commodore 1571 5-1/4 inch floppy disk drive offers 1/2 megabyte, 360K formatted storage, fast data transfers, and two operating modes—Com-



modore 64, which operates at about 300 cps, and Commodore 128 mode, which operates at about 2000 cps. The Commodore 1660 Modem 300 and 1670 Modem 1200 feature auto-answer/auto-dial, and a built-in speaker. The Commodore Mouse complements the Commodore 128 Personal Computer, and the Commodore 1901 Monochrome Monitor and 1902 RGBI/Composite Monitor both support 80- and 40-column displays.

Commodore International Ltd.  
1200 Wilson Dr.  
West Chester, PA 19380  
(215) 431-9100



### Getting It Disassembled

*Serious Programmer's Tools Available*

StarSoft has introduced their Disassembler and Disk Editor programs for the TI-99/4A. Both are \$19.95. The Disassembler can disassemble in text, data, or mnemonic instruction formats, and output to the screen, printer, or drive. It may also be used to disassemble system ROM routines as well as programs loaded from disk. It requires 32K, a disk drive, and the Editor/Assembler

cartridge. The Disk Editor allows the user to edit a disk by individual sectors and bytes rather than by file names. A full-screen editor displays the contents of a sector on the screen and allows editing in both hexadecimal and ASCII character formats. It requires 32K, a disk drive, and either an Extended BASIC, Mini Memory, or Editor/Assembler cartridge.

StarSoft  
601 Alleghany St.  
Blacksburg, VA 24060  
(703) 953-1490





# HOME COMPUTER<sup>TM</sup>

product news

## Stay-At-Home Shopping

*Obtain Mail Access Via Computer*

The Electronic Mall, a videotex shopping service for modem-equipped personal computer users, will be available later this year to subscribers of CompuServe Information Service. The Mall offers thousands of products and services from such outlets as retail stores, airlines,

travel services, insurance and record companies, and publishers, often at a discount to subscribers. Complete descriptions are provided on all products, and a feedback capability allows shop-at-home users to ask the mall manager specific questions. It is "open" 24 hours a day, every day.

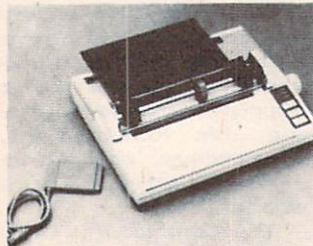
CompuServe  
P.O. Box 20212  
Columbus, OH 43220  
(614) 457-8600



## Write A Letter Home

*New Printer Connects to Many Systems*

HomeWriter 10 is a new 80-column dot-matrix printer by Epson America, Inc. that was created specifically for the home market. Its new plug-in Printer Interface Cartridges (PIC) connect the printer to most brands of personal computers, including the Commodore 64, IBM PCjr, and Apple IIc. The HomeWriter 10 operates at 100 cps in Draft mode and at 16 cps in Near Letter Quality mode. Users can choose combinations of print styles such as condensed, emphasized, or



doublestrike. The printer will be available in March for \$269, and \$60 for each PIC. An optional tractor feed (\$39.95) and cut sheet feeder (\$99.95) will also be available.

Epson America Inc.  
2780 Lomita Blvd.  
Torrance, CA 90505  
(213) 539-9140



## From Pints to Quarts to Gallons to . . .

*Unit Conversions Computed*

Tanoak Software Inc. has introduced Tanoak Conversions, a software package that performs unit conversions of more than 50 engineering, scientific, and general categories. Up to 36 units are available in each of these categories, which are

indexed in alphabetical order. The menu-driven program accepts both decimal and exponential notation for both input and output formats. The program runs on the IBM PC and compatibles with 128K RAM, and is priced at \$49.

Tanoak Software Inc.  
878 Brookline Dr.  
Sunnyvale, CA 94087  
(408) 738-8339

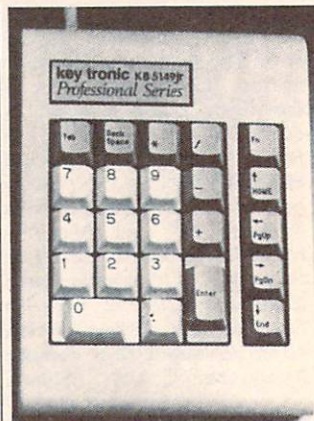


## First Letters, Now Numbers

*PCjr Gets Numeric Keypad*

Key Tronic has released its KB 5149jr, a numeric data entry pad that attaches to the PCjr. It includes keys for numeric data entry, cursor control, mathematical functions, tabs, and backspacing, and it can convert to function mode without returning to the PCjr keyboard. Its suggested retail price is \$99.95.

Key Tronic  
P.O. Box 14687  
Spokane, WA 99214  
1-800-262-6006



## A Ticklish Matter

*2 Printer Utilities for TI*

Gembar Graphics has released two printer utilities for the TI-99/4A. The Epson RX-80 Tickler and the Gemini 10X/15X Tickler employ menus to help the user set font styles, print pitches, line and form feeds,

margins, tabs, bold, underlines, subscripting, and other options. Available on diskettes only, the packages require Extended BASIC and expanded memory, and cost \$11.95 each.

Gembar Graphics  
455 Amherst Circle East  
Satellite Beach, FL 32937



## Computer Clubs Competition

*Apple to Award Prizes Again*

The Second Annual National Competition for Apple Student Computer Clubs is now under way. Sponsored by Apple Computer, Inc., students and advisors at elementary and secondary schools across the country will again compete for Apple hardware and software awards. Clubs can submit either a computer programming project that demonstrates their collective technical skills, or a project that uses personal computers to help members of the community. Club advisors can enter either an

advisor project that documents an effort to successfully build a student computer club, or a lesson plan project that demonstrates how advisors and teachers can work together to integrate personal computers into the regular educational curriculum. A total of 88 clubs and 48 advisors will be chosen as semifinalists. All U.S. clubs that are members of the Apple Computer Clubs program are eligible. Entries must be postmarked by April 15, 1985. For more information or to enter, write:

Apple Computer Clubs  
P.O. Box 948  
217 Jackson St.  
Lowell, MA 01853





## Animation By Touch

*A Tablet for Moving Graphics*

A printer dump and a built-in character set for text characterize Animation Station, a new graphics tablet by Suncom. It features side-mounted left- and right-hand function buttons, as well as a joystick emulator feature. Included with the package is a color software program called DesignLab which allows the user to cut and paste picture elements. Additional programs for creating animated movies or selecting shapes from an electronic clip art book for educational, business, or design purposes are available separately.



The touch-sensitive tablet and cursor controller is available for the Apple systems for \$89.95 and for Commodore systems for \$79.95.

Suncom  
260 Holbrook Dr.  
Wheeling, IL 60090  
(312) 459-8000



## Make It A Threesome

*Productivity for Kids & Adults*

Softsync, Inc. has released two new programs: Trio, an integrated word processor, spreadsheet, and data base for adults, and Kid Pro Quo, a word processor for kids. Trio Word can link files, imbed print commands, search and replace, and edit. Trio Calc uses a natural order of recalculation, and can vary column widths; copy formulas, columns, or rows; split the screen; and format cells. Trio File allows 17 fields per record, and all three applications offer help screens. Trio runs on the

Commodore 64 for \$69.95, and on the IBM PC/PCjr and Apple IIe/IIc for \$99.95.

Designed especially for kids ages 8 to 14, Kid Pro Quo enables kids to write text, draw and animate pictures, or compose music using just one program. The software includes a built-in picture library and music library that kids can manipulate or put into their own creations. Kid Pro Quo retails for \$29.95 for the Commodore 64, and for \$39.95 for the Apple IIe/IIc and IBM PC/PCjr.

Softsync, Inc.  
162 Madison Ave.  
New York, NY 10016  
(212) 685-2080



## The Numbers Game

*Pick Lottery & Football Winners*

Lotto Picker is the newest program for the TI-99/4A from Ridge Services. The program generates a series of random plays for all Lotto-type games using the same process of many lottery commissions. Lotto Picker has 16 of the most popular Lotto games in North America preprogrammed into it, but users can also substitute another. It is written in console BASIC, so all that is required is a cassette player or disk drive. Lotto Picker's

suggested retail price is \$25. Ridge Services also offers Pro Football Analyst (\$34.95), which helps choose NFL and USFL winners against the point spread, and Personal Inventory Program (\$20), which keeps details on personal possessions for insurance, police, and IRS purposes. Both programs are available on diskettes or cassettes, but the Personal Inventory Program also requires Extended BASIC.

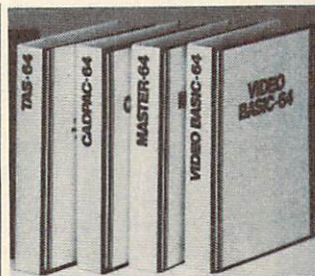
Ridge Services  
170 Broadway, Suite 201  
New York, NY 10038  
(718) 833-6335



## Expand Your Programming Vocabulary

*Various Languages Available for C-64*

Five new language programs and four productivity programs have been added to Abacus Software's product line for the Commodore 64. The Ada Training Course includes an editor, syntax checker/compiler, assembler, disassembler, and training guide for this new language. BASIC Compiler compiles to either fast-running 6510 machine code and/or compact speed-code. C Compiler conforms to the Kernighan and Ritchie standard, but without bit fields. It includes an editor, compiler, and linker. Fortran Compiler is a subset compiler based on the Fortran 77 standard, and it also includes an editor, compiler, and linker. Video BASIC is



for developers of graphics and sound software, adding 50 commands to standard BASIC. Cadpak-64 is a lightpen-based design package. Chartpak-64 is a menu-assisted pie, bar, and line-charting package. Datamat is a data management program, and Power Plan-64 is a spreadsheet with built-in graphics.

Abacus Software  
P.O. Box 7211  
Grand Rapids, MI 49510  
(616) 241-5510







## A Selective Input Routine

```

100 REM *****
110 REM * INPUT FILENAME *
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 5.2.1
180 REM APPLE II FAMILY APPLESOFT
190 REM ACCEPTS ONLY LEGAL PRODOS FILE
    NAME
200 HOME : TEXT : VTAB 5: PRINT "INPUT
    FILENAME:"
210 LTS = CHR$(8):CR$ = CHR$(13)
220 MX = 15:TS = "":VT = 7:HT = 1
230 VTAB VT:HTAB HT:PRINT TS:SPC(MX
    - LEN(TS)):HTAB LEN(TS)+1
240 IF LEN(TS) = MX THEN HTAB MX
250 IF TS = " " THEN GOSUB 320:GOTO 27
    0
260 GOSUB 340
270 IF K$ = CR$ THEN 360
280 IF K$ = LTS AND LEN(TS) < 2 THEN
    TS = " ":GOTO 230
290 IF K$ = LTS THEN TS = LEFT$(TS, L
    EN(TS)-1):GOTO 230
300 IF LEN(TS) < MX THEN TS = TS + K$
    :GOTO 230
310 TS = LEFT$(TS,MX-1) + K$:GOTO
    230
320 GET K$:IF NOT (K$ = LTS OR K$ = C
    R$ OR (K$ > "A" AND K$ < "Z"))
    THEN PRINT CHR$(7):GOTO 320
330 RETURN
340 GET K$:IF NOT (K$ = LTS OR K$ = C
    R$ OR (K$ > "A" AND K$ < "Z")
    OR (K$ > "0" AND K$ < "9"))
    THEN PRINT CHR$(7):GOTO 340
350 RETURN
360 STOP:REM LINK TO DISK ACCESS ROU
    TINE
    
```

Getting keyboard input is essential to a BASIC program. If the program uses the **INPUT** statement, any characters can be used (including control characters), and the length of an **INPUT** statement can be up to 239 characters long. But if the the input is to be limited to a specific group of characters, the **INPUT** statement is not adequate. When getting a disk file name, for example, precise input is critical. If you use just the **INPUT** statement to discern a file name, the user could easily type in an illegal name and cause a disk error to occur.

If only one or two characters is being sought, the **GET** command is easy to use; however, if the input is to include a number of characters, some logic is necessary to filter the input.

The accompanying program filters keyboard input so that only a legal ProDOS file name can be input. Although this program seems long just to get one line of input, a little extra code makes a "user-friendly" program fool-proof and easy to use.

Lines 210 and 220 set up a few variables used by the main body of the program. **LTS** is set equal to ASCII character 8, which is the [left arrow] key on the Apple IIe or IIc, and [Control] H on the Apple II or II+. The program recognizes this as a backspace key and accepts no other control characters (except a carriage return, which is set equal to the **CR\$** variable). The main section could be used as a subroutine where **MX** would be set to the maximum length of the input, **TS** would be the string variable where the input will be placed, and **VT** the row and **HT** the column where the first character of the input is to be printed on the screen.

Lines 230-350 are where the actual input is processed. Line 230 is the main entry point of the routine. Line 240 is an "end-case" check to place the cursor on top of the last character if the input has already reached maximum length. Because ProDOS file names must begin with a letter, line 320 is used if **TS** is null, to limit the input to letters only. For any other characters, line 340 filters the input. By altering these lines, this routine could be used as an all-purpose input routine, to keep your programs user-friendly.

—Roger Wood





## Using Relative Access Files

Large sequential data files can take minutes to load and save using the VIC-1541 disk drive. To speed up file access, a knowledge of relative-access files can be invaluable to a programmer. The Commodore Disk Operating System (DOS) allows for a maximum of 254 characters in a record, and a maximum of 720 records in one file. The key to faster relative file access is the use of side sectors. A side sector is a sequential file that keeps track of the exact location of a record (disk track and sector) so DOS can zero in on any record in a minimum amount of time. The DOS takes care of these details, so how side sectors work is not important here.

One important limitation of relative files is that only ASCII characters can be written to the file. Integer and real number quantities must be converted to strings before they are saved to disk with the **PRINT#** command. Therefore, numeric data being stored only as strings must be read into only string variables, using either the **INPUT#** or **GET#** statement, and then converted (via the **VAL** statement) back into numeric quantities.

The format for creating a relative access file is as follows:

**OPEN** logical file#,device#,channel#,"filename.L"+**CHR\$(record length)**

The device# on a standard VIC-1541 disk drive is 8. The logical file# is, as with other types of files, used throughout a program to refer to the file for loading or saving any data. Although the logical file# can be any number from 1 through 255, it is usually best to use the same number as the channel#. The channel# for disk access can be any number from 2 through 14. (0 and 1 are reserved for cassette access.) If you have a Commodore compatible printer it will usually use channel 4, so it is best to use 2, 3, or 5 through 14. Here's an example of an **OPEN** statement for a file called MAILLIST which has a logical file# of 2, and a record length of 127 characters: **OPEN 2,8,2,"MAILLIST.L"+CHR\$(127)**

Notice that the record length is expressed as a string value—not a numeric one. This is true of the parameters in all of the relative access file commands. A good way to set up the various **CHR\$** values is to set string variables equal to the values needed. For example, in the above file you can set **RL\$** equal to **CHR\$(127)**, so that any time you needed to write the Record Length to the file, you could use **RL\$**. This technique is especially valuable when using the **POSITION** ("P") command explained below.

Channel 15 (called the command channel) plays a critical role in Relative file access: it allows you to read or write data to an exact place in the file using the "P" command. First you must **OPEN** the command channel, and then you can position not only to a specific record in a file, but to a specific position within a record. Here's the basic format:

**OPEN 15,8,15**

**PRINT#15,"P"+CHR\$(channel#)+CHR\$(rec# low byte)+CHR\$(rec# hi byte)+CHR\$(position)**

To obtain the rec# low and hi bytes of the record number, use the one-line subroutine (here called line 10000), where the record number is passed in a variable called **RC**. When you **RETURN** from this line, **RC\$** contains the entire string value needed by the "P" command for the record number. So, if you wished to write the first character in the 20th record you could set **RC=20**, then execute the following:

line no. **GOSUB 10000:PRINT#15,"P"+FC\$+RC\$+CHR\$(1) . . . continue program**

**10000 HB=INT(RC/256):LB=RC-(256\*HB):RC\$=CHR\$(LB)+CHR\$(HB):RETURN**

The "P" command doesn't write anything—it just sets the pointer so that the next **PRINT#2**, **INPUT#2**, or **GET#2** command will access the data beginning at the first character of the 20th record. By using values other than 1 for position, the file pointer can be placed to any position in any record. DOS doesn't check to see whether the position is beyond the end of a record. The programmer must make sure that the data will fit in the record.

This "P" command does have a rather major bug in it, however. Only 254 bytes may be stored on a sector. If you choose a record length that does not divide evenly into 254 (i.e., 1, 2, 127, and 254), the "P" command will not function properly in some records. Commodore DOS only keeps track of where a record begins. If a record begins in one sector and ends in another, the "P" command fails to get to the second sector when attempting to locate the pointer directly to a character in the second sector. If you stick to 1, 2, 127, and 254 character record lengths, each record will fit in only one sector, and this bug presents no difficulties.

—Roger Wood





## REDEFINING AND USING IBM FUNCTION KEYS



```

100  * * * * *
110  * FUNCTION KEY REDEFINITION *
120  * * * * *
130  COPYRIGHT 1985
140  EMERALD VALLEY PUBLISHING CO.
150  BY WILLIAM K. BALTHROP
160  HOME COMPUTER MAGAZINE
170  VERSION 5.2.1
180  IBM PCjr W/CARTRIDGE BASIC & 128K
    MEMORY or
190  IBM PC W/BASICA and
200  COLOR/GRAPHICS ADAPTER and
210  COLOR MONITOR
220
230  CLS:KEY ON:WIDTH 80:K=0:DIM KS(5):F
    OR Z=1 TO 5:READ KS(Z):NEXT:GOSUB 3
    00
240  WDS=" "
250  BS=AS:AS=INKEY$:IF AS=" " AND BS=" "
    THEN 250 ELSE IF AS>" " THEN WDS=WDS
    +AS:GOTO 250
260  WDS=LEFT$(WDS,6):P=INSTR(KS(K),WDS)
    :IF (P-1)/5=INT(P-1)/5 THEN 270
270  IF WDS="NEXT" THEN GOSUB 300:GOTO
    240
280  IF WDS="EXIT" THEN 370
290  LOCATE 12,1:PRINT "THIS IS ";WDS:GO
    TO 240
300  K=K+1:IF K>5 THEN K=1
310  FOR Z=1 TO 10:KEY Z,MIDS(KS(K),(Z-
    1)*6+1,6):NEXT:RETURN
320  DATA EDIT COPY GRAB PASTE INSE
    R
330  DATA LOAD REDO EXIT NEXT
    TS
340  DATA BLACK BLUE GREEN CYAN RED
    MAGENTBROWN WHITE EXIT NEXT
    SQUARECIRCLERECT TRIAGLPOLYG
    NHXAGNFLIP ROTATEEXIT NEXT
350  DATA ZOOMINZOOMOT LEFTRIGHT GO
    STOP FASTERLOWEREXIT NEXT
360  DATA HIDE JUMP SWIM LOOK FIND
    SHOOT INVENTTALK EXIT NEXT
370  CLS:END
    
```

The function keys found on the IBM PC and PCjr have an almost unlimited number of uses. The only hard part is using them within BASIC programs.

One option is to use the automatic program branching available with **ON KEY(n) GOSUB**. Most of the time this method, though, proves to be more of a headache than it's worth because the program exits to the function-key routine—no matter where you are in the program (it's interrupt-driven).

The procedure outlined here will allow you to redefine the function keys and read them using the **INKEY\$** function. This not only accesses the function keys, but it offers **F0** as a link to even more keys. By using this method, you could scroll through an unlimited supply of functions. When **F0** is pressed, the next set of functions will be assigned and displayed at the bottom of the screen. **F9** is always used as the exit key, giving the user a graceful way out of a program.

The different functions for the function keys are contained in a string array, **KS()**. One element of the array is

used to contain information for each set of function keys. The array is set up in line 230 from the data statements in lines 320 through 360. These data statements contain the function key information. Because only 6 characters can be displayed on the screen for each function key, the size of each function redefinition will be limited to 6 characters (a maximum of 15 characters is possible with BASIC on the IBM).

The algorithm is simplified by making all definitions the same length. In line 260, a simple search is made through the **KS()** array for valid inputs using the **POS()** function. The input is shortened to 6 characters, and the position of those characters is found within the active function string array, **KS()**. The program then checks the position for an even boundary of 6 characters within the string array to ensure that the information read from the buffer makes a full match with one of the function definitions.

When **F10** is pressed, line 270 checks the input for the word "NEXT ". (Notice that the word "NEXT " is followed by two spaces, making it 6 characters in length.) This will cause a **GOSUB** to line 300. There, the pointer, **K**, into the function key array will be incremented. If **K** is larger than the largest element of the array, it will be reset back to 1. The next line pulls out 6 characters at a time from the new string of function information, and assigns it to the function keys. Line 280 looks for **F9** by checking for "EXIT " in the input buffer.

By using this method, you won't need to worry about an automatic branch to a function routine upsetting part of your program. In addition, you can now use function keys in place of menus. Each function key could take you to a different screen, or another menu. This makes the task of designing menu screens easy, and offers a central area for operators to find and select the next step in their process.

—William K. Balthrop



# TECH NOTES



## NUMERIC DATA COMPACTION

A major concern among TI-99/4A programmers is the machine's limited memory—only 16K-bytes. Expanding this to 48K is relatively expensive, and software which requires memory expansion sorely reduces the audience that could use it. The following routines may help eliminate some memory space problems.

The TI-99/4A is capable of containing only one type of number, referred to as a floating point. A floating point number can be extremely large or small, and can include scientific notation (such as 4.0036623E + 56). However, the cost in memory space to hold such numbers is great. No matter what value you assign to a number, it will always use up 8 bytes of memory. A further restriction is that the numbers must be positive, ranging from 0 to 65536. But, it is not always necessary to use floating point numbers—many applications work only with integer numbers. Integer numbers are numbers which have no fractional part (nothing to the right of the decimal point). The biggest advantage that the method shown below has over storing values conventionally is that it requires only 2 bytes of memory for each value, as opposed to 8 bytes with normal variables. This is a 400% savings in memory.

The most efficient use of these routines occurs when you need to contain a large number of values in memory, and you can't do it with conventional variables. (Although conventional variables should be considered first, because as with everything else in the world, nothing is free—i.e., the routines presented here are slow in comparison to simply using standard numeric variables.)

The secret to these routines is to convert a TI BASIC number into a 2-byte (16-bit) integer representation. Strings are the perfect medium for the 2-byte integer. Strings contain ASCII characters; each character uses 8 bits and can have a value between 0 and 255. By using two characters, we can create a 16-bit quantity:

CHARACTER 1  
ASCII VALUE=2  
00000010

CHARACTER 2  
ASCII VALUE=5  
00000101

The two characters above could be combined with this type of statement:

**A\$=CHRS(2)&CHRS(5)**

The first character would have a weight of 256 because it contains the high-order byte of the value. The second character would have a weight of 1 because it contains the low-order byte. The value of the above string could be found like this:

$$\begin{array}{rcl} V = (\text{ASC}(\text{SEG}\$(A$,1,1)) * 256) & + & (\text{ASC}(\text{SEG}\$(A$,2,1))) \\ (2 * 256) & + & 5 \\ & & = 517 \end{array}$$

Separate the characters from each other using the **SEG\$** function. The ASCII value of the first character is then multiplied by 256. The ASCII value of the second character is added to that result. Thus we arrive at a value of 517.

To create an integer string of a number, V, we must perform the opposite process:

**A\$=CHRS(INT(V/256))&CHRS(V-INT(V/256)\*256)**

To simplify the programming process, define these two lines as functions. As a function you can pass a number or a string, and return the converted result. The two functions are:

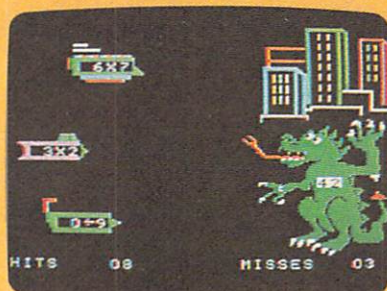
**DEF NTSS(V)=CHRS(INT(V/256))&CHRS(V-INT(V/256)\*256)**

**DEF STN(A\$)=(ASC(SEG\$(A\$,1,1))\*256)+(ASC(SEG\$(A\$,2,1)))**

—William K. Balthrop



*What do flying saucers  
and  
fire-breathing dragons  
have to do  
with multiplication  
and division?*



Name: Dragon Mix  
Program Type: Educational game  
Machine: TI-99/4A  
Manufacturer: Texas Instruments  
TI-CARES  
1-800-842-2737  
\$15.95  
Price: Poor Fair Good Excellent  
Performance:   
Engrossment:   
Documentation:

# DRAGON MIX

**A Review by Steve Nelson**  
HCM Staff

One of the oldest ways of capturing a child's interest in an educational program is to make it look like an arcade game. The idea is to *entertain* in order to *educate*. This approach can get more than old—at times, it leaves me wishing for something more original. I believe that children can and do learn without needing to be "entertained," or fooled into thinking that they are not really learning, but playing. All children have one propensity—the desire to discover new things—i.e., to learn.

## Make Learning Fun

*Dragon Mix* is an educational "game" that uses the old, standard, make-learning-fun approach. The object of this game is to defend the city from three attacking spaceships—each displaying a multiplication or division problem. A "student" may defend the city with the help of an enormous, fire-breathing dragon that bears the correct answer to one of the spaceship problems on its belly button. After solving one of the problems, the student can try to "equalize" the ship (blow it out of the sky) by making the dragon breathe fire on it. For this, the student presses the space bar when the dragon's tongue is aimed at the chosen ship (the dragon's tongue moves rapidly between the three advancing spaceships). If the answer is wrong, or if the dragon breathes fire at an incorrect spaceship, the student loses a point, and the spaceships advance even closer to the city. Once they get beyond a certain point, they will destroy part of the city by vaporizing one of the skyscrapers.

On the higher difficulty levels, the ships advance quite fast, and the child playing the game must quickly determine the correct firing angle and press the space bar when the appropriate ship is in line. Good hand/eye coordination—as much as math skill—is essential when playing this game.

## Rapid-Fire Calculations

This game also relies on the speed of the player calculating the correct answers, because the ships advance at a very fast pace. At the beginning of each game, the player may choose from 9 levels of play and 3 sets of problem ranges (multiples of 3, 6, or 9). On the upper

levels, the game would be quite challenging for grade school students. The capability to set the game to a student's skill level is one of the best features about *Dragon Mix* because it gives the student time to work on his or her skills in a less-pressured environment—and, as the student progresses, the difficulty level can be increased. Too often students are discouraged because they are unable to complete an educational game, or they cannot answer enough questions correctly before gameplay ends—all of which can lead to frustration and lost interest.

*Dragon Mix* has good graphics and sound effects, which add to the make-learning-fun aspect of the game. The documentation is helpful, with its easy-to-follow instructions. At the end of each game a screen appears that shows you the number of problems that you answered correctly and the number that you got wrong.

The program also keeps track of your highest score while playing.

*"Good hand/eye  
coordination—as much as  
math skill—is essential  
when playing this game."*

## Regressing

While I checked out this game, I tried to put myself in the place of a 10-year-old boy who is somewhat intimidated and confused about the whole idea of higher math. He understands addition and subtraction (as long as he doesn't use fractions) but multiplication and division just don't seem to be his cup of tea. Playing the game was easy enough (even in my 10-year-old state of mind), and although the ships advance rapidly, I could eliminate some of the pressure by changing the speed of the advancing ships or the difficulty level of the problems. From this perspective, the game was absorbing and I did have to *think math*.

While I'm not crazy about using video game formats to teach children, I do see their place. However, making the entire game more appealing would be a better way to go—one doesn't have to blow up aliens and monsters to keep a child's interest. I see the arcade approach more as a simple solution, rather than a well-thought-out approach to education. *Dragon Mix* is one such educational program that took the traditional, easy way out in meeting the interest/education factor. New approaches to educational software should be encouraged as the inventory of shoot-em-up learning games continues to build.



# 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:



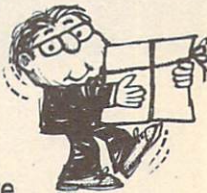
## Software

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

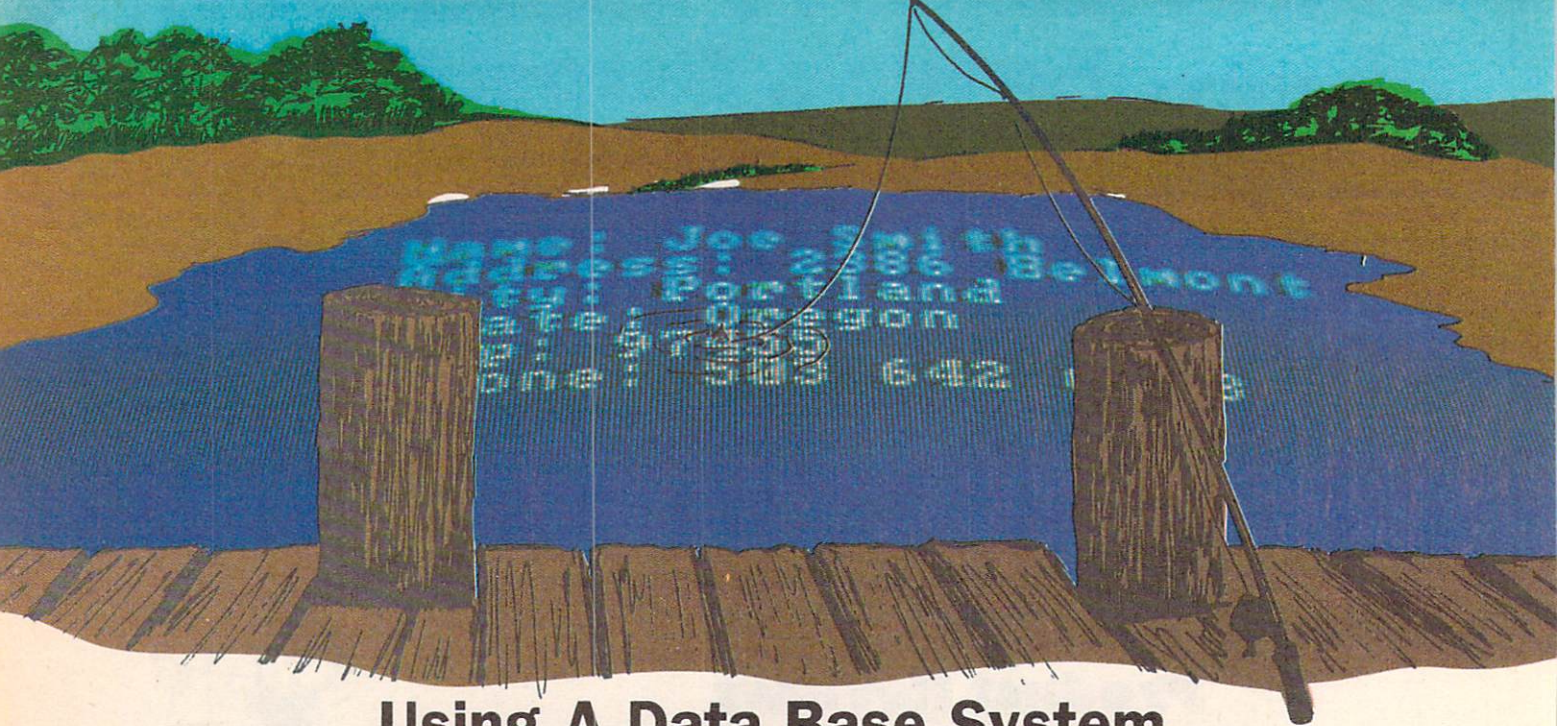
Author of "Build a LOGO Adventure"

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



# FIELD & SCREEN:



## Using A Data Base System

by Bill Crouch

*So, you couldn't wait to start typing into your powerful new data base program? And now, with a disk nearly full, you realize that your hastily-built format just won't work.*

The person using a microcomputer as an aid in a small home business, club, or organization usually only needs three programs to lift the burden of routine handiwork: a word-processing program, an electronic spreadsheet, and an information storage and retrieval program (known in computer circles as a "data-base management program"). Much has been written about the proper usage of word processors and spreadsheets, yet the novice data-base user does not have the same support. Except for the minimal "how-to" information contained in the typical program manual, the data-base user is left to struggle in the mire of unfamiliar terms and confusing commands.

This article gives a brief introduction to data bases and their jargon and then moves step-by-step through the process of solving a problem using a typical data-base program. Readers who think through this process with us will be in a much better position to assess their needs in order to either purchase or use a data-base program. The information contained in this article applies to any data-base program running on any computer.

### Background

What is a data-base management program and why use one? A "data base" is a collection of information organized around some topic or theme. We have many noncomputerized data bases in our lives. Obvious examples are an address book, a listing of property maintained for insurance purposes, or a Christmas card list. An example of a computerized data-base product is the telephone directory.

*Wouldn't it have been nice if someone had told you the right way to set up your system in the first place . . .*

If we are dealing with a sizable volume of information, or if the information we are keeping frequently changes, we can benefit from a data-base management program. Anyone who has tried

to maintain a mailing list or (can you imagine) the telephone directory by hand can testify to the need for some way to easily add and remove names, change addresses, and quickly sort the list into alphabetical or zip code order. A good data-base program allows us to do just that.

To illustrate the process of setting up a typical data base, we will design a hypothetical inventory for a large computer-users club. (See "Steps In Design," next page.) In our imaginary club, we purchase some products at wholesale—or obtain public domain software—and then loan them to our members. Our examples here will be suggestive of what can be done with this type of material; they are not an exhaustive treatment of the situation.

### Terminology

Every area of knowledge has its own special words that allow us to talk specifically about what we are doing. Computers are no exception. If we are going to be able to apply what we are learning to an actual data-base program, we will have to become familiar with the few terms listed below.

If you are keeping formal records of one kind or another, you are probably familiar with the physical objects used. Your information might be kept on a form consisting of one or several pages. The form would contain a number of questions, each one followed by a blank





space in which you can write your answer. The blank space might be short if it is to contain the abbreviation for a state of residence. It might be very long if it is asking for a street address. All of the forms of the same kind might be kept in a file folder. Or, if there are many of them, they might have to be kept in two or more folders.

From time to time, you might want to go through all of your forms and summarize the information contained in them. You might want to list together all of the forms of a certain kind in alphabetical order and add up various types of information, providing totals at the bottom of the summary. This kind of summary is the main product derived from such an assortment of forms, and it is called a *report*.

An electronic filing system is similar, but it uses its own special language for the parts of the system. Instead of a file folder, we have a *file* as the largest storage unit in our system. A data file holds all of the information on a subject that the computer can deal with at one time. So our user's club might have one file for its inventory, and another one for its Christmas card list. If our file is very large, we might have to keep it on two or more data disks.

Data files are made up of *records*. If you were the secretary of a large computer-user's club and had a file containing all information on your entire membership, then everything you needed to know about a single member would be one record. A record is the logical equivalent of a form. Just as a form may be made up of more than one page, so some data-base programs let us have more than one screen-page of information in a record. A screen page is all of the information that we can view on our computer screen at one time.

And just as a form is made up of separate questions and blanks to fill in, so a record is made up of *fields*. A field is one item of information in one record. Typical fields in a member's record might be last name, phone number, or zip code. An item on a form might have room for a long or a short answer. A field works the same way. In fact, the person who sets up the data base must specify the maximum number of characters that can exist in each field. The unit used to measure *field length* is the *character*. A character is one letter, number, or space. A social security number would be 11 characters in length (123-56-8911).

## Steps In Design

We should always follow 10 basic steps in creating a record-keeping system on a computer. These steps apply whether we are using an off-the-shelf data-base management program or are writing a record-keeping program ourselves. If followed with care, these steps will

**Figure 1.**  
**Software & Equipment Report Samples**

INVENTORY (REPORT 1)						
Product Name	Inventory Number	Cost Each	Price Each	On Hand	On Lend	Year to Date

PRICE LIST (REPORT 2)		
Product Name	Inventory Number	Price Each

PROPOSED PURCHASE (REPORT 3)				
Product Name	Inventory Number	Need	Cost Each	Total

INVENTORY VALUE (REPORT 3)				
Product Name	Inventory Number	On Hand	Cost Each	Value

ensure that your system will give you the results that you want and need. Although I have been using data-base management programs for years and have even written a data-base management program, I must still follow the same design steps listed below. There are no real short cuts.

## 1. Read the Manual

This first step should be obvious, yet it is seldom done. *Read the manual first and do the exercises carefully.* That's not much to ask when learning to use a new program, but many people start out by inserting the disk, turning on the computer, and trying to set up a file.

You should have a pad of paper at your side while working through the manual. Take notes on any program limitations (e.g., a maximum of 24 fields per

record) or problems. Try to find out the answers to your questions from the dealer who sold you the program or from your local user's group. After all, if you have trouble doing the sample problems, think of the trouble you'll have when you try to solve a real problem with the program.

## 2. Do Design Work on Paper

"Why bother writing anything down? We have a computer, don't we?" The greatest single problem in any area of serious computing is the tendency to jump into the middle of the process before taking time to work out on paper just what we are trying to do. If you have a word-processing program, you can use it to do the design paperwork. Whatever you do, don't start to set up your data base in the machine until you reach step 9. You'll be sorry if you do.

## 3. Determine Your Desired Products and Work Backwards

The natural thing to do is to make a list of the fields that you want to include in your data base and then set

***"The natural thing to do is to make a list of the fields that you want to include in your data base and then set it up. In this case, the natural thing to do is the wrong thing to do."***



it up. In this case, the *natural* thing to do is the *wrong* thing to do. If you approach it this way, you are likely to have a field or two which you never need, and to forget a field vital to some report.

You are using a data-base management program because you want to create some kind of product, usually something printed out on paper, a report of some kind. If you start with a list of carefully-designed reports to be produced and work backwards to the fields needed to make those products, you will be sure to have all of the fields you need, and will not waste valuable machine space by cluttering up your file with unneeded information.

If you need to duplicate an existing manual report, your choices are simple. If this is a new report or you want to make changes in the existing system, you may want to sit down at a typewriter (or word processor) and make up a report example containing sample data. (See Figure 1.) Once you have sample reports in hand, the rest of the process will proceed smoothly. If you also plan to view data on the screen, you can create sample report screens on paper, taking into consideration the size and layout of your computer's screen.

#### 4. Make a List of Fields

If it isn't used in a screen or a printed report, you don't need it. Write down each item on each report. If it is data that will need to be entered, put it on one list. If it is data that can be computed by the program, put it on another list. (See Figure 2.)

Now look at the list of computed fields. Write down the information that the computer will need to figure out these items. Are they already on your fields list? If not, add them. Then write down any needed formulas (VALUE = ON HAND times COST). (See Figure 3.) Will your data-base management program do this kind of operation? If you aren't sure, go back to the manual and check.

#### 5. List Field Characteristics

Most data-base programs want you to determine at least two things about each field: one is the maximum field length. Some field lengths are easy to determine. A field which contains a short zip code would have a length of five, a state abbreviation would have two, and a local telephone number would contain eight characters. But what about a last name or street address? These take a little research. If you specify a length that is too short, then you will have to abbreviate some of the longer entries. Few people like to see their last name abbreviated. If it is too long, it will waste space and may not fit on your mailing label or form. Try scanning your list of fields, looking for the longest examples. Also be aware of printer or report form limitations (e.g., if you have 3-1/2 inch mailing labels, no address line can exceed a total of 35 characters).

The other item to be determined is whether the field is to be designated as a numbers-only field (numeric field), or as a field that contains both letters and numbers (alphanumeric field, sometimes called a "string"). As a general rule, all fields should be set to alphanumeric unless you need to do math with them. Just because a field contains numbers does not mean that it should be treated as a numeric field. There is seldom a need to multiply your telephone number by your zip code. In one program, I erroneously set a field which should have contained a Social Security number as a numeric

field. When I entered a Social Security number, 560-60-8735, the computer took the dashes to be minus signs and changed it to -8235 by subtracting the second two numbers from the first—and saving the results. (Some programs also predefine types of fields according to the kind of data that these fields will store—such as a dollar sign or the date.)

#### 6. List Fields in Logical Order

The trick in designing a form is to lay out the form in as natural an order as possible. The following form would be very hard to fill out:

City \_\_\_\_\_  
Last Name \_\_\_\_\_  
Zip Code \_\_\_\_\_  
Street \_\_\_\_\_  
First Name \_\_\_\_\_  
State \_\_\_\_\_

We have come to expect to be asked for address information in the order of name, street, city, state, zip

code. Any major variation in this order confuses the person entering the data. To some degree, the same can be said about any block of data. Some arrangements will be clearer and more natural than others. Re-arrange your fields list into the order that you feel is most logical and easiest to use.

#### 7. Determine Fields for Sorting

One of the most useful tasks that can be performed by a data-base management program is arranging information into some specified order, or "sorting." You may want to print mailing labels which have been sorted into zip-code order as required by the post office. Or you may want the same information sorted by last names. You will find many useful ways to arrange your data when the computer does all of the work for you.

Some data-base management programs require you to determine the fields that you will use for sorting at the time you design the file. If so, give careful thought to the way you want the data presented in the reports. Programs of this kind often get bogged down by the presence of numerous sort keys, which also quickly eat up disk storage space. (See Figure 4 for final field layout.)

#### 8. Discuss Your Results with the Others Involved

No matter how good you are at figuring out things, others will have insights that you missed, and needs you forgot to consider. While it is wise to involve anyone else who needs the information you are storing at each step in the design, it is imperative that others be involved at this point. The next step calls for setting up the file in the computer. Things that are just ideas now will be harder to change as information is entered into the computer. An hour now could save days later.

Figure 2.

Entered Fields	Computed Fields
Product Name	Proposed Purchase
Inventory Number	Total
Cost	Value
Price	
On Hand	
On Lend	
Year to Date	
Maximum Items to Stock	(Not printed but used in the Proposed Purchase Report)

*"Just because a field contains numbers does not mean that it should be treated as a numeric field. There is seldom a need to multiply your telephone number by your zip code."*



**Figure 3.****FORMULAS USED IN COMPUTED FIELDS:**

NEED = MAXIMUM minus ON HAND minus ON LEND  
 TOTAL = PROPOSED PURCHASE times COST  
 VALUE = ON HAND times COST

**9. Set Up the File**

At last we can turn on the computer. Now is the time to review the steps given in the manual and to actually set up the file and the reports. If you have completed the previous eight steps and have all of your notes available, this step should be easy.

**10. Test It**

Make a copy of your data disk and enter about a dozen records. Then run each of your reports and check to see whether they came out the way you expected. If you made a serious design error, don't be afraid to start over again. It is easier to redesign your data file at this stage than to enter a large amount of data and have to redo it later.

If it works properly, go back and enter the rest of your data. Save the original data disk while it's unused. Now if you ever need another copy of this data base, perhaps for another year's data, you can make a new copy of your blank data disk and use that.

**A Personal Application**

Now that we have thought through the process in theory, we should take a few moments and attempt to apply it to an individual problem. We could keep track of many things, but let's pick something related to computers.

Because it doesn't take long to fill your shelves with a lot of floppy disks (or tapes), many containing several programs, you may lose track of just what is where. We will start by reading the manual. Although we are using a simple data-base program, we find no limitations that would be a problem for this application. Let's say our example limits us to 23 fields, but we won't use nearly that many.

We are careful to do all of our design work on paper, listing in the next steps the things we will need to keep track of and the various decisions we make.

We want three different products from our data base. First, we want an alphabetical listing of our programs and the numbers of the disks that they are on. Previously, we decided to sequentially number all of our disks and keep them in boxes, with the range of disks inside clearly marked on the front. Second, we want a listing of our programs by category. If we want to play an educational game, we will know our choices and where to look. Third, we want a report which gives us the dollar value of our software collection for insurance purposes. Many of our programs were purchased and would be expensive to replace. Some are "public domain" software, which is distributed free by our local computer club.

**Figure 4.**

Fields:	Field Type:	Field Length:
Product Name	Alpha	20 ← Primary Sort Key
Size	Alpha	10
Order Number	Alpha	15 ← Secondary Sort Key
Cost	Numeric	5
Price	Numeric	5
On Hand	Numeric	2
On Lend	Numeric	2
Year to Date	Numeric	3
Maximum Items to Stock *	Numeric	2
<b>Computed:</b>		
Need (Max — On Hand — On Lend)	Numeric	2
Total (Need * Cost)	Numeric	6
Value (On Hand * Cost)	Numeric	6

The reports might look like this:

**BY NAME**

Program Name: Disk #: Comments:

**BY CATEGORY**

Category: Sub-Category: Program Name: Disk #: Comments:

**VALUE REPORT**

Program Name: Cost:

Now we can look at our three reports and make a list of the fields used. It might look something like this:

Program Name  
 Disk Number  
 Comments  
 Category  
 Sub-Category  
 Cost

The final steps involve listing the field characteristics, arranging the fields in logical order, and choosing our sort keys. These three options are reflected in the following fields list:

Fields	Type	Length	Sort Key
Disk #	Alpha	20	No
Program Name	Alpha	10	Yes
Category	Alpha	15	Yes
Sub-Category	Numeric	5	Yes
Cost	Numeric	5	No
Comments	Alpha	40	No

Now we can talk to those who share our system. If it looks correct, we can now set up our data base and test it with some real data. Figure 5 shows part of one report.

**Conclusion**

Data-base management is one of the most useful functions of a computer. Its potential is limited only by the user's imagination and the relative unfamiliarity of the process. Unfortunately, the proper way to use a data-base management program is not obvious. The user needs to think through the process carefully before going to the computer. Time spent working on paper will be richly rewarded by the ease in which the data base is created and by its usefulness. You will find that a systematic approach to problem-solving with your computer always pays off in an improved product and less wasted time. The above process has been refined and tested, and it guarantees maximum usefulness from your data base program.

**Figure 5.**

File: MY DISKS

Report: EXAMPLE 5

Disk #	PROGRAM NAME	CATEGORY	SUB-CATEGORY	COST	COMMENTS
2	Space Zoom	Game	Arcade	0	Too slow
2	Subs	Game	Arcade	0	
5	Dueling Digits	Game	Educational	39.50	Arcade action
3	Hex	Utility	Programming	0	Dec/Hex convert
4	Tracer	Utility	Programming	49.95	Debugger
1	ZipWriter	Word Processor		239.95	Excellent












# HOME COMPUTER™

## PROGRAM LISTINGS

### CONTENTS

	 Page No.	 Page No.	  Page No.	 Page No.
It Figures!	84	88	92	94
Evacu-Pod	96	99	102	105
Switch 'N' Spell	106	110	113	115
Laserithmetic	117	118	119	120
The Organizer Reports	121	123	125	126

## HOME COMPUTER MAGAZINE'S

### BLANK MEDIA SERVICE



As a service to our readers who prefer to key-in their own programs, we are able to offer high-quality blank diskettes and cassettes at low prices.



	Subscriber Price	Non-Subscriber Price
<b>10 Diskettes</b> 5 1/4" certified single-sided, double density with reinforced hub rings. Bulk-packaged 10 to a set with separate white envelopes and identification labels.	<b>\$19.95</b> plus \$3.00 shipping*	<b>\$29.95</b> plus \$3.00 shipping*
<b>12 Cassette Tapes</b> C-20 digital computer cassettes (nominally 10 minutes per side) with 5 screw housing for data integrity.	<b>\$14.95</b> plus \$3.00 shipping*	<b>\$24.95</b> plus \$3.00 shipping*

\*U.S. only—Canada and Foreign Surface add \$1.50 to shipping costs.

We can offer this service at such low prices because of the large quantities of raw media we buy for our "ON TAPE" and "ON DISK" software—and we want to pass these tremendous savings on to our valuable readers.

Offer & Prices Subject To Change Without Notice.

Please Send Me:

- ☐ \_\_\_\_\_ set(s) of Diskettes  
 (10 Disks to a set)  
☐ \_\_\_\_\_ set(s) of Cassettes  
 (12 Cassettes to a set)

Please Print

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

Subscriber No. \_\_\_\_\_

(Found at the top of your address label.)

MUST BE IN US FUNDS DRAWN ON US BANK

☐ Check or Money Order Enclosed  
 Bill my ☐ VISA ☐ MASTERCARD

Total Order \$ \_\_\_\_\_

Account Number \_\_\_\_\_

Signature \_\_\_\_\_ Exp. Date \_\_\_\_\_

Phone No. ( ) \_\_\_\_\_

Home Computer Magazine  
 P.O. Box 70288 • Eugene, OR  
 97401

For information on ordering TOLL FREE, see blind-in card located at the middle of this magazine.

Please clip or photocopy coupon

PROGRAM LISTING



# PROGRAM LISTING

```

100 REM ***** IT FIGURES! *****
110 REM * * * * *
120 REM * * * * *
130 REM * * * * *
140 REM * * * * *
150 REM * * * * *
160 REM * * * * *
170 REM * * * * *
180 REM * * * * *
190 REM * * * * *
200 REM *****
210 REM *****
220 REM *****
230 REM *****
240 REM *****
250 REM *****
260 REM *****
270 REM *****
280 REM *****
290 REM *****
300 REM *****
310 REM *****
320 REM *****
330 REM *****
340 REM *****
350 REM *****
360 REM *****
370 REM *****
380 REM *****
390 REM *****
400 REM *****
410 REM *****
420 REM *****
430 REM *****
440 REM *****
450 REM *****
460 REM *****
470 REM *****
480 REM *****
490 REM *****
500 REM *****
510 REM *****
520 REM *****
530 REM *****
540 REM *****
550 REM *****
560 REM *****
570 REM *****
580 REM *****
590 REM *****
600 REM *****
610 REM *****
620 REM *****
630 REM *****
640 REM *****
650 REM *****
660 REM *****
670 REM *****
680 REM *****
690 REM *****
700 REM *****
710 REM *****
720 REM *****
730 REM *****
740 REM *****
750 REM *****
760 REM *****
770 REM *****
780 REM *****
790 REM *****
800 REM *****
810 REM *****
820 REM *****
830 REM *****
840 REM *****
850 REM *****
860 REM *****
870 REM *****
880 REM *****

```

```

890 VTAB: 22: HTAB: 3: INVERSE: PRINT "P
900 " : NORMAL INVERSE: PRINT "OUT"; NORM
910 HTAB: 17: INVERSE: PRINT "K"; NORM
920 HTAB: 25: INVERSE: PRINT "B"; NORM
930 HTAB: 23: HTAB: 1: INVERSE: PRINT "<
ESC>": NORMAL: PRINT "LEAVE";
940 " : 17: INVERSE: PRINT "J"; NORM
950 HTAB: 25: INVERSE: PRINT "I"; NORM
960 HTAB: 24: HTAB: 3: INVERSE: PRINT "L
" : NORMAL: PRINT "LOAD
970 HTAB: 17: INVERSE: PRINT "O"; NORM
980 AL: PRINT "SAVE";
990 RETURN
1000 REM: SHOW ANSWER ON SCREEN
1010 VTAB: 17: HTAB: 10: CALL - 868
1020 PRINT CHR$(64 + MQ) = "VX: RETUR
N
1030 REM: ENTRY SUPERVISOR
1040 ON QS GOSUB 1110, 2230, 3130
1050 IF IN$ = C$(15) THEN 1050
1060 GOTO 1020
1070 VTAB: 10: HTAB: 1: PRINT "DO YOU WISH
TO END THE PROGRAM?"
1080 VC = END: HC = 34: GOSUB 6030
1090 IF IN$ < ">" THEN GOSUB 650: GO
TO 1020
1100 RETURN
1110 REM: VARIABLE ENTRY
1120 IF QS < 1 THEN RETURN
1130 GOSUB 1290
1140 HX = 1
1150 HC = 1
1160 X = HX + 2 + VL(VK, 2): VC = VL(VK, 1
): GOSUB 6030
1170 IF ASC (IN$) > 31 THEN 1240
1180 X = 0
1190 FOR IT = 1 TO 15: IF IN$ = C$(IT)
THEN NEXT
1200 IF X = 15 THEN RETURN
1210 IF X = 0 THEN PRINT BL$;: GOTO 114
0
1220 ON X GOSUB 1410, 1470, 1470, 1520, 1580
, 1640, 1680, 1730, 1810, 1870, 1950, 2030
, 2090, 2160
1230 ON JP GOTO 1140, 1110
1240 STOP
1250 ET$(HX) = IN$: HX = HX + 1
1260 IF HX > 15 THEN HX = 15
1270 GOTO 1140
1280 RETURN
1290 REM: CONVERT NUMBER TO STRING ARR
AY
1300 X$ = LEFT$( (STR$(VR(VK)) + "
"), 15)
1310 FOR IT = 1 TO 15: ET$(IT) = MID$(X
$, IT, 1): NEXT
1320 RETURN
1330 REM: CONVERT STRING ARRAY TO NUMBE
R
1340 X$ = " ": FOR IT = 1 TO 15: X$ = X$ +
ET$(IT): NEXT
1350 VR(VK) = VAL (X$)
1360 RETURN
1370 REM: JUSTIFY NUMBER
1380 X$ = LEFT$( (STR$(VR(VK)) + "
"), 15)
1390 VTAB: VL(VK, 1): HTAB: VL(VK, 2) + 3: P
RINT X$;
1400 RETURN
1410 REM: EDIT CHANGE
1420 JP = 2
1430 GOSUB 1330: GOSUB 1370
1440 QS = 3
1450 MQ = VK
1460 RETURN
1470 REM: TAB
1480 JP = 2
1490 GOSUB 1330: GOSUB 1370
1500 QS = 2
1510 RETURN
1520 REM: CURSOR UP
1530 JP = 2
1540 GOSUB 1330: GOSUB 1370
1550 IF VK = 1 THEN VK = 8: RETURN
1560 VK = VK - 1
1570 RETURN
1580 REM: CURSOR DOWN
1590 JP = 2
1600 GOSUB 1330: GOSUB 1370
1610 IF VK = 8 THEN VK = 1: RETURN
1620 VK = VK + 1
1630 RETURN
1640 REM: CURSOR LEFT
1650 JP = 1
1660 IF HX > 1 THEN HX = HX - 1
1670 RETURN
1680 REM: CURSOR RIGHT
1690 JP = 1
1700 IF HX = 15 THEN HX = 1: RETURN
1710 HX = HX + 1
1720 RETURN
1730 REM: INSERT SPACE

```

**Continued.**



```

1730 JP = 1
1740 IF HX RETURN = 15 OR ET$(15) < > " " THEN
1750 FOR IT = 1 TO 14 TO HX STEP - 1
1760 ET$(IT) = ET$(IT) + 1
1770 VTAB VL(VK, 1): HTAB IT + VL(VK, 2) +
3: PRINT ET$(IT + 1): NEXT
1780 ET$(HX) = VTAB VL(VK, 1): HTAB
HX + VL(VK, 2) + 2: PRINT ET$(HX);
1790 RETURN
1800 REM BLANK FIELD
1810 JP = 1
1820 HX = 1
1830 FOR IT = 1 TO 15: ET$(IT) = " "
1840 VTAB VL(VK, 1): HTAB VL(VK, 2) + 2 +
IT: PRINT " "; NEXT
1850 RETURN
1860 REM DELETE CHARACTER
1870 JP = 1
1880 IF HX = 15 THEN 1920
1890 FOR IT = HX TO 13 + 1
1900 ET$(IT) = ET$(IT + 1)
1910 VTAB VL(VK, 1): HTAB VL(VK, 2) + IT +
2: PRINT ET$(IT): NEXT
1920 ET$(15) = VTAB VL(VK, 1): HTAB
VL(VK, 2) + 17: PRINT " ";
1930 RETURN
1940 REM EVALUATE EQUATION FROM NUMBER
ENTRY FIELDS
1950 JP = 1
1960 GOSUB 1330: GOSUB 1370
1970 IF MQ < > VK THEN MQ = VK: GOSUB 3
300
1980 GOSUB 3910
1990 HX = 1
2000 JP = 2
2010 RETURN
2020 REM DO PRINTOUT
2030 GOSUB 1330: GOSUB 1370
2040 GOSUB 5900
2050 HX = 1
2060 JP = 2
2070 RETURN
2080 REM LOAD FILE
HOME: HTAB 15: VTAB 1
INVERSE: PRINT "IT FIGURES!": NORM
AL: VTAB 3: HTAB 14: PRINT "LOAD F
ILE": DF$ = "L"
2110 VTAB 5: HTAB 3: PRINT "INPUT FILE N
AME:"
2120 GOSUB 6280: IF TS$ = " " THEN 2140
2130 FL$ = TS$: GOSUB 6720: GOSUB 6480
2140 GOSUB 650: JP = 2: RETURN
2150 REM SAVE FILE
HOME: HTAB 15: VTAB 1
INVERSE: PRINT "IT FIGURES!": NORM
AL: VTAB 3: HTAB 14: PRINT "SAVE F
ILE": DF$ = "S"
2180 VTAB 5: HTAB 3: PRINT "INPUT FILE N
AME:"
2190 GOSUB 6280: IF TS$ = " " THEN 2210
2200 FL$ = TS$: GOSUB 6720: GOSUB 6590
2210 GOSUB 650: JP = 2: RETURN
2220 REM VARIABLE DESCRIPTION ENTRY
2230 IF QS < > 2 THEN RETURN
2240 GOSUB 2400
2250 HX = 1
2260 HC = HX + VE(VK, 2): VC = VE(VK, 1): G
OSUB 6030
2270 IF ASC(INS) > 31 THEN 2350
2280 X = 0
2290 FOR IT = 1 TO 13: IF INS = C$(IT)
THEN X = IT
2300 NEXT
2310 IF X = 13 THEN RETURN
2320 IF X = 0 THEN PRINT BL$: GOTO 226
0
2330 ON X GOSUB 2460, 2520, 2520, 2570, 2630
, 2690, 2730, 2780, 2860, 2920, 3000, 3080
2340 ON JP GOTO 2260, 2230
2350 ET$(HX) = INS: HX = HX + 1
2360 IF HX > 20 THEN HX = 20
2370 GOTO 2260
2380 RETURN
2390 REM CONVERT STRING TO STRING ARRAY
FOR IT = 1 TO 20: ET$(IT) = MIDS(V
D$(VK), IT, 1): NEXT
2410 RETURN
2420 REM CONVERT STRING ARRAY TO STRIN
G
2430 VD$(VK) = " ": FOR IT = 1 TO 20: VD$(
VK) = VD$(VK) + ET$(IT): NEXT
2440 RETURN
2450 REM EDIT CHANGE
2460 JP = 2
2470 GOSUB 2430
2480 QS = 3
2490 MQ = VK
2500 RETURN
2510 REM TAB
2520 JP = 2
2530 GOSUB 2430
2540 QS = 1
2550 RETURN
2560 REM CURSOR UP
2570 JP = 2
2580 GOSUB 2430

```

```

2590 IF VK = 1 THEN VK = 8: RETURN
2600 VK = VK - 1
2610 RETURN
2620 REM CURSOR DOWN
2630 JP = 2
2640 GOSUB 2430
2650 IF VK = 8 THEN VK = 1: RETURN
2660 VK = VK + 1
2670 RETURN
2680 REM CURSOR LEFT
2690 JP = 1
2700 IF HX > 1 THEN HX = HX - 1
2710 RETURN
2720 REM CURSOR RIGHT
2730 JP = 1
2740 IF HX = 20 THEN HX = 1: RETURN
2750 HX = HX + 1
2760 RETURN
2770 REM INSERT SPACE
2780 JP = 1
2790 IF HX = 20 OR ET$(20) < > " " THEN
RETURN
2800 FOR IT = 19 TO HX STEP - 1
2810 ET$(IT + 1) = ET$(IT)
2820 VTAB VE(VK, 1): HTAB IT + VE(VK, 2):
PRINT ET$(IT + 1): NEXT
2830 ET$(HX) = VTAB VE(VK, 1): HTAB
HX + VE(VK, 2): PRINT ET$(HX);
2840 RETURN
2850 REM BLANK FIELD
2860 JP = 1
2870 HX = 1
2880 FOR IT = 1 TO 20: ET$(IT) = " "
2890 VTAB VE(VK, 1): HTAB VE(VK, 2) + IT:
PRINT " "; NEXT
2900 RETURN
2910 REM DELETE CHARACTER
2920 JP = 1
2930 IF HX = 20 THEN 2970
2940 FOR IT = HX TO 19
2950 ET$(IT) = ET$(IT + 1)
2960 VTAB VE(VK, 1): HTAB VE(VK, 2) + IT:
PRINT ET$(IT): NEXT
2970 ET$(20) = VTAB VE(VK, 1): HTAB
VE(VK, 2) + 20: PRINT " ";
2980 RETURN
2990 REM SOLVE EQUATION FROM VARIABLE D
ESCRIPTION
3000 JP = 1
3010 GOSUB 2430
3020 IF MQ < > VK THEN MQ = VK: GOSUB 3
300
3030 GOSUB 3910
3040 HX = 1
3050 JP = 2
3060 RETURN
3070 REM DO PRINTOUT
3080 GOSUB 2430
3090 GOSUB 5900
3100 HX = 1: JP = 2
3110 RETURN
3120 REM EQUATION ENTRY
3130 GOSUB 3300
3140 HX = HS(MQ): VC = 14
3150 IF QS = 1 THEN RETURN
3160 HC = HX: GOSUB 6030
3170 IF ASC(INS) > 31 THEN 3250
3180 X = 0
3190 FOR IT = 1 TO 13: IF INS = C$(IT)
THEN X = IT
3200 NEXT
3210 IF X = 13 THEN RETURN
3220 IF X = 0 THEN PRINT BL$: GOTO 316
0
3230 ON X GOSUB 3350, 3350, 3400, 3470, 3510
, 3550, 3580, 3610, 3670, 3720, 3780, 3870
3240 GOTO 3150
3250 EQ$(HX) = INS: HX = HX + 1
3260 IF HX > 80 THEN HX = 80
3270 GOTO 3160
3280 RETURN
3290 REM EQUATION PRINT
3300 FOR IT = 1 TO 80
3310 VTAB 14: HTAB IT: EQ$(IT) = MIDS(M
Q$(MQ), IT, 1): PRINT EQ$(IT): NEXT
3320 VX = 0: GOSUB 990
3330 RETURN
3340 REM EDIT CHANGE
3350 QS = 1: HS(MQ) = HX
3360 GOSUB 3810
3370 GOSUB 1290
3380 RETURN
3390 REM TAB
3400 IF HX > 1 AND HX < = 20 THEN HX
= 21: RETURN
3410 IF HX > 21 AND HX < 40 THEN HX =
40: RETURN
3420 IF HX = 40 THEN HX = 41: RETURN
3430 IF HX > 41 AND HX < = 60 THEN H
X = 61: RETURN
3440 IF HX > 61 AND HX < 80 THEN HX =
80: RETURN
3450 IF HX = 80 THEN HX = 1: RETURN
3460 REM CURSOR UP
3470 IF HX < = 40 THEN HX = HX + 40: RE
TURN
3480 HX = HX - 40

```

Continued



```

3490 RETURN
3500 REM CURSOR DOWN
3510 IF HX > 40 THEN HX = HX - 40: RETURN
3520 HX = HX + 40
3530 RETURN
3540 REM CURSOR LEFT
3550 IF HX > 1 THEN HX = HX - 1
3560 RETURN
3570 REM CURSOR RIGHT
3580 IF HX < 80 THEN HX = HX + 1
3590 RETURN
3600 REM INSERT CHARACTER
3610 IF HX = 80 OR EQ$(80) < " " THEN
3620 FOR IT = 79 TO HX STEP -1
3630 EQ$(IT + 1) = EQ$(IT): VTAB 14: HTA
3640 B IT + 1: PRINT EQ$(IT + 1): NEXT
3650 VTAB 14: HTAB HX: EQ$(HX) = " ": PRI
3660 RETURN
3670 REM BLANK FIELD
3680 HX = 1
3690 FOR IT = 1 TO 80: EQ$(IT) = " ": NEXT
3700 VTAB 14: HTAB IT: PRINT " ": NEXT
3710 RETURN
3720 REM DELETE CHARACTER
3730 IF HX < 80 THEN 3750
3740 FOR IT = HX TO 79: EQ$(IT) = EQ$(IT
3750 + 1)
3760 VTAB 14: HTAB IT: PRINT EQ$(IT): N
3770 EXT
3780 EQ$(80) = " ": VTAB 14: HTAB 80: PR
3790 INT
3800 RETURN
3810 REM EVALUATE EQUATION FROM EQUATI
3820 ON FIELD
3830 GOSUB 3910
3840 RETURN
3850 REM STRING ARRAY TO STRING CONVER
3860 T
3870 HS(MQ) = HX
3880 MQ$(MQ) = " "
3890 FOR IT = 1 TO 85
3900 MQ$(MQ) = MQ$(MQ) + EQ$(IT): NEXT
3910 RETURN
3920 REM DO PRINTOUT
3930 GOSUB 3810
3940 GOSUB 5900
3950 RETURN
3960 REM PARSE AND SOLVER
3970 VTAB 17: HTAB 1: CALL - 868
3980 BH(0) = 2: GOSUB 4230
3990 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4000 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4010 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4020 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4030 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4040 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4050 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4060 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4070 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4080 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4090 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4100 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4110 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4120 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4130 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4140 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4150 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4160 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4170 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4180 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4190 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4200 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4210 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4220 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4230 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4240 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4250 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4260 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4270 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4280 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4290 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4300 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4310 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4320 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4330 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4340 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4350 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4360 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4370 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4380 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4390 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4400 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4410 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4420 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4430 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4440 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4450 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4460 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4470 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4480 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4490 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4500 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4510 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4520 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4530 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4540 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4550 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4560 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4570 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4580 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4590 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4600 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4610 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4620 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4630 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4640 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4650 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4660 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4670 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4680 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4690 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4700 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4710 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4720 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4730 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4740 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4750 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4760 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4770 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4780 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4790 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4800 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4810 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4820 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4830 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4840 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4850 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4860 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4870 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4880 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4890 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4900 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4910 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4920 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4930 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4940 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4950 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4960 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4970 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4980 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
4990 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5000 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5010 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5020 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5030 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5040 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5050 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5060 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5070 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5080 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5090 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)
5100 PT = 0: SK%(1) = 0: SK%(2) = 0: FG(2)

```

Continued



```

5110 IF EQ$(PT) THEN GOSUB 5130
5120 NEXT CK: PT = PT - 1: GOTO 5130
5130 BH(0) = 6: GOSUB 4230
5140 REM RETURN ALPHABETIC VARIABLE CHECK
5150 FOR EQ$(PT) = 1 TO 16
5160 IF EQ$(CK) THEN 5200
5170 NEXT CK
5180 PC$(7) = CHR$(48 + CK)
5190 BH(0) = 7: GOSUB 4230
5200 REM FUNCTION EVALUATE
5210 IF SK$(2) THEN JP = 5: RETURN
5220 ON ASC(RIGHT$(SK$(SK%(1)),1))
5230 GOSUB 5300,5310,5320,5330,5340,
5240 5350,5360,5370,5380,5400,5410
5250 SK$(1) = SK%(1) - 1
5260 IF SK$(1) OR SK$(SK%(1)) = "20"
5270 THEN RETURN
5280 ON VAL(LEFT$(SK$(SK%(1)),1))
5290 GOSUB 5500,5640
5300 SK$(SK%(2)) = SIN(SK$(SK%(2))): RET
5310 SK$(SK%(2)) = COS(SK$(SK%(2))): RET
5320 SK$(SK%(2)) = TAN(SK$(SK%(2))): RET
5330 SK$(SK%(2)) = ATN(SK$(SK%(2))): RET
5340 SK$(SK%(2)) = INT(SK$(SK%(2))): RET
5350 SK$(SK%(2)) = RND(SK$(SK%(2))): RET
5360 SK$(SK%(2)) = SGN(SK$(SK%(2))): RET
5370 SK$(SK%(2)) = ABS(SK$(SK%(2))): RET
5380 IF SK$(SK%(2)) < 0 THEN JP = 5: RETU
5390 SK$(SK%(2)) = SQR(SK$(SK%(2))): RET
5400 SK$(SK%(2)) = EXP(SK$(SK%(2))): RET
5410 IF SK$(SK%(2)) < = 0 THEN JP = 5: R
5420 SK$(SK%(2)) = LOG(SK$(SK%(2))): RET
5430 REM RIGHT PARENTHESIS EVALUATE
5440 IF SK$(1) = 0 OR SK$(SK%(1)) < > "
5450 20" THEN JP = 2: RETURN
5460 SK$(1) = SK%(1) - 1
5470 IF SK$(1) > 0 AND SK$(SK%(1)) < > "
5480 20" THEN ON VAL(LEFT$(SK$(SK%(1)),1))
5490 GOSUB 5240,5240,5450,5500,
5500 5640,5640
5490 REM BINARY OPERATORS EVALUATE
5500 IF SK$(2) < = 1 THEN JP = 4: RETUR
5510 ON VAL(RIGHT$(SK$(SK%(1)),1)) G
5520 OSUB 5540,5550,5560,5570,5610
5530 SK$(1) = SK%(1) - 1: SK%(2) = SK%(2)
5540 RETURN
5550 SK$(SK%(2)) = SK$(SK%(2)) - 1 + S
5560 SK$(SK%(2)) = SK$(SK%(2)) - 1 - S
5570 SK$(SK%(2)) = SK$(SK%(2)) - 1 * S
5580 SK$(SK%(2)) = SK$(SK%(2)) - 1 / S
5590 IF SK$(SK%(2)) > 0 THEN 5600
5600 JP = 4: IF PT < 80 THEN VTAB 14: H
5610 TAB PT + 1: PRINT EQ$(PT + 1)
5620 RETURN
5630 SK$(SK%(2)) = SK$(SK%(2)) - 1 / S
5640 K(SK%(2)): RETURN
5650 SK$(SK%(2)) = SK$(SK%(2)) - 1 ^ S
5660 K(SK%(2)): RETURN
5670 REM UNARY OPERATORS EVALUATE
5680 IF SK$(2) = 0 THEN JP = 3: RETURN
5690 IF SK$(SK%(1)) = "52" THEN SK$(SK%(2)
5700 ) = SK$(SK%(2)) - 1
5710 SK$(1) = SK%(1) - 1
5720 REM NUMERIC CONSTANTS EVALUATE
5730 SK = VAL(RIGHT$(PC$(LEN(PC$))
5740 ) - 1))
5750 GOSUB 5840
5760 REM VARIABLES EVALUATE
5770 SK = VR(ASC(RIGHT$(PC$(1)) - 48
5780 ))
5790 GOSUB 5840
5800 REM PUSH TO OPERATOR STACK
5810 SK$(1) = SK%(1) + 1
5820 SK$(SK%(1)) = PC$
5830 RETURN
5840 REM PUSH TO OPERAND STACK
5850 SK$(2) = SK%(2) + 1

```

```

5860 SK$(SK%(2)) = COMMON
5870 REM RETURN NUMERIC HANDLING ROUTI
5880 GOSUB 5810
5890 IF SK$(SK%(1)) OR SK$(SK%(1)) = "20"
5900 THEN RETURN
5910 IF LEFT$(SK$(SK%(1)),1) = "5" THE
5920 N GOSUB 5640
5930 IF SK%(1) > 0 AND LEFT$(SK$(SK%(1)
5940 ),1) = "4" THEN GOSUB 5500
5950 RETURN
5960 REM PRINTOUT
5970 HOME: PRINT " "
5980 PRINT "CHR$(4) "PR#1"
5990 PRINT " "
6000 IT FIG
6010 PRINT " "
6020 PRINT " "
6030 PRINT " "
6040 PRINT " "
6050 PRINT " "
6060 PRINT " "
6070 PRINT " "
6080 PRINT " "
6090 PRINT " "
6100 PRINT " "
6110 PRINT " "
6120 PRINT " "
6130 PRINT " "
6140 PRINT " "
6150 PRINT " "
6160 PRINT " "
6170 PRINT " "
6180 PRINT " "
6190 PRINT " "
6200 PRINT " "
6210 PRINT " "
6220 PRINT " "
6230 PRINT " "
6240 PRINT " "
6250 PRINT " "
6260 PRINT " "
6270 PRINT " "
6280 PRINT " "
6290 PRINT " "
6300 PRINT " "
6310 PRINT " "
6320 PRINT " "
6330 PRINT " "
6340 PRINT " "
6350 PRINT " "
6360 PRINT " "
6370 PRINT " "
6380 PRINT " "
6390 PRINT " "
6400 PRINT " "
6410 PRINT " "
6420 PRINT " "
6430 PRINT " "
6440 PRINT " "
6450 PRINT " "
6460 PRINT " "
6470 PRINT " "
6480 PRINT " "
6490 PRINT " "
6500 PRINT " "
6510 PRINT " "
6520 PRINT " "
6530 PRINT " "
6540 PRINT " "
6550 PRINT " "
6560 PRINT " "

```

Continued



```

6570 RETURN
6580 REM ** SAVE DISK ROUTINE **
6590 PRINT D$;"OPEN";FLS;"D1"
6600 PRINT D$;"CLOSE";FLS
6610 PRINT D$;"DELETE";FLS
6620 PRINT D$;"OPEN";FLS
6630 PRINT D$;"WRITE";FLS
6640 FOR IT=1 TO 8
6650 PRINT VR(IT)
6660 PRINT VD$(IT)
6670 PRINT MQ$(IT)
6680 NEXT IT
6690 PRINT D$;"CLOSE";FLS

```

```

6700 RETURN
6710 REM ** PLACE DISK IN DRIVE **
6720 VTAB=9:HTAB=1:PRINT "PLACE DISK I
N DRIVE 1 AND PRESS RETURN"
6730 IF PD THEN PRINT CHR$(4);"PREFIX
D1"
6740 GOSUB 6760: IF KB < > 141 THEN GO
TO 6740
6750 RETURN
6760 KB=PEEK(-16384): IF KB > 127
THEN POKE -16388,0: RETURN
6770 GOTO 6760

```

HCM

## IT FIGURES!

COMMODORE 64

```

100 REM ** IT FIGURES! **
110 REM ** COPYRIGHT 1985 **
120 REM EMERALD VALLEY PUBLISHING CO
130 REM BY ROBERT PASCHELKE
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 5.2.1
170 REM C-64 BASIC
180 REM
190 REM
200 GOSUB 250
210 GOSUB 510
220 GOSUB 930
230 PRINT "SHIFT CLR";:END
240 REM ** INITIALIZATION **
250 DIM VR(8),BH(7),FC$(11),UN$(2),BN$(
5),FG(2),VD$(8),SK$(80),SK%(
2)
260
270 DIM C$(14),VL(8,2),MQ$(8),EQ$(85),
ET$(20),VK$(16),HS(8)
280 FOR IT=1 TO 8:VR(IT)=0:NEXT
290 FOR IT=1 TO 8:VE(IT,1)=IT+2:VE(IT,2
)=20:NEXT
300 VD$(0)="":FOR IT=1 TO 19:VD$(0)=VD
$(0)+":NEXT
310 FOR IT=1 TO 8:VD$(IT)=VD$(0):NEXT
320 FOR IT=1 TO 8:HS(IT)=1:NEXT
330 LFS=CHR$(157):RTS=CHR$(29)
340 QS=1:VK=1:MQ=1:MQ$(0)=
350 FOR IT=1 TO 8:EQ$(IT)="":MQ$(0)=M
Q$(0)+":NEXT
360 FOR IT=1 TO 8:MQ$(IT)=MQ$(0):NEXT
370 FOR IT=1 TO 20:ET$(IT)=":NEXT
380 FOR IT=1 TO 8:VK$(IT)=CHR$(64+IT):N
EXT
390 DV$="":FOR IT=1 TO 40:DV$=DV$+"-":N
EXT
400 VX=0:FOR IT=1 TO 8:VL(IT,1)=IT+2:VL
(IT,2)=1:NEXT
410 C$(14)=CHR$(95):C$(13)=CHR$(20):C
$(12)=CHR$(13)
420 FOR IT=1 TO 11:READ X$:C$(IT)=CHR
$(ASC(X$)):NEXT
430 FOR IT=1 TO 11:READ FC$(IT):NEXT
440 FOR IT=1 TO 5:READ BN$(IT):NEXT
450 FOR IT=1 TO 2:READ UN$(IT):NEXT
460 RETURN
470 DATA "F1","F5","SHIFT CRSRUP",
"CRSRDOWN","SHIFT CRSRLEFT",
"CRSRRIGHT","SHIFT INST","SHIFT
F2","F7","F3","SHIFT F4",
"DATA","SIN","COS","TAN","INT",
"RND","SGN","ABS","SQR","EXP","LOG",
"DATA","+","-","/","*",
510 REM ** MAIN SECTION **
520 PRINT "SHIFT CLR";HTAB=15:VTAB=1:G
OSUB 790
530 PRINT "CTRL RVSON";IT FIGURES;CTRL
RVSON
540 FOR IT=1 TO 8:VTAB=VL(IT,1):HTAB=VL
(IT,2):GOSUB 790
550 PRINT VK$(IT)";:IF VR(IT)<0 THEN
PRINT VR(IT)
560 VTAB=VE(IT,1):HTAB=VE(IT,2):GOSUB 79
0
570 PRINT "VD$(IT)":NEXT
580 VTAB=12:HTAB=1:GOSUB 790:PRINT DV$;:
GOSUB 3240
590 VTAB=19:HTAB=1:GOSUB 790:PRINT DV$;:
GOSUB 620
600 RETURN
610 REM ** PROMPTS **
620 FOR IT=20 TO 24:VTAB=IT:GOSUB 850:NE
XT
630
640 VTAB=20:HTAB=1:GOSUB 790:PRINT "CTRL
RVSON";CTRL RVSON;ED SWITCH";
650 HTAB=14:GOSUB 790:PRINT "CTRL RVSON";
CRSR LEFT;CTRL RVSON;LEFT";
660 HTAB=29:GOSUB 790:PRINT "CTRL RVSON";
INST;CTRL RVSON;SPACE";
670 VTAB=21:HTAB=1:GOSUB 790:PRINT "CTRL
RVSON";F7;CTRL RVSON;SOLVE";
680 HTAB=14:GOSUB 790:PRINT "CTRL RVSON";
CRSR RT;CTRL RVSON;RIGHT";
690 HTAB=29:GOSUB 790:PRINT "CTRL RVSON";
DEL;CTRL RVSON;DEL CHAR";
700 VTAB=22:HTAB=1:GOSUB 790:PRINT "CTRL
RVSON";F3;CTRL RVSON;PRINT OUT";

```

```

710 HTAB=14:GOSUB 790:PRINT "CTRL RVSON";
CRSR UP;CTRL RVSON;UP";
720 HTAB=29:GOSUB 790:PRINT "CTRL RVSON";
F2;CTRL RVSON;DEL FLD";
730 VTAB=23:HTAB=1:GOSUB 790:PRINT "CTRL
RVSON";LT ARO;CTRL RVSON;EXIT";
740 HTAB=14:GOSUB 790:PRINT "CTRL RVSON";
CRSR DN;CTRL RVSON;DOWN";
750 HTAB=29:GOSUB 790:PRINT "CTRL RVSON";
RTRN;CTRL RVSON;TAB";
760 VTAB=24:HTAB=1:GOSUB 790:PRINT "CTRL
RVSON";F5;CTRL RVSON;SAVE";
770 HTAB=14:GOSUB 790:PRINT "CTRL RVSON";
F4;CTRL RVSON;LOAD";
780 RETURN
790 REM ** SIMULATE VTAB HTAB **
800 HTAB=HTAB-1
810 IF HTAB>0 AND HTAB<=39 THEN GOTO 83
0
820 IF HTAB>=40 AND HTAB<=79 THEN VTAB=
VTAB+1:HTAB=HTAB-40
830 POKE 781,VTAB:POKE 782,HTAB:POKE 783
,0:SYS 65520
840 RETURN
850 REM ** SIMULATE LINE ERASE **
860 FOR IT=1 TO 40:PRINT "":NEXT
870 VTAB=VTAB+1:HTAB=1:GOSUB 790:
880 RETURN
890 REM ** SHOW ANSWER ON SCREEN **
900 VTAB=18:HTAB=1:GOSUB 790:GOSUB 850
910 VTAB=18:HTAB=1:GOSUB 790:PRINT CHR$(
64+MQ)";VX
920 RETURN
930 REM ** ENTRY SUPERVISOR **
940 ON QS GOSUB 1020,2040,3020
950 IF IN$=C$(14) THEN ET$(HX)="":GOTO
970
960 GOTO 940
970 PRINT "SHIFT CLR";:VTAB=10:HTAB=1:
GOSUB 790:
980 PRINT "DO YOU WISH TO END THE PROGRA
M (Y/N)?";
990 VC=10:HC=38:GOSUB 6670
1000 IF IN$<>"Y" THEN GOSUB 510:GOTO 930
1010 RETURN
1020 REM ** VARIABLE ENTRY **
1030 IF QS<>1 THEN RETURN
1040 GOSUB 1180
1050 HX=1
1060 HC=HX+2+VL(VK,2):VC=VL(VK,1):GOSUB 6
670
1070 IF (IN$>"CTRL BLUE")*(IN$<"-")*(IN$
<>")*(IN$<>";")*(IN$<>CHR$(34))TH
EN 1140
1080 VTAB=VC:HTAB=HC:GOSUB 790:PRINT ET$(
HX)
1090 X=0:FOR IT=1 TO 14:IF IN$=C$(IT)TH
EN X=IT
1100 NEXT:IF X=0 THEN GOTO 1060
1110 IF X=14 THEN RETURN
1120 ON X GOSUB 1360,1230,1470,1530,1590,16
30,1680,1770,1900,1980,6430,1420,18
20
1130 ON JP GOTO 1060,1030,510
1140 ET$(HX)=IN$:HX=HX+1
1150 IF HX>15 THEN HX=15
1160 GOTO 1060
1170 RETURN
1180 REM ** CONVERT NUMBER TO STRING ARR
AY **
1190 X$=LEFT$(STR$(VR(VK))+
",16)
1200 IF LEFT$(X$,1)=" " THEN X$=RIGHT$(X$,L
EN(X$)-1)+
1210 FOR IT=1 TO 15:ET$(IT)=MID$(X$,IT,1
):NEXT
1220 RETURN
1230 REM ** SAVE FROM NUMBER ENTRY FIE
LD **
1240 GOSUB 1270:GOSUB 1310
1250 GOSUB 6250
1260 RETURN
1270 REM ** CONVERT STRING ARRAY TO NUMB
ER **
1280 X$="":FOR IT=1 TO 15:X$=X$+ET$(IT)
:NEXT
1290 VR(VK)=VAL(X$)
1300 RETURN
1310 REM ** JUSTIFY NUMBER **
1320 X$=LEFT$(STR$(VR(VK))+
",16)

```

Continued



```

1330 IF VAL(X$) < 0 THEN X$ = "-" + X$
1340 VTAB = VL(VK, 1) : HTAB = VL(VK, 2) + 2 : GOSUB 790 : PRINT X$;
1350 RETURN
1360 REM *** MOVE TO EDIT FIELD ***
1370 JP = 2
1380 GOSUB 1270 : GOSUB 1310
1390 QS = 3
1400 MQ = VK
1410 RETURN
1420 REM *** TAB *****
1430 JP = 2
1440 GOSUB 1270 : GOSUB 1310
1450 QS = 2
1460 RETURN
1470 REM ***** CURSOR UP *****
1480 JP = 2
1490 GOSUB 1270 : GOSUB 1310
1500 IF VK = 1 THEN VK = 8 : RETURN
1510 VK = VK - 1
1520 RETURN
1530 REM *** CURSOR DOWN ****
1540 JP = 2
1550 GOSUB 1270 : GOSUB 1310
1560 IF VK = 8 THEN VK = 1 : RETURN
1570 VK = VK + 1
1580 RETURN
1590 REM ***** CURSOR LEFT *****
1600 JP = 1
1610 IF HX > 1 THEN HX = HX - 1
1620 RETURN
1630 REM *** CURSOR RIGHT *****
1640 JP = 1
1650 IF HX = 15 THEN HX = 1 : RETURN
1660 HX = HX + 1
1670 RETURN
1680 REM ***** INSERT SPACE *****
1690 JP = 1
1700 IF HX = 15 OR ET$(15) < > " " THEN RETURN
1710 FOR IT = 14 TO HX STEP -1
1720 ET$(IT+1) = ET$(IT)
1730 VTAB = VL(VK, 1) : HTAB = IT + VL(VK, 2) + 3 : GOSUB 790 : PRINT ET$(IT+1) ; NEXT
1740 ET$(HX) = " " : VTAB = VL(VK, 1) : HTAB = HX + VL(VK, 2) + 2 : GOSUB 790 : PRINT ET$(HX);
1750 RETURN
1760 REM ***** BLANK FIELD ****
1770 JP = 1 : HX = 1
1780 FOR IT = 1 TO 15 : ET$(IT) = " "
1790 VTAB = VL(VK, 1) : HTAB = VL(VK, 2) + 2 + IT : GOSUB 790 : PRINT " " ; NEXT
1800 RETURN
1810 REM *** DELETE CHARACTER ****
1820 JP = 1
1830 IF HX = 1 THEN 1890
1840 FOR IT = HX - 1 TO 14
1850 ET$(IT) = ET$(IT+1)
1860 VTAB = VL(VK, 1) : HTAB = VL(VK, 2) + IT + 2 : GOSUB 790 : PRINT ET$(IT) ; NEXT
1870 ET$(15) = " " : VTAB = VL(VK, 1) : HTAB = VL(VK, 2) + 17 : GOSUB 790 : PRINT " " ; HX = HX - 1
1880 RETURN
1890 REM **EVALUATE EQUATION FROM NUMBER ENTRY FIELDS ****
1900 JP = 1
1910 GOSUB 1270 : GOSUB 1310
1920 IF MQ <> VK THEN MQ = VK : GOSUB 3240
1930 GOSUB 3910
1940 HX = 1
1950 JP = 2
1960 RETURN
1970 REM ***** DO PRINTOUT *****
1980 GOSUB 1270 : GOSUB 1310
1990 GOSUB 6060
2000 HX = 1
2010 JP = 2
2020 RETURN
2030 REM *** VARIABLE DESCRIPTION ENTRY ***
2040 IF QS <> 2 THEN RETURN
2050 GOSUB 2250
2060 HX = 1
2070 HC = HX + VE(VK, 2) : VC = VE(VK, 1) : GOSUB 667
2080 IF (INS > " FCTRL BLU") * (INS < "-" ) * (INS <> " ") * (INS <> " ") * (INS <> CHR$(34)) THEN 2210
2090 VTAB = VC : HTAB = HC : GOSUB 790 : PRINT ET$(HX)
2100 X = 0
2110 FOR IT = 1 TO 14 : IF INS = CIS(IT) THEN
2120 X = IT
2130 NEXT
2140 IF X = 14 THEN RETURN
2150 IF X = 0 THEN GOTO 2080
2160 ON X GOSUB 2350, 2310, 2460, 2520, 2580, 2620, 2660, 2750, 2890, 2970
2170 IF X = 11 THEN GOSUB 6430
2180 IF X = 12 THEN GOSUB 2410
2190 IF X = 13 THEN GOSUB 2810
2200 ON JP GOTO 2080, 2050, 510
2210 ET$(HX) = INS : HX = HX + 1
2220 IF HX > 15 THEN HX = 15
2230 GOTO 2080
2240 RETURN
2250 REM ** CONVERT STRING TO STRING ARR

```

```

22760 FOR IT=1 TO 20:ETS(IT)=MIDS(VDS(VK
22770 RETURN
22780 REM *** CONVERT STRING TO STRING AR
22790 VDS(VK)="":FOR IT=1 TO 20:VDS(VK)=
22800 VDS(VK)+ETS(IT):NEXT
23000 REM *** SAVE FROM VARIABLE DESCRIP
23100 OPTION FIELD
23200 GOSUB 2280
23300 GOSUB 2280
23400 REM *** MOVE TO EDIT FIELD **
23500 JP=2
23600 GOSUB 2280
23700 QS=3
23800 MQ=VK
23900 RETURN
24000 REM ***** TAB *****
24100 JP=2
24200 GOSUB 2280
24300 QS=1
24400 RETURN
24500 REM *** CURSOR UP *****
24600 JP=2
24700 GOSUB 2280
24800 IF VK=1 THEN VK=8:RETURN
24900 VK=VK-1
25000 RETURN
25100 REM *** CURSOR DOWN *****
25200 JP=2
25300 GOSUB 2280
25400 IF VK=8 THEN VK=1:RETURN
25500 VK=VK+1
25600 RETURN
25700 REM *** CURSOR LEFT ****
25800 JP=1
25900 IF HX>1 THEN HX=HX-1
26000 RETURN
26100 REM *** CURSOR RIGHT *****
26200 JP=1
26300 IF HX=15 THEN HX=1:RETURN
26400 HX=HX+1
26500 RETURN
26600 REM ***** INSERT SPACE *****
26700 JP=1
26800 IF HX=20 OR ETS(20)<>" " THEN RETURN
26900 FOR IT=19 TO HX STEP -1
27000 ETS(IT+1)=ETS(IT)
27100 VTAB=VE(VK,1):HTAB=IT+VE(VK,2):GOSUB
27200 B790:PRINT ETS(IT+1):NEXT
27300 ETS(HX)="":VTAB=VE(VK,1):HTAB=HX+V
27400 E(VK,2):GOSUB 790:PRINT ETS(HX);
27500 RETURN
27600 REM *** BLANK FIELD ****
27700 JP=1
27800 HX=1
27900 FOR IT=1 TO 20:ETS(IT)="":
28000 VTAB=VE(VK,1):HTAB=VE(VK,2)+IT:GOSUB
28100 B790:PRINT:;:NEXT
28200 RETURN
28300 REM ***** DELETE CHARACTER *****
28400 JP=1
28500 IF HX=1 THEN 2880
28600 FOR IT=HX-1 TO 19
28700 ETS(IT)=ETS(IT+1)
28800 VTAB=VE(VK,1):HTAB=VE(VK,2)+IT:GOSUB
28900 B790:PRINT ETS(IT):NEXT
29000 ETS(20)="":VTAB=VE(VK,1):HTAB=VE(V
29100 K,2)+20:GOSUB 790:PRINT:;:HX=HX-1
29200 RETURN
29300 REM ** SOLVE EQUATION FROM VARIABLE
29400 E DESCRIPTION FIELD *****
29500 JP=1
29600 GOSUB 2280
29700 IF MQ<>VK THEN MQ=VK:GOSUB 3240
29800 GOSUB 3910
29900 HX=1
30000 JF=2
30100 RETURN
30200 REM ***** DO PRINTOUT *****
30300 GOSUB 2280
30400 GOSUB 6060
30500 HX=1:JP=2
30600 RETURN
30700 REM *** EQUATION ENTRY *****
30800 GOSUB 3240
30900 HX=HS(MQ):VC=14
31000 IF QS=1 THEN RETURN
31100 HC=HS:GOSUB 6670
31200 IF (INS<>"")*(INS<"-")*(INS
31300 <<"")*(INS<>CHR$(34))TH
31400 EN3200
31500 VTAB=VC:HTAB=HC:GOSUB 790:PRINT EQS(
31600 HX)
31700 X=0
31800 FOR IT=1 TO 14:IF INS=CIS(IT) THEN X
31900 =IT
32000 NEXT
32100 IF X=14 THEN RETURN
32200 IF X=0 GOTO 3060
32300 ON X GOSUB 3300,3370,3410,3450,3490,
32400 3520,3550,3620,3730,3830
32500 IF X=11 THEN GOSUB 6430
32600 IF X=12 THEN GOSUB 3350
32700 IF X=13 THEN GOSUB 3670
32800 JF=3 GOTO 510

```



```

3190 GOTO 3050
3200 EQ$(HX)=INS:HX=HX+1
3210 IF HX>80 THEN HX=80
3220 GOTO 3060
3230 RETURN
3240 REM ***** PRINT EQUATION *****
3250 VTAB=14:HTAB=1:GOSUB 790:FOR IT=1 TO
80
3260 EQ$(IT)=MID$(MQ$(MQ),IT,1)
3270 PRINT EQ$(IT);:NEXT
3280 VX=0:GOSUB 890
3290 RETURN
3300 REM ***** EDIT CHANGE *****
3310 QS=1:HS(MQ)=HX
3320 GOSUB 3760
3330 GOSUB 1180
3340 RETURN
3350 REM ***** TAB *****
3360 IF HX>=1 AND HX<=20 THEN HX=21:RETU
RN
3370 IF HX>=21 AND HX<40 THEN HX=40:RETU
RN
3380 IF HX>=40 AND HX<=60 THEN HX=61:RET
URN
3390 IF HX>=61 AND HX<80 THEN HX=80:RETU
RN
3400 IF HX=80 THEN HX=1:RETURN
3410 REM ***** CURSOR UP *****
3420 IF HX<=40 THEN HX=HX+40: RETURN
3430 HX=HX-40
3440 RETURN
3450 REM ***** CURSOR DOWN *****
3460 IF HX>40 THEN HX=HX-40:RETURN
3470 HX=HX+40
3480 RETURN
3490 REM ***** CURSOR LEFT *****
3500 IF HX>1 THEN HX=HX-1
3510 RETURN
3520 REM ***** CURSOR RIGHT *****
3530 IF HX<80 THEN HX=HX+1
3540 RETURN
3550 REM ***** INSERT CHARACTER **
3560 IF HX=80 THEN RETURN
3570 VTAB=14:HTAB=80:GOSUB 790:FOR IT=79
TO HX STEP-1
3580 EQ$(IT+1)=EQ$(IT)
3590 PRINT EQ$(IT+1)"F2SHIFT CRSRLEFT";
:NEXT
3600 EQ$(HX)=" ":PRINT " ";
3610 RETURN
3620 REM ***** BLANK FIELD *****
3630 HX=1
3640 FOR IT=1 TO 80:EQ$(IT)="FSPACE"
3650 VTAB=14:HTAB=IT:GOSUB 790:PRINT "FSPA
CE";:NEXT
3660 RETURN
3670 REM ***** DELETE CHARACTER **
3680 IF HX=1 THEN 3720
3690 VTAB=14:HTAB=HX-1:GOSUB 790:FOR IT=H
X-1 TO 79:EQ$(IT)=EQ$(IT+1)
3700 PRINT "F2SHIFT CRSRLEFT";EQ$(IT)
:NEXT
3710 EQ$(80)=" ":HX=HX-1
3720 RETURN
3730 REM ***** EVALUATE EQUATION **
3740 GOSUB 3910
3750 K=9:RETURN
3760 REM ***** STRING ARRAY TO STRING
CONVERT *****
3770 HS(MQ)=HX
3780 MQ$(MQ)=" "
3790 FOR IT=1 TO 85
3800 IF EQ$(IT)=CHR$(160) THEN EQ$(IT)=CH
R$(32)
3810 MQ$(MQ)=MQ$(MQ)+EQ$(IT):NEXT
3820 RETURN
3830 REM ***** DO PRINTOUT *****
3840 GOSUB 3760
3850 GOSUB 6060
3860 RETURN
3870 REM ***** SAVE FROM EQUATION ENTRY
*****
3880 GOSUB 3760
3890 GOSUB 6250
3900 RETURN
3910 REM ***** PARSER AND SOLVER **
3920 VTAB=17:HTAB=1:GOSUB 790:GOSUB 850
3930 BH(0)=2:GOSUB 4290
3940 PT=0:SK%(1)=0:SK%(2)=0:FG(2)=0:SK(1)
=0
3950 PT=PT+1:GOSUB 4200:IF EQ$(PT)=" "AND
PT<=80 THEN 3950
3960 IF ASC(EQ$(PT))=160 THEN EQ$(PT)=CH
R$(32):GOTO 3950
3970 IF PT>80 THEN 4130
3980 IF EQ$(PT)=" "THEN VTAB=14:HTAB=PT:
GOSUB 790:PRINT " ";:GOTO 4130
3990 PC$="00"
4000 FOR IT=1 TO 7
4010 IF BH(IT)=1 THEN ON IT GOSUB 4660, 47
60, 4830, 4990, 4990, 5070, 5300
4020 IF PC$<>"00" THEN 4090
4030 NEXT IT
4040 JP=1
4050 GOSUB 4450
4060 HX=PT:IF HX>80 THEN HX=80
4070 VTAB=14:HTAB=HX:GOSUB 790:PRINT EQ$(
HX)
4080 RETURN
4090 JP=0
4100 ON VAL(LEFT$(PC$,1))GOSUB 5940, 5940,
5600, 5940, 5940, 5860, 5900
4110 IF JP=0 THEN 3950
4120 GOTO 4050
4130 IF SK%(1)>0 THEN JP=1:GOSUB 4450:RETU
RN
4140 VX=SK(1):GOSUB 890
4150 VTAB=VL(MQ,1):HTAB=VL(MQ,2):GOSUB 79
0:PRINT
4160 VTAB=VL(MQ,1):HTAB=VL(MQ,2):GOSUB 79
0
4170 VR(MQ)=VX:PRINT VK$(MQ)"=";:IF VR(M
Q)<0 THEN PRINT
4180 PRINT VR(MQ)
4190 RETURN
4200 REM ***** PARSER TRACKER *****
4210 IF PT>80 THEN PRINT "F3SHIFT CRSRLEFT
";:GOTO 4280
4220 VTAB=14:HTAB=PT:GOSUB 790:GOTO 4240
4230 VTAB=15:HTAB=PT-39:GOSUB 790
4240 FOR IT=0 TO 3
4250 IF IT=0 THEN PRINT "FCTRL RVSON"EQ$
(PT)"FCTRL RVSOFF"SHIFT CRSRLEFT"
";
4260 IF IT=3 THEN PRINT EQ$(PT);
4270 NEXT
4280 RETURN
4290 REM ***** STATE SEARCH DEFINITION **
*****
4300 FOR BX=1 TO 7:BH(BX)=0:NEXT
4310 ON BH(0)GOSUB 4320, 4330, 4340, 4330, 43
50, 4340, 4340:RETURN
4320 GOSUB 4370:RETURN
4330 GOSUB 4360:GOSUB 4370:GOSUB 4400:GOSUB
4410:GOSUB 4420:RETURN
4340 GOSUB 4380:GOSUB 4390:RETURN
4350 GOSUB 4360:GOSUB 4370:GOSUB 4410:GOSUB
4420:RETURN
4360 BH(1)=1:RETURN
4370 BH(2)=1:RETURN
4380 BH(3)=1:RETURN
4390 BH(4)=1:RETURN
4400 BH(5)=1:RETURN
4410 BH(6)=1:RETURN
4420 BH(7)=1:RETURN
4430 RETURN
4440 REM ***** SYNTAX ERROR PROMPT **
*****
4450 FOR SX=20 TO 23:HTAB=1:VTAB=SX:GOSU
B 790:GOSUB 850:NEXT
4460 VTAB=24:HTAB=1:GOSUB 790:FOR IT=1 TO
39:PRINT " ";:NEXT
4470 VTAB=21:HTAB=1:GOSUB 790
4480 ON JP GOSUB 4550, 4560, 4570, 4620, 4630
4490 VTAB=24:HTAB=1:GOSUB 790:PRINT "PRESS
ANY KEY TO CONTINUE";
4500 GET IN$:IF IN$=" "THEN 4500
4510 FOR SX=20 TO 23:VTAB=SX:HTAB=1:GOSU
B 790:GOSUB 850:NEXT
4520 VTAB=24:HTAB=1:GOSUB 790:FOR IT=1 TO
39:PRINT " ";:NEXT
4530 GOSUB 640
4540 RETURN
4550 PRINT "SYNTAX ERROR":RETURN
4560 PRINT "UNMATCHED PARENTHESIS":RETURN
4570 PRINT "OR + SIGN NOT FOLLOWED":RETURN
4580 PRINT "BY NUMBER OR VARIABLE":RETURN
4590 PRINT "BINARY OPERATOR DOES NOT HAVE
TWO"
4600 PRINT "NUMBERS OR VARIABLES":RETURN
4610 PRINT "FUNCTION DOES NOT HAVE AN ARG
UMENT":RETURN
4620 PRINT "DIVIDE BY ZERO ERROR":RETURN
4630 PRINT "ILLEGAL USE OF FUNCTION":RETU
RN
4640 RETURN
4650 REM ***** CHECK FOR FUNCTIONS **
*****
4660 CK$="":FOR CK=0 TO 2:CK$=CK$+EQ$(PT
+CK):NEXT
4670 FOR CK=1 TO 11
4680 IF LEFT$(CK$,LEN(FCS(CK)))=FCS(CK) T
HEN PC$="1"+CHR$(48+CK):GOTO 4710
4690 NEXT
4700 RETURN
4710 PT=PT+1:GOSUB 4200:PT=PT+1:GOSUB 4200
4720 BH(0)=1:GOSUB 4300
4730 RETURN
4740 REM ***** LEFT PARENTHESIS CHECK **
*****
4750 IF EQ$(PT)="(" THEN 4780
4760 IF EQ$(PT)="(" THEN 4780
4770 RETURN
4780 PC$="20"
4790 FG(2)=FG(2)+1
4800 BH(0)=2:GOSUB 4300
4810 RETURN
4820 REM ***** RIGHT PARENTHESIS CHECK **
*****
4830 IF EQ$(PT)=")" THEN 4850
4840 RETURN
4850 PC$="30"
4860 FG(2)=FG(2)-1
4870 BH(0)=3:GOSUB 4300
4880 RETURN
4890 REM ***** BINARY OPERATORS CHECK **
*****
4900 CK$="":FOR IT=0 TO 2:CK$=CK$+EQ$(PT
+IT):NEXT
4910 FOR CK=1 TO 5

```

Continued



```

4920 IF LEFT$(CK$, LEN(BN$(CK)))=BN$(CK) T
4930 HEN4950
4940 NEXT CK
4950 RETURN
4960 PC$="4"+STR$(CK)
4970 BH(0)=4:GOSUB4300
4980 REM
4990 ** UNARY OPERATORS CHECK **
5000 FOR CK=1 TO 2
5010 IF EQ$(PT)=UN$(CK) THEN 5030
5020 NEXT CK
5030 PC$="5"+STR$(CK)
5040 BH(0)=5:GOSUB4300
5050 RETURN
5060 REM
5070 ** NUMERIC CONSTANTS CHECK **
5080 FG(1)=0
5090 IF EQ$(PT)="." OR (EQ$(PT)>="0" AND
5100 EQ$(PT)<="9") THEN 5110
5110 PC$="6"
5120 IF EQ$(PT)=". " AND FG(1)=0 THEN FG(
5130 1)=1:PC$=PC$+EQ$(PT):GOTO5150
5140 IF EQ$(PT)= "+" AND EQ$(PT)<="9" TH
5150 EN PC$=PC$+EQ$(PT):GOTO5150
5160 IF EQ$(PT)= "-" THEN PC$=PC$+EQ$(PT)
5170 :GOTO5150
5180 PT=PT+1:GOTO5230
5190 PT=PT+1:GOSUB4200
5200 IF PT>80 THEN 52
5210 IF EQ$(PT)="-" OR EQ$(PT)="+" THEN 5
5220 FOR CK=1 TO 2
5230 IF EQ$(PT)>="0" AND EQ$(PT)<="9" TH
5240 EN 5270
5250 NEXT CK:PT=PT+1:GOTO5230
5260 BH(0)=6:GOSUB4300
5270 PC$="7"+STR$(CK)
5280 BH(0)=7:GOSUB4300
5290 RETURN
5300 REM
5310 ** ALPHABETIC VARIABLE CHE
5320 FOR CK=1 TO 16
5330 IF EQ$(PT)=VK$(CK) THEN 5340
5340 NEXT CK
5350 PC$="7"+CHR$(48+CK)
5360 BH(0)=7:GOSUB4300
5370 RETURN
5380 REM
5390 ** FUNCTION EVALUATE *
5400 IF SK$(2)=0 THEN JP=5:RETURN
5410 X=0:X=ASC(RIGHT$(SK$(SK%(1)),1))-4
5420 ONXGOSUB5450,5460,5470,5480,5490,55
5430 00,5510,5520,5530,5550,5560
5440 SK$(1)=SK%(1)-1
5450 IF SK$(1)=0 OR SK$(SK%(1))="20" THE
5460 N RETURN
5470 ON VAL (LEFT$(SK$(SK%(1)),1))-3 GOS
5480 UB5670,5810
5490 RETURN
5500 SK$(SK%(2))=SIN(SK$(SK%(2))):RETURN
5510 SK$(SK%(2))=COS(SK$(SK%(2))):RETURN
5520 SK$(SK%(2))=TAN(SK$(SK%(2))):RETURN
5530 SK$(SK%(2))=ATN(SK$(SK%(2))):RETURN
5540 SK$(SK%(2))=INT(SK$(SK%(2))):RETURN
5550 SK$(SK%(2))=RND(SK$(SK%(2))):RETURN
5560 SK$(SK%(2))=SGN(SK$(SK%(2))):RETURN
5570 SK$(SK%(2))=ABS(SK$(SK%(2))):RETURN
5580 IF SK$(SK%(2))<0 THEN JP=5:RETURN
5590 SK$(SK%(2))=SQR(SK$(SK%(2))):RETURN
5600 SK$(SK%(2))=EXP(SK$(SK%(2))):RETURN
5610 IF SK$(SK%(2))<0 THEN JP=5:RETURN
5620 SK$(SK%(2))=LOG(SK$(SK%(2))):RETURN
5630 RETURN
5640 REM
5650 ** RIGHT PARENTHESIS EV
5660 ALUATE
5670 IF SK$(1)=0 OR SK$(SK%(1))<>"20" TH
5680 EN JP=2:RETURN
5690 SK$(1)=SK%(1)-1
5700 IF SK$(1)>0 AND SK$(SK%(1))<>"20" T
5710 HEN5640
5720 RETURN
5730 ON VAL (LEFT$(SK$(SK%(1)),1))GOSUB538
5740 0,5380,5600,5670,5810,5810,5810
5750 RETURN
5760 REM
5770 ** BINARY OPERATORS EVALUAT
5780 E
5790 IF SK$(2)<1 THEN JP=4:RETURN
5800 ON VAL(RIGHT$(SK$(SK%(1)),1))GOSUB5
5810 710,5720,5730,5740,5780
5820 SK$(1)=SK%(1)-1:SK$(2)=SK%(2)-1
5830 RETURN
5840 SK$(SK%(2)-1)=SK(SK%(2)-1)+SK(SK%(2)
5850 ):RETURN
5860 SK$(SK%(2)-1)=SK(SK%(2)-1)-SK(SK%(2)
5870 ):RETURN
5880 SK$(SK%(2)-1)=SK(SK%(2)-1)*SK(SK%(2)
5890 ):RETURN
5900 SK$(SK%(2)-1)=SK(SK%(2)-1)/SK(SK%(2)
5910 ):RETURN

```

```

5740 IF SK$(SK%(2))<>0 THEN 5770
5750 JP=4:IF PT<80 THEN VTAB=14:HTAB=PT+
5760 1:GOSUB790:PRINT EQ$(PT+1)
5770 RETURN
5780 SK$(SK%(2)-1)=SK(SK%(2)-1)/SK(SK%(2)
5790 ):RETURN
5800 SK$(SK%(2)-1)=SK(SK%(2)-1)*SK(SK%(2)
5810 ):RETURN
5820 REM
5830 ** UNARY OPERATIONS EVALU
5840 ATE
5850 IF SK$(2)=0 THEN JP=3:RETURN
5860 IF SK$(SK%(1))="52" THEN SK$(SK%(2))
5870 =SK(SK%(2))
5880 SK$(1)=SK%(1)-1
5890 RETURN
5900 REM
5910 ** NUMERIC CONSTANTS EVAL
5920 UATE
5930 SK$=VAL(RIGHT$(PC$, (LEN(PC$)-1)))
5940 GOSUB6010
5950 RETURN
5960 REM
5970 ** VARIABLES EVALUATE
5980 SK$=VR(ASC(RIGHT$(PC$,1))-48)
5990 GOSUB6010
6000 RETURN
6010 REM
6020 ** PUSH TO OPERATOR STAC
6030 K
6040 SK$(1)=SK%(1)+1
6050 SK$(SK%(1))=PC$
6060 RETURN
6070 REM
6080 ** PUSH TO OPERAND STACK
6090 SK$(2)=SK%(2)+1
6100 SK$(SK%(2))=SK:RETURN
6110 REM
6120 ** COMMON NUMERIC HANDLIN
6130 G ROUTINES
6140 GOSUB5980
6150 IF SK$(1)=0 OR SK$(SK%(1))="20" THE
6160 N RETURN
6170 IF LEFT$(SK$(SK%(1)),1)="5" THEN GO
6180 SUB5810
6190 IF SK$(1)>0 AND LEFT$(SK$(SK%(1)),1)
6200 ="4" THEN GOSUB5670
6210 RETURN
6220 REM
6230 ** PRINTOUT
6240 PRINT"SHIFT CLR":VTAB=10:HTAB=15
6250 :GOSUB790:PRINT"MAKE SURE"
6260 VTAB=12:HTAB=15:GOSUB790:PRINT"PRIN
6270 TER IS"
6280 VTAB=14:HTAB=18:GOSUB790:PRINT"READ
6290 Y"
6300 VTAB=24:HTAB=1:GOSUB790:PRINT"PRESS
6310 ANY KEY TO CONTINUE"
6320 GET IN$:IF IN$=" " THEN 6110
6330 PRINT"SHIFT CLR":VTAB=10:HTAB=14
6340 :GOSUB790:PRINT"PLEASE WAIT"
6350 VTAB=12:HTAB=14:GOSUB790:PRINT"PRIN
6360 TING IN"
6370 VTAB=14:HTAB=16:GOSUB790:PRINT"PROG
6380 RESS"
6390 OPEN4,4
6400 PRINT#4,"IT FIGURES!"
6410 PRINT#4,"":PRINT#4,""
6420 FOR GI=1 TO 8
6430 PRINT#4,"VR(GI)"
6440 PRINT#4,VDS(GI)
6450 PRINT#4,MQ$(GI):NEXT
6460 PRINT#4,"":CLOSE4
6470 GOSUB510
6480 RETURN
6490 REM
6500 ** SAVE EQUATION ***
6510 INPUT"SHIFT CLR":2CRSRDOWN:2CRSR
6520 IGH:INPUT FILE NAME:NMS=NMS+LEFT$
6530 (NMS,16):IFNMS=" " THEN GOTO6350
6540 PRINT"CRSRDOWN:2CRSRRIGHT TAPE OR
6550 DISK (T/D)?"
6560 GET IN$:IF IN$=" " THEN 6280
6570 IF IN$="D" THEN PRINT"CRSRDOWN"
6580 ** SAVING FILE ** PLEASE WAIT ***:GOT
6590 O6330
6600 IF IN$="T" THEN 6320
6610 GOTO6270
6620 OPEN1,1,1,NMS:GOSUB6360:QS=1:JP=3:C
6630 LOSE1:RETURN
6640 OPEN15,8,15:PRINT#15,"S0:"+NMS
6650 OPEN1,8,8,0:"+NMS+",S,W:GOSUB6610
6660 :GOSUB6360
6670 QS=1:JP=3:RETURN
6680 FOR IT=1 TO 8:PRINT#1,VR(IT):NEXT
6690 FOR IT=1 TO 8:PRINT#1,VDS(IT):NEXT
6700 FOR IT=1 TO 8:PRINT#1,MQ$(IT):NEXT
6710 IF IN$="T" THEN RETURN
6720 GOSUB6610:CLOSE1:CLOSE15:RETURN
6730 VDS(IT)=VDS(0):RETURN
6740 MQ$(IT)=MQ$(0):RETURN
6750 REM
6760 ** LOAD EQUATION *****
6770 INPUT"SHIFT CLR":2CRSRDOWN:2CRSR
6780 IGH:ENTER FILE NAME:NMS=NMS+LEFT$
6790 (NMS,14):IFNMS=" " THEN 6530
6800 PRINT"CRSRDOWN:2CRSRRIGHT FROM TA
6810 PE OR DISK (T/D)?"
6820 GET IN$:IF IN$=" " THEN 6460
6830 IF IN$="D" THEN 6510
6840 IF IN$="T" THEN 6500
6850 GOTO6460
6860 OPEN1,1,0,NMS:GOSUB6540:QS=1:JP=3:C
6870 LOSE1:RETURN

```

Continued



```

6510 PRINT "CRSR DOWN" : **GETTING FILE*
6520 **PLEASE WAIT**
6530 OPEN 15,8,15:OPEN 1,8,8,"0":"+NMS+",S,
6540 R:GOSUB 6610:GOSUB 6540
6550 QS=1:JP=3:RETURN
6560 FOR IT=1 TO 8:INPUT#1,VR(IT):NEXT
6570 FOR IT=1 TO 8:INPUT#1,VDS(IT):IF VD
6580 $(IT) THEN GOSUB 6410
6590 NEXT
6600 IF IN$="T" THEN RETURN
6610 GOSUB 6610:CLOSE 1:CLOSE 15:RETURN
6620 REM **DISK ERROR ROUTINE**
6630 INPUT#15,EN,EM$,ET,ES:IF EN<20 THEN
6640 RETURN
6650 PRINT "SHIFT CLR":HTAB=1:VTAB=11:
6660 GOSUB 790:PRINT "DISK ERROR"EN:SHIFT
6670 CRSRLEFT,EM$,ET:SHIFT CRSRLEF
6680 T,ES
6690 HTAB=1:VTAB=24:GOSUB 790:PRINT "PRESS
6700 ANY KEY TO CONTINUE":
6710 GET IN$:IF IN$=" " THEN 6650

```

```

6660 CLOSE 1:CLOSE 15:QS=1:GOTO 220
6670 REM **GET CHARACTER**
6680 VTAB=VC:HTAB=HC:GOSUB 790
6690 FOR IT=0 TO 3
6700 IF IT<>0 THEN 6730
6710 IF (QS=1)OR(QS=2) THEN PRINT "CTRL RVS
6720 ON "ETS(HX)"SHIFT CRSRLEFT":GOTO
6730 PRINT "CTRL RVSON"EQ$(HX)"SHIFT C
6740 RSRLEFT":
6750 IF IT<>2 THEN 6760
6760 IF (QS=1)OR(QS=2) THEN PRINT "CTRL RVS
6770 OFF"ETS(HX)"SHIFT CRSRLEFT":GOT
6780 6760
6790 PRINT "CTRL RVSOFF"EQ$(HX)"SHIFT
6800 CRSRLEFT":
6810 GET IN$:IF IN$<>" " THEN IT=3
6820 NEXT IT
6830 IF IN$=" " THEN 6690
6840 IF (IN$<"Y")+(IN$>"1")+(IN$="")+(IN
6850 $="")+(IN$=CHR$(34)) THEN 6810
6860 VT=VC:HT=HC:GOSUB 790:PRINT "CTRL RV
6870 SOFF"IN$:
6880 PRINT "CTRL RVSOFF":RETURN

```

HCM

## IT FIGURES!

IBM PC &amp; IBM PCjr

```

100 *****
110 **IT FIGURES!
120 *****
130 COPYRIGHT 1985
140 EMERALD VALLEY PUBLISHING CO.
150 BY ROBERT PASCHKE
160 AND THE HCM STAFF
170 HOME COMPUTER MAGAZINE
180 VERSION 5.2.1
190 IBM PC: W/CARTRIDGE BASIC
200 FROM DOS 2.1 or
210 IBM PC W/BASICA
220
230 ON ERROR GOTO 2350
240
250 KEY OFF
260 TRUE=-1:FALSE=0:RANDOMIZE TIMER
270 CLS
280 FOR I=1 TO 7:KEY I,CHR$(I):NEXT I
290 :KEY 8,CHR$(18):KEY 9,CHR$(19)
300 CLS:LOCATE 12,15:PRINT "IT FIGURES!
310 :LOCATE 22,1:PRINT "PRESS";CHR$(1
320 7):CHR$(217):TO CONTINUE:GOSUB
330 2270
340 CLS:LOCATE 12,1:PRINT "INITIALIZING
350 :RESTORE 1850:FOR Z=1 TO 10:RE
360 AD A:TS=TS+CHR$(A):NEXT Z:VAR=1
370 K1$="":FOR Z=1 TO 9:READ A:K1$=K1$
380 +CHR$(A):NEXT Z
390 DIM VR(9),VDS(8),VRS(8),SK(84),SK$(
400 84),HL$(11)
410 RESTORE 1980:FOR Z=1 TO 11:READ HL
420 $(Z):NEXT Z
430 FUNC$="SIN COS TAN ATN INT RND SGN
440 ABS SQR EXP LOG":OP$="+-*/^":SN$=
450 "+"
460 DEF FN F(A$)=INT(INSTR(FN$,A$,1)/4)
470 +1
480 DEF FN O(A$)=INSTR(OP$,A$,1)
490 DEF FN EQ$(X)=MID$(VRS(VAR),X,1)
500 GOSUB 410:FOR Z=17 TO 23:LOCATE Z,1
510 :PRINT SPACES(40):NEXT
520 LOCATE 18,2:PRINT "ARE YOU SURE YOU
530 WANT TO":PRINT "EXIT THE PROGRAM
540 (Y/N)?":GOSUB 2270:IF A$="Y" OR A
550 $="N" OR A$=CHR$(13) THEN EP=0:GOSU
560 B 1870:DONE=FALSE:PV.A$="":GOTO 380
570 ELSE IF A$<>"Y" AND A$<>"N" THEN 3
580 90
590 CLS:END
600 CLS:GOSUB 2030:Y=3:X=2:GOSUB 2050:L
610 OCATE VAR+2,2,1
620 EP=1:VAR=1
630 IF EP<1 THEN EP=1
640 ON EP GOSUB 510,460,480:IF DONE THE
650 N RETURN ELSE EP=EP+1:IF EP>3 THEN
660 EP=1
670 GOTO 430
680 GOSUB 1880
690 IN$=STR$(VR(VAR)):GOSUB 2070:VR(VAR
700 )=VAL(IN$):LOCATE VAR+2,5:PRINT LEF
710 T$(STR$(VR(VAR))+SPACES(14),14):RE
720 TURN
730 GOSUB 1890
740 IN$=LEFT$(VDS(VAR)+SPACES(14),14):G
750 OSUB 2070:VDS(VAR)=IN$
760 LOCATE 2+VAR,21:PRINT VDS(VAR):RETU
770 RN
780 GOSUB 1870:LOCATE VAR+2,2,1,0,31
790 GOSUB 2270
800 IF A$=CHR$(13) THEN ND=0:RETURN
810 P=INSTR(K1$,RIGHT$(A$,1)):IF P=0 TH
820 EN GOTO 520 ELSE ON P GOTO 620,1420
830 ,560,550,600,580,1250,1300,2400
840 DONE=TRUE:RETURN

```

```

560 FOR Z=17 TO 23:LOCATE Z,1,0:PRINT S
570 PACES(40):NEXT:LOCATE 18,2:PRINT
580 SURE YOU WANT TO:PRINT "ERASE ALL
590 OF THE SCREEN (Y/N)?":PV.A$="":G
600 OSUB 2270:IF A$="Y" OR A$="N" THEN
610 EP=0:GOSUB 1870:RETURN ELSE IF A$<>
620 "Y" AND A$<>"N" THEN 560
630 FOR Z=1 TO 8:VR$(Z)="":VR(Z)=0:VDS(
640 Z)="":NEXT Z:EP=EP-1:GOSUB 410:GOSU
650 B 1870:RETURN
660 VAR=VAR-1:IF VAR<1 THEN VAR=8
670 ND=1:GOTO 510
680 VAR=VAR+1:IF VAR>8 THEN VAR=1
690 ND=1:GOTO 510
700 GOSUB 1840:ST=2:GOSUB 720:CP=1:PL=0
710 :DP=0:SK1=0:SK2=0:C.INIT=TRUE
720 EQ$=MID$(VRS(VAR),CP,1):IF CP=76 OR
730 EQ$=" " OR EQ$="9" THEN
740 710
750 IF NOT C.INIT THEN GOSUB 2280
760 ER=INT(CP/38)+1:EC=CP-((ER-1)*38):G
770 OSUB 2280:C.INIT=FALSE
780 PCS="00":FOR Q=1 TO 8:BH=VAL(MID$(
790 BH$,Q,1)):IF BH THEN ON Q GOSUB 770
800 ,780,790,800,810,820,910
810 IF PCS<>"00" THEN 690
820 NEXT Q:JP=1:GOSUB 1340:RETURN
830 JP=0:ON VAL(LEFT$(PCS,1)) GOSUB 920
840 ,920,940,920,920,970,980:IF JP=0 TH
850 EN CP=CP+1:GOTO 630
860 GOSUB 1340:RETURN
870 SOUND 1000,3:EP=0:IF SK1>0 THEN JP=
880 1:GOSUB 1340:RETURN ELSE VR(VAR)=SK
890 (1):GOSUB 1840:RETURN
900 ON ST GOTO 730,740,750,740,760,750,
910 750
920 BH$="01000000":RETURN
930 BH$="1100111":RETURN
940 BH$="0011000":RETURN
950 BH$="1100011":RETURN
960 P=INSTR(FUNC$,OP$)+MID$(VRS(VAR),CP,
970 3)+":IF P=0 THEN RETURN ELSE PCS
980 =1+CHR$(49+INT(P/4)):CP=CP+2:ST=1
990 :GOSUB 720:RETURN
1000 IF EQ$=" " THEN PCS="20":PL=PL+1:ST
1010 =2:GOSUB 720:RETURN ELSE RETURN
1020 IF EQ$="9" THEN PCS="30":PL=PL-1:ST
1030 =3:GOSUB 720:RETURN ELSE RETURN
1040 P=INSTR(OP$,EQ$):IF P=0 THEN RETURN
1050 ELSE PCS="4"+STR$(P):ST=4:GOSUB 72
1060 0:RETURN
1070 P=INSTR(SN$,EQ$):IF P=0 THEN RETURN
1080 ELSE PCS="5"+RIGHT$(STR$(P),1):ST=
1090 5:GOSUB 720:RETURN
1100 DP=0:N$=EQ$:IF N$<>" " AND (N$<="0"
1110 "OR N$>"9") THEN RETURN ELSE PCS="6
1120 "
1130 IF N$=" " AND DP=0 THEN DP=1:PCS=PC
1140 $+N$:GOTO 870
1150 IF N$>="0" AND N$<="9" THEN PCS=PCS
1160 +N$:GOTO 870
1170 IF N$="E" THEN PCS=PCS+N$:CP=CP+1:I
1180 F CP>76 THEN 900 ELSE N$=EQ$:DP=0:G
1190 OTO 880
1200 GOTO 900
1210 IF CP=76 THEN 900 ELSE CP=CP+1:IF C
1220 P>76 THEN 900 ELSE N$=MID$(VRS(VAR)
1230 ,CP,1):GOTO 830
1240 IF INSTR(SN$,EQ$)>0 THEN PCS=PCS+EQ
1250 $:CP=CP+1:IF CP>76 THEN 900 ELSE N$
1260 =EQ$
1270 GOTO 830
1280 ST=0:GOSUB 720:CP=CP-1:RETURN
1290 N$=EQ$:IF N$>="A" AND N$<="H" THEN
1300 PCS="7"+STR$(ASC(N$)-16):ST=7:GOSUB
1310 720:RETURN ELSE RETURN

```

Continued



```

9200 SK1=SK1+1:SK$(SK1)=PC$:RETURN
9300 SK2=SK2+1:SK$(SK2)=V:RETURN
9400 IF SK1=9 OR SK$(SK1)<>"20" THEN JP=
2:RETURN ELSE SK1=SK1-1
9500 IF SK1>0 AND SK$(SK1)<>"20" THEN ON
VAL(MID$(SK$(SK1),1,1))GOSUB 990,
990,940,1130,1200,1200,1200
9600 RETURN
9700 V=VAL(MID$(PC$,2)):GOSUB 1220:RETUR
N
9800 V=VR(VAL(MID$(PC$,2))-48):GOSUB 122
0:RETURN
9900 IF SK2=0 THEN JP=5:RETURN
10000 ON ASC(MID$(SK$(SK1),LEN(SK$(SK1)),
1)) - 48 GOSUB 1020,1030,1040,1050,10
60,1070,1080,1090,1100,1110,1120
10100 SK1=SK1-1:IF SK1=0 OR SK$(SK1)="20"
THEN RETURN ELSE ON VAL(MID$(SK$(SK
1),1,1)) - 3 GOSUB 1130,1200:RETURN
10200 SK$(SK2)=SIN(SK$(SK2)):RETURN
10300 SK$(SK2)=COS(SK$(SK2)):RETURN
10400 SK$(SK2)=TAN(SK$(SK2)):RETURN
10500 SK$(SK2)=ATN(SK$(SK2)):RETURN
10600 SK$(SK2)=INT(SK$(SK2)):RETURN
10700 SK$(SK2)=RND(1)*SK$(SK2):RETURN
10800 SK$(SK2)=SGN(SK$(SK2)):RETURN
10900 SK$(SK2)=ABS(SK$(SK2)):RETURN
11000 SK$(SK2)=SQR(SK$(SK2)):RETURN
11100 SK$(SK2)=EXP(SK$(SK2)):RETURN
11200 SK$(SK2)=LOG(SK$(SK2)):RETURN
11300 IF SK2<=1 THEN JP=4:RETURN
11400 ON VAL(RIGHT$(SK$(SK1),1)) GOSUB 11
50,1160,1170,1180,1190:SK1=SK1-1:SK
2=SK2-1:RETURN
11500 SK$(SK2-1)=SK$(SK2-1)+SK$(SK2):RETURN
11600 SK$(SK2-1)=SK$(SK2-1)-SK$(SK2):RETURN
11700 SK$(SK2-1)=SK$(SK2-1)*SK$(SK2):RETURN
11800 IF SK$(SK2)=0 THEN JP=4:RETURN ELSE
SK$(SK2-1)=SK$(SK2-1)/SK$(SK2):RETURN
11900 SK$(SK2-1)=SK$(SK2-1)*SK$(SK2):RETURN
12000 IF SK2=0 THEN JP=3:RETURN ELSE IF S
K$(SK1)="52" THEN SK$(SK2)=SK$(SK2)
K1=SK1-1:RETURN
12100 GOSUB 9300:IF SK1=0 OR SK$(SK1)="20"
THEN RETURN ELSE IF LEFT$(SK$(SK1),
1)="5" THEN GOSUB 1200
12200 IF SK1>0 AND MID$(SK$(SK1),1,1)="4"
THEN GOSUB 1130
12300 RETURN
12400 REM LOAD FILE ROUTINE
12500 FOR Z=17 TO 23:LOCATE Z,1:PRINT SPA
CES(39);CHR$(13);:NEXT:LOCATE 18,2:
PRINT "LOAD FILE:INPUT 'ENTER FILE
NAME:':F$:GOSUB 2290
12700 OPEN F$ FOR INPUT AS #1
12800 FOR Z=1 TO 8:INPUT #1,VR(Z):NEXT
Z:FOR Z=1 TO 8:INPUT #1,VD$(Z):NEXT
Z:FOR Z=1 TO 8:INPUT #1,VS(Z):NEX
T
12900 Z:CLOSE #1:EP=3:ERR.FLAG=TRUE:GOS
UB 2050:GOTO 1870
12900 ERR.FLAG=TRUE:GOTO 1870
13000 REM SAVE FILE ROUTINE
13100 FOR Z=17 TO 23:LOCATE Z,1:PRINT SPA
CES(39);CHR$(13);:NEXT:LOCATE 18,2:
PRINT "SAVE FILE:INPUT 'ENTER FIL
E NAME:':F$:GOSUB 2290
13200 OPEN F$ FOR OUTPUT AS #1
13300 FOR Z=1 TO 8:PRINT #1,VR(Z):NEXT
Z:FOR Z=1 TO 8:PRINT #1,VD$(Z):NEXT
Z:FOR Z=1 TO 8:PRINT #1,VS(Z):NEX
T
13400 Z:CLOSE #1:EP=3:ERR.FLAG=TRUE:GOT
O 1870
13400 FOR J=1 TO 7:SOUND 1000,1:LOCATE 16
+J,1:PRINT SPACES(40);:SOUND 800,1
:NEXT:ON JP GOSUB 1360,1370,1380,139
0,1400,1410
13500 LOCATE 18,2:PRINT "PRESS ";CHR$(17)
:CHR$(217);:TO CONTINUE:GOSUB
2270:ERR.FLAG=TRUE:EP=3:RETURN
13600 LOCATE 18,2:PRINT "?SYNTAX ERROR":R
ETURN
13700 LOCATE 18,2:PRINT "?UNMATCHED PAREN
THESIS":RETURN
13800 LOCATE 18,2:PRINT "? ' ' or ' ' NOT
FOLLOWED BY VARIABLE OR CONSTANT":
RETURN
13900 LOCATE 18,2:PRINT "?DIVIDE BY ZERO
ERROR":RETURN
14000 LOCATE 18,2:PRINT "?NO ARGUMENT IN
FUNCTION":RETURN
14100 LOCATE 18,2:PRINT "?OVERFLOW ERROR"
:RETURN
14200 GOSUB 1900:GOSUB 1840:ER=1:EC=1:CP=
1:LOCATE 15,2,1,0,31
14300 GOSUB 2280
14400 JS="":WHILE JS="":JS=INKEY$:WEND:K=
ASC(JS):IF K>31 AND K<127 THEN 1570
14500 IF LEN(JS)=2 THEN 1600
14600 IF K<>8 THEN 1490 ELSE IF CP=1 THEN
14300 ELSE IF CP=2 THEN VR$(VAR)=RI
GHT$(VR$(VAR),LEN(VR$(VAR))-1):GOSU
B 1700:GOSUB 1790:GOTO 1430
14700 IF CP>LEN(VR$(VAR)) THEN VR$(VAR)=L
EFT$(VR$(VAR),CP-2):GOSUB 1700:GOSU
B 1790:GOTO 1430
14800 VR$(VAR)=LEFT$(VR$(VAR),CP-2)+RIGH
T$(VR$(VAR),LEN(VR$(VAR))-(CP-1)):GO
SUB 1700:GOSUB 1790:GOTO 1430

```

```

14900 IF K=3 THEN VR$(VAR)=""CP=1:EC=1:E
R=1:LOCATE 14,1:PRINT SPACES(80);:G
OTO 1430
15000 IF K=13 THEN 1810
15100 IF K>96 AND K<123 THEN K=K-32
15200 PRINT CHR$(K);
15300 IF CP>LEN(VR$(VAR)) THEN VR$(VAR)=M
IDS(VR$(VAR))+SPACES(76),1,CP)
15400 IF CP=1 THEN VR$(VAR)=CHR$(K)+MID$(
VR$(VAR),2,LEN(VR$(VAR))-1)
15500 IF CP=LEN(VR$(VAR)) THEN VR$(VAR)=M
IDS(VR$(VAR),1,LEN(VR$(VAR))-1)+CHR
$(K):GOTO 1580
15600 VR$(VAR)=MID$(VR$(VAR),1,CP-1)+CHR
$(K)+MID$(VR$(VAR),CP+1,LEN(VR$(VAR)
)-CP)
15700 VR$(VAR)=LEFT$(VR$(VAR),84)
15800 GOSUB 2280:EC=EC+1:CP=CP+1:IF EC>38
THEN EC=1:ER=ER+1:IF ER>2 THEN ER=
1:CP=1
15900 GOTO 1430
16000 CP=(ER-1)*38+EC:P=INSTR("SRKMPH",RI
GHT$(JS,1)):IF P>0 THEN ON P GOSUB
1620,1650,1700,1720,1740,1760:GOTO
1430 ELSE 1440
16100 REM
16200 IF (CP=1) AND (LEN(VR$(VAR))>1) THE
N VR$(VAR)=MID$(VR$(VAR),2,LEN(VR$(
VAR))-1):GOTO 1690 ELSE IF SP=1 THE
N GOTO 1790
16300 IF CP>LEN(VR$(VAR)) THEN RETURN
16400 VR$(VAR)=MID$(VR$(VAR),1,CP-1)+MID
$(VR$(VAR),CP+1,LEN(VR$(VAR))-CP):GO
TO 1790
16500 GOSUB 2280:IF CP>LEN(VR$(VAR)) OR L
EN(VR$(VAR))>76 THEN RETURN
16600 IF CP=1 THEN VR$(VAR)="+VR$(VAR)
VR$(VAR)=MID$(VR$(VAR),CP,LEN(VR$(V
AR))-CP+1):GOTO 1790
16700 VR$(VAR)=LEFT$(VR$(VAR),CP-1)+
IDS(VR$(VAR),CP,LEN(VR$(VAR))-CP+1)
:GOTO 1790
16800 VR$(VAR)=""ER=1:EC=1:RETURN
16900 GOSUB 1790:GOSUB 1800:RETURN
17000 GOSUB 2280:EC=EC+1:CP=CP+1:IF EC<1
THEN EC=38:ER=ER-1:IF ER<1 THEN ER=
3:CP=84
17100 GOTO 1770
17200 GOSUB 2280:EC=EC+1:CP=CP+1:IF EC>38
THEN EC=1:ER=ER+1:IF ER>3 THEN ER=
1:CP=1
17300 GOTO 1770
17400 GOSUB 2280:ER=ER+1:CP=CP+38:IF ER>2
THEN ER=1:CP=CP-38
17500 GOTO 1770
17600 GOSUB 2280:ER=ER-1:CP=CP-38:IF ER<1
THEN ER=2:CP=CP+38
17700 IF CP>LEN(VR$(VAR)) THEN VR$(VAR)=V
R$(VAR)+
17800 RETURN
17900 LOCATE 14,2,0:PRINT LEFT$(VR$(VAR)+
SPACES(38),38);:LOCATE 15,2:PRINT M
IDS(VR$(VAR))+SPACES(38),39,38):RETU
RN
18000 REM
18100 P=INSTR(VR$(VAR),""):IF P=0 THEN E
P=EP-1:SW=FALSE:RETURN ELSE IF P=1
AND LEN(VR$(VAR))=1 THEN VR$(VAR)=
"":GOTO 1810
18200 IF P=LEN(VR$(VAR)) THEN VR$(VAR)=LE
FT$(VR$(VAR),LEN(VR$(VAR))-1):GOTO
1800
18300 CP=P:GOSUB 1610:GOTO 1800
18400 LOCATE 12,2:PRINT SPACES(17);:LOCAT
E 12,2:PRINT CHR$(64+VAR);:VR(V
AR):GOSUB 1790:LOCATE 2+VAR,5:PRINT VR
$(VAR):RETURN
18500 DATA 18,2,3,4,5,13,6,7,18,19
18600 DATA 7,18,3,19,80,72,5,6,4
18700 RESTORE 1900:GOTO 1910
18800 RESTORE 2000:GOTO 1910
18900 RESTORE 2010:GOTO 1910
19000 RESTORE 2020:GOTO 1910
19100 READ AS:IF AS=PV.AS AND ERR.FLAG=FA
LSE THEN RETURN
19200 ERR.FLAG=FALSE
19300 FOR Z=17 TO 23:LOCATE Z,1,0:PRINT S
PACES(40);:NEXT
19400 LOCATE 17,INT((40-LEN(AS))/2):PRINT
AS:FOR Z=19 TO 23:READ A
19500 IF A<>0 THEN LOCATE Z,2,0:PRINT HL$
(A);
19600 NEXT Z:PV.AS=AS:IF LEFT$(AS,1)="S"
THEN LOCATE 19,29:PRINT "F6 - SAVE"
:LOCATE 20,29:PRINT "F4 - PRINT"
:LOCATE 24,1:PRINT "CHR$(17);CHR$
(217);:TAB TO NEXT FIELD - ACCE
PT INPUT":LOCATE 1,1,1,1,7:RETURN
19800 DATA DEL - DELETE,INS - INSERT,F3
- ERASE LINE,F7 - CALCULATE EQUATI
ON,F8 - EDIT EXPRESSION,F9 - EXIT
,F3 - CLEAR ENTRIES,F5 - LOAD,F6
- SAVE,BKSP - BACK SPACE,F4 - PRIN
TOUT
19900 DATA SELECT VARIABLE,4,5,6,7,8,11
20000 DATA EDIT VALUE,1,10,2,3,0,0
20100 DATA EDIT DESCRIPTION,1,10,2,3,0,0

```

Continued



```

2020 DATA EDIT EXPRESSION,1,2,3,10,0,0
2030 REM DRAW SCREEN
2040 FOR Z=2 TO 10:LOCATE Z,20,0:PRINT C
    HRS(179):NEXT:LOCATE 11,1:PRINT ST
    RINGS(19,196):CHRS(193):STRINGS(20,
    196):LOCATE 13,1:PRINT STRINGS(40,1
    96):LOCATE 16,1:PRINT STRINGS(40,2
    05):LOCATE 1,15:PRINT "IT FIGURES!"

2050 REM DISPLAY VARIABLE INFO
2060 FOR I=1 TO 8:LOCATE I+2,2:PRINT C
    HRS(64+I):SPACES(ABS(VR(I)<0)):
    VR(I):LOCATE I+2,21:PRINT VDS(I):
    NEXT I:LOCATE 12,2:PRINT CHRS(64+VA
    R):SPACES(ABS(VR(I)<0)):VR(VAR
    ):RETURN
2070 CP=1:INS=LEFT$(INS+SPACES(14),14):L
    OCATE VAR+2,(EP-2)*16+5,1
2080 IF EP=2 AND VR(VAR)>=0 THEN LOCATE
    VAR+2,6,1:CP=2
2090 KS=INKEY$:IF KS="" THEN 2090 ELSE I
    F LEN(KS)>1 THEN 2200 ELSE IF ASC(K
    S)=19 OR ASC(KS)=13 THEN RETURN
2100 IF ASC(KS)<>8 THEN 2140
2110 IF CP=1 THEN 2090 ELSE IF CP=2 THEN
    INS=RIGHT$(INS,13)+":CP=CP-1:GOT
    O 2250
2120 IF CP=14 THEN INS=LEFT$(INS,13)+
    ":CP=CP-1:GOTO 2250
2130 INS=LEFT$(INS,CP-1)+RIGHT$(INS,LEN(
    INS)-(CP-1)):CP=CP-1:GOTO 2250
2140 IF ASC(KS)=3 THEN INS=SPACES(14):CP
    =1:GOTO 2250
2150 IF ASC(KS)<32 OR ASC(KS)>128 THEN S
    OUND 110,2:GOTO 2090
2160 IF EP=2 THEN P=INSTR("+-.0123456789
    E"):IF P=0 THEN SOUND 110,2:GOT
    O 2090
2170 IF CP=1 THEN INS=KS+RIGHT$(INS,LEN(
    INS)-1):PRINT KS:CP=CP+1:GOTO 2090
2180 IF CP=14 THEN INS=LEFT$(INS,LEN(INS
    )-1)+KS:PRINT KS:LOCATE VAR+2,(EP-
    2)*16+4+CP,1:GOTO 2090
2190 INS=LEFT$(INS,CP-1)+KS+RIGHT$(INS,L
    EN(INS)-CP):CP=CP+1:PRINT KS:GOTO
    2090
2200 KS=RIGHT$(KS,1):P=INSTR("KMRS",KS):
    IF P=0 THEN SOUND 110,2:GOTO 2090 E
    LSE ON P GOTO 2210,2220,2240,2260
2210 IF CP=1 THEN 2090 ELSE CP=CP-1:GOTO
    2230

```

```

2220 IF CP=14 THEN 2090 ELSE CP=CP+1
2230 LOCATE VAR+2,(EP-2)*16+4+CP,1:GOTO
    2090
2240 INS=LEFT$(INS,CP-1)+CHRS(32)+RIGHT$(
    INS,LEN(INS)-(CP-1)):INS=LEFT$(INS
    +SPACES(14),14)
2250 LOCATE VAR+2,(EP-2)*16+5:PRINT INS:
    GOTO 2230
2260 INS=LEFT$(INS,CP-1)+RIGHT$(INS,LEN(
    INS)-CP):INS=LEFT$(INS+SPACES(14),1
    4):GOTO 2250
2270 AS=INKEY$:IF AS="" THEN 2270 ELSE R
    ETURN
2280 IF ER=1 THEN LOCATE 14,EC+1,1:RETUR
    N ELSE LOCATE 15,EC+1,1:RETURN
2290 I=INSTR(FS,""):IF I=0 THEN 2310
2300 IF I>1 THEN FS=LEFT$(FS,I-1)
2310 IF LEN(FS)>9 THEN FS=LEFT$(FS,9)
2320 FOR I=1 TO LEN(FS):XS=MIDS(FS,I,1
    ):IF XS="a" AND XS<="z" THEN MIDS(
    FS,I,1)=CHRS(ASC(XS)-32)
2330 NEXT I
2340 FS=FS+".FIG":RETURN
2350 RESTORE 2380:CLS:FOR Z=1 TO 20:READ
    A,AS:IF A=ERR THEN LOCATE 12,1:PRI
    NT AS:GOTO 2370
2360 NEXT:LOCATE 12,1:PRINT "ERROR";ERR;
    IN LINE:ERL
2370 LOCATE 22,1:PRINT "PRESS ";CHRS(17)
    :CHRS(217):TO CONTINUE":GOSUB 227
    0:PV,AS=RESUME 410
2380 DATA 5,ILLEGAL FUNCTION USED,6,NUMB
    ER OUT OF RANGE-overflow,11,DIVI
    SION BY ZERO ATTEMPTED,24,DEVICE TI
    ME OUT,25,DEVICE FAILURE,27,OUT OF
    PAPER,53,FILE NOT FOUND,54,BAD FILE
    MODE,55,FILE ALREADY OPEN,57,DEVIC
    E I/O ERROR,61,DISK IS FULL
2390 DATA 64,BAD FILE NAME,67,TOO MANY F
    ILES,BAD FILE NAME,68,DEVICE NOT
    AVAILABLE,69,COMMUNICATIONS OVERFL
    OW,70,DISK IS WRITE PROTECTED,71,DI
    SK IS NOT READY,72,DISK MEDIA ERROR
    ,75,PATH FILE ACCESS ERROR,76,CAN'T
    FIND THAT PATH
2400 FOR Z=1 TO 8:LPRINT CHRS(64+Z):"
    :VR(Z):TAB(25):VDS(Z):LPRINT VR$(Z)
    :LPRINT:NEXT:EP=0:RETURN

```

HOM

## IT FIGURES!

TI-99/4A

```

1100 I *****
1110 I * IT FIGURES! *
1120 I *****
1130 I COPYRIGHT 1985
1140 I EMERALD VALLEY PUBLISHING CO.
1150 I BY ROBERT PASCHELKE
1160 I AND THE HCM STAFF
1170 I HOME COMPUTER MAGAZINE
1180 I VERSION 5.2.1
1190 I TI EXTENDED BASIC
1200 I TITLE SCREEN
1210 I

2220 DISPLAY AT(12,9)ERASE ALL:"IT FIGUR
    ES!":DISPLAY AT(22,3):"PRESS ENT
    ER TO CONTINUE":GOSUB 1710
2230 DISPLAY AT(12,1)ERASE ALL:"INITIALI
    ZING":RESTORE 1730:KS="":
    FOR Z=1 TO 10:READ A:KS=KS&
    CHRS(A):NEXT Z:VAR=1
2240 K1$=K1$&CHRS(A):NEXT Z:READ A:
    K1$=K1$&CHRS(A):NEXT Z
2250 DIM VR(8),VDS(8),VRS(8),SK(84),SKS(
    84),HLS(11)
2260 FNS=SIN COS TAN ATN INT RND SGN A
    BS SQR EXP LOG:OPS="+-*/^":
    SNS="+
2270 DEF F(A$)=INT(POS(FNS,A$,1)/4)+1
2280 DEF O(A$)=POS(OPS,A$,1)
2290 DEF EQ$(X)=SEG$(VRS(VAR),X,1)
2300 CALL CHAR(136,FFFFFFFFFFFFFFFF)
2310 GOSUB 330:GOSUB 360:DISPLAY A
    T(12,1)ERASE ALL:"ARE YOU SURE YOU
    WANT TO EXIT THE PROGRAM NOW (Y/
    N)?"
2320 ACCEPT AT(14,1)VALIDATE("YN")SIZE(1
    ):AS=:IF AS="N" THEN 310 ELSE CAL
    L CLEAR:END
2330 ON ERROR 1890:CALL CLEAR:CALL
    SCREEN(2):FOR Z=1 TO 14:CALL
    COLOR(Z,5,15):NEXT Z:CALL VCHA
    R(1,31,136,96)
2340 DISPLAY AT(1,9):"IT FIGURES!":FO
    R Z=1 TO 8:DISPLAY AT(Z+1,1):CHR
    $(64+Z):VR(Z):TAB(16):VDS(Z)
    :NEXT Z
2350 CALL HCHAR(10,1,136,288):GOSUB 17
    20:RESTORE 1810:FOR Z=1 TO 11
    :READ HLS(Z):NEXT Z:RETURN
2360 EP=1:VAR=1
2370 ON EP GOSUB 450,390,430:IF DONE
    THEN RETURN ELSE EP=EP+1:IF EP>3
    GOTO 370
2380 GOTO 370
2390 GOSUB 1760

```

```

4000 ACCEPT AT(1+VAR,4)SIZE(-12)VALIDATE
    (NUMERIC):AS=:CALL KEY(0,K,S):IF
    K=13 OR S=0 THEN CALL SOUND(1,880
    ,0)ELSE GOTO 400
410 IF AS="" THEN AS="0"
420 VR(VAR)=VAL(AS):DISPLAY AT(VAR+1,
    3)SIZE(13):VR(VAR):RETURN
430 GOSUB 1770
440 ACCEPT AT(VAR+1,17)SIZE(-14):VDS(VA
    R):CALL KEY(0,K,S):IF K=13 OR S
    =0 THEN CALL SOUND(1,880,0):RETUR
    N ELSE GOTO 440
450 CALL SPRITE(1,136,2,VAR*8,17):IF
    ND=0 THEN ND=1:GOSUB 1750
460 CALL KEY(0,K,S):C=ABS(C-1):IF S
    =0 THEN CALL COLOR(1,C+1):GOTO 4
    60 ELSE CALL SOUND(1,880,0):CALL
    COLOR(1,1)
470 IF K=13 THEN ND=0:RETURN
480 DONE=0:ND=0:P=POS(K1$,CHRS(K
    ,1)):IF P=0 THEN 460 ELSE ON P GOT
    O 570,1350,500,490,550,530,1200,124
    0,1870
490 DONE=1:RETURN
500 FOR Z=19 TO 24:CALL HCHAR(Z,3,32
    ,28):NEXT Z:DISPLAY AT(20,1):
    ARE YOU SURE YOU WANT TO ERASE A
    LL OF THE SCREEN (Y/N)?"
510 ACCEPT AT(22,7)SIZE(1)VALIDATE("YNY
    n"):AS=:IF AS="O" OR AS="N" THEN
    EP=0:GOSUB 1750:RETURN
520 FOR Z=1 TO 8:VRS(Z)=VR(Z):VR(Z)=
    0:VDS(Z)=:CALL HCHAR(Z+1,19
    ,32,12):NEXT Z:EP=EP-1:GOSU
    B 340
530 VAR=VAR-1:IF VAR<1 THEN VAR=8
540 ND=1:GOTO 450
550 VAR=VAR+1:IF VAR>8 THEN VAR=1
560 ND=1:GOTO 450
570 GOSUB 1720:ST=2:GOSUB 680:
    CP=1:PL=0:DP=0:SK1=0:SK
    2=0
580 CALL SPRITE(1,136,2,97,17):C=0
590 IF CP=84 OR EQ$(CP)=OR EQ$(CP)=
    " " OR EQ$(CP)= THEN 600
600 CPX=INT(CP/28):CPY=CP-CPX*28:C
    ALL LOCATE(1,97+CPX*8,17+CPY*8)
610 PC$="":FOR Z=1 TO 7:BH=VAL(
    SEG$(BH$,Z,1)):IF BH=1 THEN ON Z
    GOSUB 730,740,750,760,770,780,860
620 C=ABS(C-1):CALL COLOR(1,C+1):I
    F PC$<>"" THEN 640

```

Continued



```

630 NEXT Z :: IF RETURN JP=0 THEN GOSUB 1280 :: GOSUB
640 JP=0 :: ON VAL (SEG$(PC$,1,1)) GOSUB
870,870,890,870,870,920,930 :: IF J
P=0 THEN CP=1 :: GOTO 590
650 IF JP=0 THEN JP=1 :: GOSUB 1280 :: RETURN
660 RETURN :: ELSE CALL GOSUB 1280 :: RETURN
670 IF SK1>0 THEN IF JP=0 THEN JP=1 ::
GOSUB 1280 :: RETURN ELSE VR(VAR)=SK(1) :: GO
SUB 1720 :: RETURN
680 ON ST GOTO 690,700,710,700,720,710,
710
690 BHS="01000000" :: RETURN
700 BHS="11000111" :: RETURN
710 BHS="00110000" :: RETURN
720 BHS="11000011" :: RETURN
730 P=POS(FNS) :: &SEG$(VR$(VAR),CP,3)&
="1" :: IF P=0 THEN RETURN ELSE PC$
="1" & CHR$(49+(INT(P/4))) :: CP=CP+2
740 IF EQ$(CP) :: GOSUB 680 :: RETURN
PL+1 :: ST=2 :: GOSUB 680 :: RETURN
750 IF EQ$(CP) :: THEN PC$="30" :: PL=
PL-1 :: ST=3 :: GOSUB 680 :: RETURN
760 P=POS(OP$,EQ$(CP),1) :: IF P=0 THEN
RETURN ELSE PC$="4" & STR$(P) :: ST=4
770 P=POS(SN$,EQ$(CP),1) :: IF P=0 THEN
RETURN ELSE PC$="5" & STR$(P) :: ST=5
780 DP=0 :: NS=EQ$(CP) :: IF NS<>" " AND
(NS<="0" OR NS>="9") THEN RETURN ELSE
PC$="6"
790 IF NS<=" " AND DP=0 THEN DP=1 :: PC$
=PC$&NS :: AND GOTO 820
800 IF NS>=" " AND NS<="9" THEN PC$=PC$
&NS :: GOTO 820
810 IF NS="E" THEN PC$=PC$&NS :: CP=CP+
1 :: IF CP>84 THEN GOTO 830 ELSE EQ$(
CP) :: DP=0 :: GOTO 830
820 IF CP>84 THEN 850 ELSE CP=CP+1 :: I
F CP>84 THEN 850 ELSE NS=EQ$(CP) ::
GOTO 790
830 IF POS(SN$,EQ$(CP),1)>0 THEN PC$=PC
$&EQ$(CP) :: CP=CP+1 :: IF CP>84 THE
N GOTO 790
840 ST=6 :: GOSUB 680 :: CP=CP-1 :: RET
URN
850 NS=EQ$(CP) :: IF NS>="A" AND NS<="H"
THEN PC$="7" & STR$(ASC(NS)-16) :: ST
=7 :: GOSUB 680 :: RETURN ELSE RETU
RN
870 SK1=SK1+1 :: SK$(SK1)=PC$ :: RETURN
880 SK2=SK2+1 :: SK$(SK2)=V$ :: RETURN
890 IF SK1=0 OR SK$(SK1)<>"20" THEN JP=
2 :: RETURN ELSE SK1=SK1-1
900 IF SK1>0 AND SK$(SK1)<>"20" THEN ON
VAL (SEG$(SK$(SK1),1,1)) GOSUB 940,9
40,890,1080,1150,1150,1150
910 RETURN
920 V=VAL (SEG$(PC$,2,LEN(PC$)-1)) :: GOS
UB 1170 :: RETURN
930 V=VR (VAL (SEG$(PC$,2,LEN(PC$)-1))-48
:: GOSUB 1170 :: RETURN
940 IF SK2=0 THEN JP=5 :: RETURN
950 ON ASC (SEG$(SK$(SK1),1,1))-48 GOSUB
970,980,990,1000,1010,
1020,1030,1040,1050,1060,1070
960 SK1=SK1-1 :: IF SK1=0 OR SK$(SK1)=
"20" THEN RETURN ELSE ON VAL (SEG$(SK
$(SK1),1,1))-3 GOSUB 1080,1150 :: R
ETURN
970 SK$(SK2)=SIN (SK$(SK2)) :: RETURN
980 SK$(SK2)=COS (SK$(SK2)) :: RETURN
990 SK$(SK2)=TAN (SK$(SK2)) :: RETURN
1000 SK$(SK2)=ATN (SK$(SK2)) :: RETURN
1010 SK$(SK2)=INT (SK$(SK2)) :: RETURN
1020 SK$(SK2)=RND (SK$(SK2)) :: RETURN
1030 SK$(SK2)=SGN (SK$(SK2)) :: RETURN
1040 SK$(SK2)=ABS (SK$(SK2)) :: RETURN
1050 SK$(SK2)=SQR (SK$(SK2)) :: RETURN
1060 SK$(SK2)=EXP (SK$(SK2)) :: RETURN
1070 SK$(SK2)=LOG (SK$(SK2)) :: RETURN
1080 IF SK2<=1 THEN JP=4 :: RETURN
1090 ON VAL (SEG$(SK$(SK1),1,1))-48 GOSUB
1100,1110,1120,1130,1140 ::
SK1=SK1-1 :: SK2=SK2-1 :: RETURN
1100 SK$(SK2-1)=SK$(SK2-1)+SK$(SK2) :: RETUR
N
1110 SK$(SK2-1)=SK$(SK2-1)-SK$(SK2) :: RETUR
N
1120 SK$(SK2-1)=SK$(SK2-1)*SK$(SK2) :: RETUR
N
1130 IF SK$(SK2)=0 THEN JP=4 :: RETURN EL
SE SK$(SK2-1)=SK$(SK2-1)/SK$(SK2) :: RE
TURN
1140 SK$(SK2-1)=SK$(SK2-1)^SK$(SK2) :: RETUR
N
1150 IF SK2=0 THEN JP=3 :: RETURN ELSE I
F SK$(SK1)="52" THEN SK$(SK2)=-SK$(SK
2)
1160 SK1=SK1-1 :: RETURN

```

```

1170 GOSUB 880 :: IF SK1=0 OR SK$(SK1)=
"20" THEN RETURN ELSE IF SEG$(SK$(SK
1),1,1)="5" THEN GOSUB 1150
1180 IF SK1>0 AND SEG$(SK$(SK1),1,1)="4"
THEN GOSUB 1080
1190 RETURN
1200 FOR Z=18 TO 24 :: CALL HCHAR(Z,3,32
,28) :: NEXT Z :: DISPLAY AT(19,9):"
LOAD FILE";
1210 DISPLAY AT(22,1):"ENTER DEVICE & FI
LE NAME:" :: ACCEPT AT(23,1) SIZE(15
):FS :: IF FS="" THEN EP=0 :: RETUR
N
1220 OPEN #1:FS,INTERNAL,INPUT,FIXED 10
4
1230 FOR Z=1 TO 8 :: INPUT #1:VR(Z),VD$(
Z),VR$(Z) :: NEXT Z :: CLOSE #1 :: E
P=0 :: GOSUB 330 :: RETURN
1240 FOR Z=18 TO 24 :: CALL HCHAR(Z,3,32
,28) :: NEXT Z :: DISPLAY AT(19,9):"
SAVE FILE";
1250 DISPLAY AT(22,1):"ENTER DEVICE & FI
LE NAME:" :: ACCEPT AT(23,1) SIZE(15
):FS :: IF FS="" THEN EP=0 :: RETUR
N
1260 OPEN #1:FS,INTERNAL,OUTPUT,FIXED 10
4
1270 FOR Z=1 TO 8 :: PRINT #1:VR(Z),VD$(
Z),VR$(Z) :: NEXT Z :: CLOSE #1 :: E
P=0 :: IF FS="CS1" THEN GOSUB 330 ::
RETURN ELSE RETURN
1280 CALL SOUND(200,110,0) :: FOR Z=19 TO
24 :: CALL HCHAR(Z,3,32,28) :: NEXT
Z :: ON JP GOSUB 1300,1310,1320,13
30,1340
1290 DISPLAY AT(24,1):"PRESS ENTER TO CO
NTINUE" :: GOSUB 1710 :: EP=0 :: RE
TURN
1300 DISPLAY AT(22,1):"SYNTAX ERROR" ::
RETURN
1310 DISPLAY AT(22,1):"UNMATCHED PARENTH
ESIS" :: RETURN
1320 DISPLAY AT(22,1):"-' OR '+' SIGN N
OT FOLLOWED BY NUMBER OR VARIABLE" ::
RETURN
1330 DISPLAY AT(22,1):"DIVIDE BY ZERO ER
ROR" :: RETURN
1340 DISPLAY AT(22,1):"NO ARGUMENT IN FU
NCTION" :: RETURN
1350 GOSUB 1780 :: GOSUB 1720 :: ER=1 ::
EC=1 :: CP=1
1360 CALL SPRITE(#1,136,2,97+(ER-1)*8,17
+(EC-1)*8)
1370 CALL KEY(0,K,S) :: C=ABS(C-1) :: CALL
COLOR(#1,C+1) :: IF S=0 THEN 1370 E
LSE CALL COLOR(#1,1) :: CALL SOUND(1
,880,0)
1380 IF K=13 THEN 1680 ELSE P=POS(K$,CHR
$(K),1) :: IF P>0 THEN 1470 ELSE IF
K<32 OR K>128 THEN 1370
1390 CALL HCHAR(12+ER,2+EC,K)
1400 IF CP>LEN(VR$(VAR)) THEN VR$(VAR)=SE
G$(VR$(VAR)&RPT$(CHR$(32),84),1,CP)
1410 IF CP=1 THEN VR$(VAR)=CHR$(K)&SEG$(
VR$(VAR),2,LEN(VR$(VAR))-1)
1420 IF CP=LEN(VR$(VAR)) THEN VR$(VAR)=SE
G$(VR$(VAR),1,LEN(VR$(VAR))-1)&CHR$(
K) :: GOTO 1450
1430 VR$(VAR)=SEG$(VR$(VAR),1,CP-1)&CHR$(
K)&SEG$(VR$(VAR),CP+1,LEN(VR$(VAR))
-CP)
1440 VR$(VAR)=SEG$(VR$(VAR),1,84)
1450 EC=EC+1 :: CP=CP+1 :: IF EC>28 THEN
EC=1 :: ER=ER+1 :: IF ER>3 THEN ER
=1 :: CP=1
1460 GOTO 1360
1470 CP=(ER-1)*28+EC :: ON P GOSUB 1500,
1530,1560,1170,1370,1490,1580,1600,
1620,1640
1480 IF P=6 THEN 1680 ELSE 1360
1490 RETURN
1500 IF (CP=1)*(LEN(VR$(VAR))>1) THEN VR$(
VAR)=SEG$(VR$(VAR),2,LEN(VR$(VAR))
-1) :: GOTO 1570 ELSE IF SP=1 THEN 1
560
1510 IF CP>LEN(VR$(VAR)) THEN RETURN
1520 VR$(VAR)=SEG$(VR$(VAR),1,CP-1)&SEG$(
VR$(VAR),CP+1,LEN(VR$(VAR))-CP) ::
GOTO 1570
1530 IF CP=LEN(VR$(VAR)) OR LEN(VR$(VAR))
>83 THEN RETURN
1540 IF CP=1 THEN VR$(VAR)="" & VR$(VAR) ::
VR$(VAR)=SEG$(VR$(VAR),1,84) :: GO
TO 1570
1550 VR$(VAR)=SEG$(VR$(VAR),1,CP-1)&" " &
SEG$(VR$(VAR),CP,LEN(VR$(VAR))-CP+1
) :: GOTO 1570
1560 VR$(VAR)="" :: FOR Z=13 TO 15 :: C
ALL HCHAR(Z,3,32,28) :: NEXT Z :: ER
=1 :: EC=1 :: RETURN
1570 GOSUB 1670 :: RETURN
1580 EC=EC-1 :: CP=CP-1 :: IF EC<1 THEN
EC=28 :: ER=ER-1 :: IF ER<1 THEN ER
=3 :: CP=84
1590 GOTO 1650
1600 EC=EC+1 :: CP=CP+1 :: IF EC>28 THEN
EC=1 :: ER=ER+1 :: IF ER>3 THEN ER
=1 :: CP=1

```

Continued







[illegible]

```

1390 GOSUB 600: CR = CR * .8: S = S * 1.2
1400 : IF (TR D4 = 3) AND D3 = 136
1410 AND TR = 4.1: THEN TR = 24 AND D4 = 21 T
1420 TR = 0: GOTO 1280
1430 F = 0: V = 0: V2 = 0: GOSUB 310: GOTO
1440 IF V < 15 THEN 1490
1450 GOSUB 600: CR = CR * .6: E = E / 2:
1460 IF (TR D4 = 3) AND D3 = 136
1470 AND TR = 4.1: THEN TR = 24 AND D4 = 21 T
1480 TR = 0: F = 0: V = 0: V2 = 0: GOSUB 31
1490 : GOTO 510
1500 IF V < 25 THEN 1510
1510 GOSUB 1260: GOSUB 2410: CR = CR * .4
1520 : GOTO 1280
1530 GOSUB 1260: GOSUB 2400: CR = CR * .1
1540 : GOTO 1280
1550 HGR: HOME: FOR A = 0 TO 19: VTAB
1560 A + 1: HTAB 1: CALL PRNT: PRINT TR$
1570 (OF 0) - 1, A): CALL REST: NEXT : RE
1580 TURN
1590 GOSUB 600: GOSUB 660: D = 0: RETURN
1600 REM LEVEL #1 TERRAIN
1610 DATA "LL=LLLL=====NN=====
1620 DATA "LLLLLLLLLLLL=====NNNNNN=====
1630 DATA "LLJJLLLLLLLL=====NNNNNN=====
1640 DATA "LLLLLLLLLLLLLLLL=====NNOOONN=====
1650 DATA "LLLLLLLLLLLLLLLL=====NNNNNN=====
1660 DATA "LLLLLLLLLLLL=====N=NNNNNN=====
1670 DATA "LLLLLLLLLLLL=====NN=NNN=====
1680 DATA "LLLLLLLLLL=====N=====
1690 DATA "LL=====NNN=====L=====
1700 DATA "LL=====NNNNNNNNNN=====L=====
1710 DATA "NNNN=NN=N=NNNNNN=====L=====
1720 DATA "NNNNNN=====LLJLL=====
1730 DATA "NONNNNN=NN=====LLLLLLLL=====
1740 DATA "OOOONNNNNNNNN=====LLLLLLLL=====
1750 REM LEVEL #2 TERRAIN
1760 DATA "LLLLLLLLLLLLLLLLLLLL=====
1770 DATA "LLLLLLLLLL=LLLLLL=====NN=====
1780 DATA "LLLLLLLL=LLLLLL=====NNN=====
1790 DATA "LLJJL=====LLL=====NNN=====
1800 DATA "LL=LL=====LL=====NNNN=NN=====
1810 DATA "LL=LL=NN=L=====NNNONNNNN=====
1820 DATA "=====N=LL=====NNOOONN=====
1830 DATA "=====LLL=====NNNNNONNN=====
1840 DATA "=====NN=LL=NNNNNNNNNNNN=====
1850 DATA "N=NNNN=====NNNNNNNNNN=N=====
1860 DATA "ONNNNOON=====NNNNNONN=====LLL=====
1870 DATA "ON=NNOOONN=NOOONNN=LL=LLL=====
1880 DATA "N=====NONNN=NOONNN=====LLLL=====
1890 DATA "=====NNNN=OONNN=====LLLL=====
1900 DATA "=====NNN=ONNNNN=====L=L=====
1910 DATA "=====NNNN=ONN=====LLLLLLLL=====
1920 DATA "=====NNNNOOON=====LLL=LLLL=====
1930 DATA "N=====NNNNN=L=LLJJL=====
1940 DATA "ON=====LLL=====

```

**Continued**



```

1950 DATA " ; ON=====LLLL=====
1960 REM " LEVEL #3 TERRAIN
1970 DATA " ==LLL=NNNNNNNNNNNNNNNNNN==LLL=
1980 DATA " =LLL=NNONOOOOOONNNNNN=LLL=
1990 DATA " =LLLL=NOOOO;;; ; ONNNOON=LL=
2000 DATA " =LLJLL=NONOOOOOONNNNOOON=LL=
2010 DATA " LLLLLL=NNNNNNNNNNNNNNNNNN=L=
2020 DATA " LLLLLLL=N=====NONN=LL=NN
2030 DATA " =LLLLLL=NN=NNON=LL=NNO
2040 DATA " ==L=L=NNN=N=NNONN=L=NOO
2050 DATA " N=====NNO=N=N=NONN=L=NO;
2060 DATA " ONN=====NOO=NN=OONNN=NNO
2070 DATA " ; ON=NN=NOOO=OO= ; ; OONN=NN
2080 DATA " ; O=NNNNNNONNNNNNNN ; ; OONN=N=N
2090 DATA " ; ; ; =NNNNOOOOOOOO=N=L=
2100 DATA " ; ; ; N=N=OOOOOOOOOOOON=L=LL=
2110 DATA " ; ; ; ON=NOOOO ; ; ; OOO ; OOO=L=L=N=LL=
2120 DATA " NNN=NNOOO ; ; ; O ; ; ; ON=L=N=LLL
2130 DATA " NNN=NNNNOO ; ; ; OOO ; ON=L=N=LLLL
2140 DATA " NNN=N=NNNNOOOOOOONN=LLLLLLJLL
2150 DATA " NN=NNONNN=NOOOOOOON=LL=LLLLL
2160 DATA " ONNN=====NNNNNNNNN=LL=N=L=
2170 REM LEVEL #4 TERRAIN
2180 DATA " NNNNNNNNNNNNNNNNNNNNNNOOOOOO ; O
2190 DATA " O=====NN=NNNNNNNNN=NNOOOO ; ; ;
2200 DATA " O=LLL=NNN=NNNNNNNN=NNNNOO ; ; ; ;
2210 DATA " N=LJL=NNN=NNNNNNN=NNNNNO ; ; ; ;
2220 DATA " N=LLL=====NNOOOOO ; ; ; ;
2230 DATA " N=====NN=NNNNNNNNNOO ; ; ; ; O
2240 DATA " NOONNNNOONNNNNNNNNNOO ; ; ; ; OOO
2250 DATA " OOOOOOOOONNOONNNNNNNNOO ; ; ; ; O
2260 DATA " OO ; ; ; OONNNNOONNNNNNNNO ; ; ; ; ;
2270 DATA " O ; ; ; ; ONOOOOONNNNNNNNO ; ; ; ; ; O
2280 DATA " ; ; ; OOOONO ; ; ; ; OOOOOO ; ; ; ; ; O ;
2290 DATA " ; ; ; ONNNNOO ; ; ; O ; ; ; ; ; ; ; ; ; O ;
2300 DATA " OOOONOOO ; ; ; ; O ; ; ; ; ; ; ; ; ; O ; O
2310 DATA " OONNO ; ; ; ; ; ; ; ; ; ; ; ; ; OOOOOO
2320 DATA " OONO ; ; ; ; ; ; ; ; ; ; ; ; ; OOOONOOONN
2330 DATA " OONNN ; O ; ; O ; ; ; ; ; ; ; ; ; OONNNO=
2340 DATA " OOOONOO ; O ; ; ; ; ; ; ; ; ; NNN=N=LLL=
2350 DATA " OOOO ; ; ; ; O ; ; ; ; ; ; ; ; ; N ; =LJL=
2360 DATA " OOO ; ; ; ; ; O ; ; ; ; ; ; ; ; ; N=LLL=
2370 DATA " O ; ; ; ; ; ; O ; ; ; ; ; ; ; ; ; ONNNNO
2380 REM MESSAGES
2390 CALL SOUND: CALL SOUND: VTAB 23: HT
AB 35: INPUT "=>" ; AS: HOME: RETUR
NN
2400 HOME: VTAB 22: PRINT "YOUR SHIP HAS
S CRASHED. YOUR CREW DIED": PRINT
IN THE EXPLOSION." : POKE 8,253:
GOTO 2390
2410 HOME: VTAB 22: PRINT "YOU CAME IN
TOO FAST. YOU AND YOUR": PRINT
CREW ARE STRANDED HERE." : POKE 8,2
55: GOTO 2390
2420 HOME: VTAB 22: PRINT "A ROUGH LAND
ING WAS LOST BECAUSE OF A LEAK." : POKE
8,127: GOTO 2390
2430 HOME: VTAB 22: PRINT "NOT A BAD
LANDING. HOWEVER THE": PRINT "LAND
ING GEAR IS DAMAGED." : POKE 8,249:
GOTO 2390
2440 HOME: VTAB 22: PRINT "A PERFECT
LANDING. ALL SYSTEMS": PRINT "ARE
GO, FOR LIFT-OFF." : POKE 8,3: GOTO
2390

```

[illegible]



```

100 REM *****
110 REM ***** EVACU-POD *****
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 5.2.1 *****
190 REM ***** C-64 BASIC *****
200 REM ***** DISPLAY TITLE SCREEN *****
210 POKE 53280,12:POKE 53281,12:POKE 53
220 TS=EVACU-POD
230 PRINT "SHIFT CLR CTRL BLK 10CRSRD
OWN"
240 PRINT "12CRSRDOWN PRESS SPAC
RETURN SPACE TO SPACE CONTINUE CT
RL BLK X=14:Y=10
250 K=PEEK(197):GOSUB1860:S=ABS(S-1):PO
KE 199,S
260 FOR I=1 TO 50:NEXT:PRINTTS:IF K<>1
THEN 250
270 REM INITIALIZE THE PROGRAM
280 PRINT "SHIFT CLR 5CRSRDOWN"
LEASE WAIT WHILE I PREPARE":PRINT "P
CRSRDOWN YOUR TS
290 CS=55296:SP=53248:SD=54272:BL$=""
:DIM MS$(8,2)
300 FOR I=832 TO 1023:READ A:POKE I,A:N
EXT
310 FOR I=704 TO 767:READ A:POKE I,A:N
EXT
320 FOR I=SD TO SD+24:READ A:POKE I,A:N
EXT:FOR I=0 TO 2:POKE 2040+I,13+I:N
EXT
330 FOR I=1 TO 8:FOR J=0 TO 2:READ MS$(
I,J):NEXT:NEXT
340 POKE SP+39,0:POKE SP+40,0:POKE SP+4
1,2
350 T=0:TR=T:TT=T:TR=T:V1=T:V2=T:B=T:H1
=T:H=T:F=T:D1=T:D2=T:V=T
360 D3=48:DX=D3:D4=74:DY=D4:A3=28:A4=19
5:CR=1:S=5000
370 POKE SP,D3:POKE SP+1,D4:POKE SP+2,A
3:POKE SP+3,A4:POKE SP+4,A3:POKE SP
+5,A4
380 POKE SP+16,6
390 REM GET GAME OPTIONS
400 POKE 198,0
410 IF FG=0 THEN Z1=PEEK(63):Z2=PEEK(64
):Z3=PEEK(65):Z4=PEEK(66):FG=1:GOTO
430
420 POKE 63,Z1:POKE 64,Z2:POKE 65,Z3:PO
KE 66,Z4
430 FOR I=0 TO 2:A=4:READ AS:PRINT "SH
IFT CLR AS:5CRSRDOWN
440 FOR J=1 TO 4:READ AS:PRINT "AS:
PRINT:NEXT
450 X=0:Y=23:GOSUB1860:PRINT"SELECT ONE
":O=0
460 GOSUB1880:IF (VAL(K$)<1 OR VAL(K$)>
A) AND (ASC(K$)<>13 OR O=0) THEN 460
470 IF ASC(K$)<>13 THEN O=VAL(K$):PRINT
K$:SHIFT CRSRLEFT::GOTO460
480 IF O<>4 OR I<>1 OR A=9 THEN 510
490 A=9:Y=16:GOSUB1860:PRINT"ENTER GRAV
ITY
500 PRINT"MOON=2, MARS=4, VENUS=6":PRIN
T"GRAVITY(1 TO 9)":GOTO460
510 IF I=1 THEN G=0:2:EF=(5-O)/1.5:IF E
F<0 THEN EF=.5
520 IF I=1 THEN E=G*12500:TH=((S+E)*G)
/EF)/20
530 O(I)=O:NEXT
540 ON O(0) GOSUB2010,2240,2470,2700
550 GOSUB670:GOSUB800:POKE SP+21,3
560 REM MAIN CONTROL LOOP
570 GOSUB930:IF F>1 THEN POKE SD+11,129
:GOTO590
580 POKE SD+11,0
590 GOSUB700:IF TR=0 AND H>0 THEN TR=1
600 IF (TR=1 OR TR=3) AND H1=0 THEN 1570
610 IF TR=2 AND H>0 THEN TR=3:CR=CR*2
620 IF TR=4 AND H1>0 THEN TR=TT:CR=CR*1
.1
630 IF D=1 AND H1<200 THEN D=0:GOSUB800
:GOTO650
640 IF D=0 AND H1>199 THEN D=1:GOSUB750
650 T=T+1:IF T=1500 THEN MS=8:GOSUB1810
:GOTO1480
660 GOTO560
670 REM DISPLAY THE PANEL
680 GOSUB1930:PRINT:PRINT"CTRL BLK BAS
E SURF"
690 PRINT"H.VEL FUEL":PRINT"V
.VEL POWER":RETURN
700 REM DISPLAY CONTROL PANEL VALUES
710 X=4:Y=22:A=9:N=H:GOSUB1880:X=20:N=H
1:GOSUB1880
720 X=6:Y=23:A=7:N=SQR(D1*2+D2*2)*.62.5:
GOSUB1880:X=22:IF E<0 THEN E=0
730 N=INT(E):N$=STR$(N):L=LEN(N$):GOSUB
1960
740 X=5:Y=24:A=8:N=V:GOSUB1880:X=21:N=F
/1000:GOSUB1880:RETURN
750 REM DISPLAY RADAR ABOVE 200 METERS
760 X=27:Y=0:GOSUB1860:FOR I=0 TO 21:PR
INT "CRSRDOWN 3SHIFT CRSRLEFT"
::NEXT

```

```

770 X=29:Y=1:GOSUB1860
780 PRINT"CTRL RVSOFF"CRSRDOWN SHIF
T CRSRLEFTCRSRDOWN SHIFT CRSRLE
FTCRSRDOWN SHIFT CRSRLEFTCRSR
RDOWN SHIFT CRSRLEFTCRSRDOWN
CTRL BLK CRSRDOWN SHIFT CRSRLEFT
CRSRDOWN SHIFT CRSRLEFTCRSRD
OWN SHIFT CRSRLEFTCRSRDOWN SHI
FT CRSRLEFTCTRL BLK CRSRDOWN S
HIFT CRSRLEFTCRSRDOWN SHIFT CRS
RLEFTCRSRDOWN SHIFT CRSRLEFTCR
SRDOWN SHIFT CRSRLEFTCTRL RED
CRSRDOWN SHIFT CRSRLEFTCRSRDO
WN SHIFT CRSRLEFTCRSRDOWN SHIF
T CRSRLEFTCRSRDOWN SHIFT CRSRL
EFTCTRL YEL CRSRDOWN SHIFT CRSR
RDOWN SHIFT CRSRLEFTCRSRDOWN
CRSRRIGHT":
790 PRINT"CTRL GRN":RETURN
800 REM DISPLAY RADAR BELOW 200 METERS
810 X=27:Y=0:GOSUB1860:FOR I=0 TO 21:PR
INT "CRSRDOWN 3SHIFT CRSRLEFT"
::NEXT
820 POKE SD+4,33:FOR I=1 TO 50:NEXT:POK
E SD+4,0
830 X=26:Y=0:GOSUB1860
840 FOR I=200 TO 0 STEP -20
850 PRINTI 2CRSRDOWN LEFTS("6SHIFT C
RSRLEFT",LEN(STR$(I))+1)::NEXT
860 X=30:Y=0:GOSUB1860:PRINT"CMDR A C
RSRDOWN SHIFT CRSRLEFT"
870 FORI=1 TO 20:PRINT"CMDR Q CRSRDOW
N SHIFT CRSRLEFT":NEXT:PRINT"SH
IFT CRSRUP CMDR Z"
880 A=0:IF H<6201 THEN A=6
890 IF H<4201 THEN A=2
900 IF H<2201 THEN A=7
910 IF H<201 THEN A=5
920 POKE 646,A:X=32:Y=21:GOSUB1860:PRIN
T"CTRL RVSON"::RETURN
930 REM GET INPUT FROM KEYBOARD
940 K=PEEK(197)
950 IF K=14 THEN D2=D2-.25:E=E-10+G:GOT
O1080
960 IF K=23 THEN D2=D2+.25:E=E-10+G:GOT
O1080
970 IF K=13 THEN D1=D1-.25:E=E-10+G:GOT
O1080
980 IF K=18 THEN D1=D1+.25:E=E-10+G:GOT
O1080
990 IF K=22 THEN F=((S+E)*(G+.5))/EF:GO
TO1080
1000 IF K=56 THEN FOR I=1 TO 5:NEXT:POKE
198,0:WAIT 198,1:GOTO930
1010 IF K=30 THEN F=F+TH:GOTO1080
1020 IF K=33 THEN F=F+TH*5:GOTO1080
1030 IF K=38 THEN F=F+TH*10:GOTO1080
1040 IF K=34 THEN F=F+TH:GOTO1080
1050 IF K=37 THEN F=F+TH*5:GOTO1080
1060 IF K=42 THEN F=F+TH*10:GOTO1080
1070 GOTO1090
1080 POKE SD+4,33
1090 REM CALCULATE VEL, ALT, AND FUEL
1100 VO=128-(TH*400-F)/(100*G):IFVO>128
THEN VO=128
IF VO<0 THEN VO=0
1110 POKE SD+22,VO
1120 IF E<0 THEN E=0:F=0
1130 IF F<0 THEN F=0
1140 IF F=0 THEN POKE SP+21,3:POKE SP+11
,0:GOTO1170
1160 POKE SP+21,7:POKE SP+11,65
1170 V2=(F*EF)/(S+E)-G:V=V+V2:IF V<0 AND
H1<0 THEN V=0:V2=0
1180 B=(V1+(V2/2)):V1=V:H=H+B:E=E-INT(F/
3000)
1190 REM CHECK FOR A COLLISION
1200 POKE SD+4,0:IF H<0 THEN H=0:H1=0:G
OTO1280
1210 X1=INT((D3-21)/8+.5):Y1=INT((D4-47)
/8+.5):CC=PEEK(CS+X1+Y1*40)AND15
1220 IF CC=7 THEN H1=H-2000:GOTO1270
1230 IF CC=2 THEN H1=H-4000:GOTO1270
1240 IF CC=6 THEN H1=H-6000:GOTO1270
1250 IF CC=0 THEN H1=H-8000:GOTO1270
1260 H1=H
1270 IF H1<0 THEN H1=0
1280 IF (D1<>0 OR D2<>0) AND H1=0 THEN MS=1:GO
SUB1420:GOSUB1810:CR=CR*.1:GOTO1480
1290 IF H>9935 THEN 1330
1300 A4=266-(H/(500/8)):IF D=0 AND H1<20
0 THEN A4=195-H1*.8
IF A4<50 THEN A4=50
1310 POKE SP+3,A4:POKE SP+5,A4
1320 DX=DX+D1:IF DX<24 OR DX>216 THEN DX
=DX-D1:E=E*.95:GOTO1350
1340 D3=INT(DX)
1350 DY=DX+D2:IF DY<50 OR DY>210 THEN DY
=DY-D2:E=E*.95:GOTO1370
1360 D4=INT(DY)
1370 POKE SP,D3:POKE SP+1,D4:RETURN
1380 REM LANDING NOT ON A BASE
1390 TT=TR:TR=4
1400 IF V>14 AND V<.01 THEN MS=7:GOSUB18
10:F=0:V=0:V1=0:GOTO550
1410 MS=1:GOSUB1420:GOSUB1810:CR=.2*TR*
R:GOTO1480

```

Continued



```

1420 REM SHOW CRASH
1430 POKE SD+11,0:POKE SD+18,129:POKE SP
+3,195:POKE SP+5,195
1440 FOR I=1 TO 128:POKE SP+39,I: NEXT:PO
RE 2040,11:POKE SD+18,128
1450 FOR I=1 TO 1000: NEXT
1460 POKE SD+18,128:POKE SP+21,PEEK(SP+2
1) AND 254:POKE 2040,13:POKE SD+18,
0
1470 RETURN
1480 REM PLAY AGAIN?
1490 POKE SP+21,0:GOSUB 1930
1500 IF TR=4 THEN TR=TT
1510 IF TR=1 THEN T=1000
1520 PRINT "CTRL BLK YOUR SCORE IS "INT(
(2000-2*T)+E/G+(500)*CR)*TR
1530 PRINT "WANT TO PLAY AGAIN? (Y/N)";
1540 K=PEEK(197):IF K<>25 AND K<>39 THEN
1540
1550 IF K=39 THEN POKE 198,0:PRINT "SHIF
T CLR":END
1560 GOTO 350
1570 REM PERFECT LANDING
1580 IF (D3<>48 OR D4<>74) AND (D3<>192
OR D4<>186) THEN GOTO 1380
1590 IF V>0 OR V<-4 THEN 1630
1600 IF D3=48 AND D4=74 AND TR=3 THEN MS=6:G
OSUB 1810:CR=CR+.15:TR=4.1:GOTO 1480
1610 IF D3=48 AND D4=74 AND TR=1 THEN MS=5:G
OSUB 1810:F=0:V=0:V2=0:TR=0:GOTO 550
1620 MS=5:GOSUB 1810:F=0:V=0:V2=0:TR=2:CR
=CR+.12:GOTO 550
1630 REM NOT BAD LANDING
1640 IF V<-10 THEN GOTO 1700
1650 MS=4:GOSUB 1810:CR=CR+.8:S=S+.12:GOS
UB 670
1660 IF (TR=1 OR TR=3) AND D3=192 AND D4
=186 THEN TR=2:GOTO 1690
1670 IF TR=3 AND D3=48 AND D4=74 THEN TR
=4.1:GOTO 1480
1680 TR=0
1690 F=0:V=0:V2=0:GOTO 550
1700 REM ROUGH LANDING
1710 IF V<-15 THEN GOTO 1790
1720 MS=3:GOSUB 1810:CR=CR+.6:E=E/2:GOSUB
670
1730 IF (TR=1 OR TR=3) AND D3=192 AND D4
=186 THEN TR=2:GOTO 1690
1740 IF TR=3 AND D3=48 AND D4=74 THEN TR
=4.1:GOTO 1480
1750 TR=0:F=0:V=0:V2=0:GOTO 550
1760 REM TOO FAST LANDING
1770 IF V<-25 THEN GOTO 1790
1780 MS=2:GOSUB 1420:GOSUB 1810:CR=CR+.4:G
OTO 1480
1790 REM YOU BLEW THE LANDING
1800 MS=1:GOSUB 1420:GOSUB 1810:CR=CR+.1:G
OTO 1480
1810 REM PRINT MESSAGE
1820 GOSUB 1930:PRINT "CTRL BLK";:FOR I=
0 TO 2:PRINT MS$(MS,I):NEXT
1830 PRINT "PRESS RETURN";
1840 K=PEEK(197):IF K<>1 THEN 1840
1850 RETURN
1860 REM PLACE CURSOR AT X,Y
1870 POKE 781,Y:POKE 782,X:POKE 783,0:SY
S 65520:PRINT
1880 REM PRINT FORMATTED NUMBERS
1890 N=INT(N*100):N$=STR$(N):L=LEN(N$):N
$=LEFT$(N$,L-2)+"."+RIGHT$(N$,2):L=
L+1
1900 IF N=0 THEN N$=" 0.00":L=5
1910 IF L>A THEN N$="."TILT":L=7
1920 GOSUB 1860:PRINT "CTRL BLK"LEFT$(BL
$,A-L)N$:RETURN
1930 REM ERASE THE PANEL
1940 POKE SD+11,0
1950 X=0:FOR I=21 TO 24:Y=I:GOSUB 1860
1960 PRINT
1970 NEXT:Y=21:GOSUB 1860:RETURN
1980 REM INPUT ONE CHARACTER
1990 POKE 204,0:POKE 207,0:GET K$:IF K$=
" THEN 1990
2000 POKE 204,1:POKE 207,1:RETURN
2010 REM DRAW LEVEL ONE TERRAIN
2020 PRINT "SHIFT CLR CTRL RVSON CTRL
GRN CTRL YEL CTRL RED CTRL GRN CTRL
RL YEL "
2030 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL RED CTRL GRN
TRL YEL "
2040 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL RED CTRL GRN
CTRL YEL "
2050 PRINT "CTRL RVSON CTRL GRN
L WHT SHIFT W CTRL GRN CTRL
L YEL "
2060 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL RED CTRL RED CTRL
L BLU CTRL YEL CTRL YEL "
2070 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL RED CTRL GRN
CTRL YEL "

```

```

2080 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL RED CTRL YEL
L CTRL RED CTRL YEL CTRL YEL
2090 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL RED CTRL YEL
EL CTRL RED CTRL YEL CTRL YEL
2100 PRINT "CTRL RVSON CTRL YEL CTRL
GRN CTRL YEL CTRL YEL CTRL RE
D CTRL YEL CTRL YEL "
2110 PRINT "CTRL RVSON CTRL YEL CTRL
GRN CTRL YEL CTRL YEL CTRL
RED CTRL YEL CTRL GRN CTRL
L YEL "
2120 PRINT "CTRL RVSON CTRL YEL CTRL
L GRN CTRL YEL CTRL RED CTRL
CTRL YEL CTRL RED CTRL YEL
L CTRL GRN CTRL YEL CTRL YEL
2130 PRINT "CTRL RVSON CTRL YEL CTRL
CTRL RED CTRL YEL CTRL RED
CTRL YEL CTRL YEL CTRL GRN CT
RL YEL "
2140 PRINT "CTRL RVSON CTRL YEL
CTRL RED CTRL YEL CTRL YEL
TRL GRN "
2150 PRINT "CTRL RVSON CTRL YEL
CTRL RED CTRL BLU CTRL RED
CTRL YEL CTRL GRN CTRL RED
2160 PRINT "CTRL RVSON CTRL YEL
CTRL RED CTRL YEL CTRL YEL
TRL GRN "
2170 PRINT "CTRL RVSON CTRL YEL CTRL
RED CTRL YEL CTRL RED CTRL
TRL YEL CTRL RED CTRL YEL
CTRL GRN "
2180 PRINT "CTRL RVSON CTRL RED CTRL
RL YEL CTRL RED CTRL YEL CT
RL RED CTRL YEL CTRL RED CTR
L YEL CTRL GRN "
2190 PRINT "CTRL RVSON CTRL RED
CTRL YEL CTRL GRN
CTRL WHT SHIFT W CTRL GRN
2200 PRINT "CTRL RVSON CTRL RED CTRL
BLU CTRL RED CTRL YEL CTRL
RL RED CTRL YEL CTRL GRN
2210 PRINT "CTRL RVSON CTRL BLU
RL RED CTRL YEL CTRL
GRN "
2220 PRINT "CTRL RVSON CTRL BLU CTRL
BLK CTRL BLU CTRL RED CTRL
CTRL YEL CTRL GRN
TRL BLK "
2230 RETURN
2240 REM DRAW LEVEL TWO TERRAIN
2250 PRINT "SHIFT CLR CTRL RVSON CTRL
GRN "
2260 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL GRN CTRL YEL
L CTRL RED CTRL YEL CTRL YEL
2270 PRINT "CTRL RVSON CTRL GRN
CTRL YEL CTRL GRN CTRL YEL
L CTRL RED CTRL YEL CTRL YEL
2280 PRINT "CTRL RVSON CTRL GRN
L WHT SHIFT W CTRL GRN CTRL YEL
CTRL GRN CTRL YEL
CTRL RED CTRL YEL
2290 PRINT "CTRL RVSON CTRL GRN
TRL YEL CTRL GRN CTRL YEL
RED CTRL RED CTRL YEL CTRL
2300 PRINT "CTRL RVSON CTRL YEL CTRL
GRN CTRL YEL CTRL RED CTRL
L YEL CTRL GRN CTRL YEL C
TRL RED CTRL BLU CTRL RED
2310 PRINT "CTRL RVSON CTRL YEL
CTRL RED CTRL YEL CTRL GRN
CTRL YEL CTRL RED CTRL BLU
CTRL RED CTRL YEL CTRL BLU
2320 PRINT "CTRL RVSON CTRL GR
N BLU CTRL RED CTRL YEL CTR
L BLU CTRL RED CTRL YEL
2330 PRINT "CTRL RVSON CTRL RED
CTRL YEL CTRL GRN CTRL YEL
CTRL RED CTRL YEL
2340 PRINT "CTRL RVSON CTRL RED
YEL CTRL RED CTRL YEL CTRL
CTRL RED CTRL YEL CTRL RED
CTRL YEL
2350 PRINT "CTRL RVSON CTRL BLU CTRL
RED CTRL BLU CTRL RED CTRL
YEL CTRL RED CTRL YEL
CTRL GRN CTRL YEL
2360 PRINT "CTRL RVSON CTRL BLU
RED CTRL YEL CTRL RED CTRL B
LU CTRL RED CTRL YEL CTRL
RED CTRL YEL CTRL GRN CTRL
YEL CTRL GRN CTRL YEL
2370 PRINT "CTRL RVSON CTRL RED CTRL
YEL CTRL RED CTRL BLU CTRL
RED CTRL YEL CTRL RED CTRL
RL YEL CTRL GRN CTRL YEL
2380 PRINT "CTRL RVSON CTRL RED
CTRL YEL CTRL BLU CTRL RED
EL CTRL YEL CTRL GRN CTRL Y

```

*Continued*



```
2390 PRINT "CTRL RVSON CTRL RED
      CTRL YEL CTRL BLU CTRL RED
      CTRL YEL CTRL GRN CTRL YEL
      CTRL GRN CTRL YEL
2400 PRINT "CTRL RVSON CTRL RED
      CTRL YEL CTRL BLU CTRL RED
      CTRL YEL CTRL GRN CTRL
      YEL
2410 PRINT "CTRL RVSON CTRL RED
      CTRL BLU CTRL RED CTRL YEL
      CTRL GRN CTRL YEL CTRL GR
      N CTRL YEL
2420 PRINT "CTRL RVSON CTRL RED
      CTRL YEL CTRL GRN CTRL
      YEL CTRL GRN CTRL WHIT SHIFT
      W CTRL GRN CTRL YEL
2430 PRINT "CTRL RVSON
      CTRL GRN CTRL YEL
      CTRL GR
2440 PRINT "CTRL RVSON
      CTRL YEL
2450 PRINT "CTRL RVSON
      CTRL GRN
      CTRL YEL
2460 RETURN
2470 REM DRAW LEVEL THREE TERRAIN
2480 PRINT "SHIFT CLR CTRL RVSON CTRL
      YEL CTRL GRN CTRL YEL CTR
      L RED CTRL BLU CTRL RED
      CTRL YEL CTRL GRN CTRL YEL
      CTRL RED
2490 PRINT "CTRL RVSON CTRL YEL CTRL
      GRN CTRL YEL CTRL RED CT
      RL BLU CTRL RED CTRL BLU CT
      RL RED CTRL YEL CTRL GRN
      CTRL YEL CTRL RED
2500 PRINT "CTRL RVSON CTRL YEL CTRL
      GRN CTRL YEL CTRL RED CTR
      L BLU CTRL RED CTRL BLU
      CTRL RED CTRL YEL CTRL GRN
      CTRL YEL CTRL RED
      CTRL GRN
2510 PRINT "CTRL RVSON CTRL YEL CTRL
      GRN CTRL WHIT SHIFT W CTRL GRN
      CTRL YEL CTRL RED CTRL BLU
      CTRL RED CTRL BLU
      RED CTRL YEL CTRL GRN CTRL
      YEL
2520 PRINT "CTRL RVSON CTRL GRN
      CTRL YEL CTRL RED CTRL BLU
      CTRL RED CTRL BLU CTRL RED
      CTRL YEL CTRL GRN CTRL YEL
2530 PRINT "CTRL RVSON CTRL YEL
      CTRL RED CTRL YEL CTRL RED
      CTRL BLU CTRL RED CTRL YEL
      CTRL GRN CTRL YEL CTRL RED
2540 PRINT "CTRL RVSON CTRL YEL CTRL
      GRN CTRL YEL CTRL RED
      CTRL BLU CTRL RED CTRL YEL
      CTRL GRN CTRL YEL CTRL RED
      CTRL BLU
2550 PRINT "CTRL RVSON CTRL YEL CTR
      L GRN CTRL YEL CTRL GRN CTRL
      YEL CTRL RED CTRL YEL CTRL
      RED CTRL BLU CTRL RED CTRL
      YEL CTRL GRN CTRL YEL CTRL
      RED CTRL BLU CTRL BLK
2560 PRINT "CTRL RVSON CTRL RED CTRL
      YEL CTRL RED CTRL BLU
      CTRL YEL CTRL RED CTRL YEL CT
      RL RED CTRL BLU CTRL RED CTR
      L YEL CTRL GRN CTRL YEL CTRL
      RED CTRL BLU CTRL BLK
2570 PRINT "CTRL RVSON CTRL BLU CTRL
      RED CTRL YEL CTRL RED CT
      RL BLU CTRL YEL CTRL BLU
      CTRL RED CTRL YEL CTRL RED
      CTRL BLU CTRL BLK
2580 PRINT "CTRL RVSON CTRL BLU CTRL
      RED CTRL YEL CTRL RED CTRL
      YEL CTRL RED CTRL BLU CTRL
      YEL CTRL BLK CTRL BLU CTRL
      RED CTRL YEL CTRL RED CT
      RL BLU
2590 PRINT "CTRL RVSON CTRL BLK CTRL
      BLU CTRL YEL CTRL RED CT
      RL BLU CTRL RED CTRL BLK CT
      RL BLU CTRL RED CTRL YEL C
      TRL RED CTRL YEL CTRL RED
2600 PRINT "CTRL RVSON CTRL BLK CTR
      L YEL CTRL RED CTRL BLU
      CTRL YEL CTRL RED CTRL YEL
      CTRL GRN CTRL YEL CTRL RE
      D
2610 PRINT "CTRL RVSON CTRL BLK CTR
      L RED CTRL YEL CTRL RED CTRL
      YEL CTRL BLU CTRL RED
      CTRL YEL CTRL GRN CTRL YEL
      CTRL GRN CTRL YEL CTRL RED
2620 PRINT "CTRL RVSON CTRL BLK CTRL
      BLU CTRL RED CTRL YEL CTRL R
      ED CTRL BLU CTRL BLK CTR
      L BLU CTRL YEL CTRL GRN CTR
      L YEL CTRL RED CTRL YEL CTRL
      GRN CTRL YEL
2630 PRINT "CTRL RVSON CTRL RED CTR
      L YEL CTRL RED CTRL BLU C
      TRL BLK CTRL BLU CTRL RED C
      TRL YEL CTRL GRN CTRL YEL C
      TRL RED CTRL YEL CTRL GRN
```

```
2640 PRINT "CTRL RVSON CTRL RED CTR
      L YEL CTRL RED CTRL BLU CTR
      CTRL BLK CTRL RED CTRL RED CT
      RL YEL CTRL GRN CTRL YEL CTR
      L RED CTRL YEL CTRL GRN CTR
      L YEL CTRL RED CTRL YEL CTR
      L RED CTRL BLU CTRL RED C
      TRL YEL CTRL GRN CTRL RED WHT
      SHIFT W CTRL GRN CTRL YEL
2660 PRINT "CTRL RVSON CTRL RED CTRL
      YEL CTRL RED CTRL BLU CTRL
      RED CTRL YEL CTRL RED CTRL
      BLU CTRL RED CTRL YEL CTRL GR
      N CTRL YEL CTRL GRN CTRL
      RL YEL
2670 PRINT "CTRL RVSON CTRL RED CTR
      RL YEL CTRL RED CTRL BLU
      CTRL RED CTRL YEL CTRL GRN
      CTRL YEL CTRL RED CTRL YEL
      CTRL GRN CTRL YEL CTRL RED
2680 PRINT "CTRL RVSON CTRL YEL
      CTRL RED CTRL BLU CTRL RED
      CTRL YEL CTRL BLU CTRL RE
      D CTRL YEL CTRL RED CTRL BL
      U
2690 RETURN
2700 REM DRAW LEVEL FOUR TERRAIN
2710 PRINT "SHIFT CLR CTRL RVSON CTRL
      RED CTRL BLU
      CTRL BLK CTRL BLU
      CTRL YEL CTRL YEL CTRL RED
      CTRL RED CTRL RED CTRL BLU
      CTRL BLK CTRL BLU
2730 PRINT "CTRL RVSON CTRL YEL CTRL
      GRN CTRL YEL CTRL RED CT
      RL YEL CTRL RED CTRL YEL CT
      RL RED CTRL BLU CTRL BLK
      CTRL BLU
2740 PRINT "CTRL RVSON CTRL RED CTRL
      YEL CTRL GRN CTRL WHIT SHIFT W
      CTRL GRN CTRL YEL CTRL RED
      CTRL YEL CTRL RED CTRL YEL
      CTRL RED CTRL BLU CTRL BLK
      CTRL BLU
2750 PRINT "CTRL RVSON CTRL RED CTRL
      YEL CTRL GRN CTRL YEL CTRL
      CTRL RED CTRL BLU CTRL B
      LK CTRL BLU
2760 PRINT "CTRL RVSON CTRL RED CTRL
      YEL CTRL RED CTRL YEL C
      TRL RED CTRL BLU CTRL BLK
      CTRL BLU CTRL BLK CTRL BLU
2770 PRINT "CTRL RVSON CTRL RED CTRL
      BLU CTRL RED CTRL BLU CT
      RL RED CTRL BLU CTRL BLK
      CTRL BLU
2780 PRINT "CTRL RVSON CTRL RE
      D CTRL BLU CTRL RED CTRL BLU
      CTRL BLK CTRL BLU CTR
      L BLK
2790 PRINT "CTRL RVSON CTRL BLU CTRL
      BLK CTRL BLU CTRL RED C
      TRL BLU CTRL RED CTRL BLU CT
      RL BLK
2800 PRINT "CTRL RVSON CTRL BLU CTRL
      BLK CTRL BLU CTRL RED CTRL
      BLU CTRL RED CTRL BLU CT
      RL BLK CTRL BLU CTRL BLK
2810 PRINT "CTRL RVSON CTRL BLU
      CTRL RED CTRL BLU CTRL BLK
      CTRL BLU CTRL BLK CTRL
      BLU CTRL BLK
2820 PRINT "CTRL RVSON CTRL BLU CTR
      L RED CTRL BLU CTRL BLK CTR
      CTRL BLU CTRL BLK CTRL B
      LU CTRL BLK CTRL BLU CTRL
      BLU
2830 PRINT "CTRL RVSON CTRL RED CT
      RL BLU CTRL BLK CTRL BLU
      CTRL BLK CTRL BLU CTRL
      BLK CTRL BLU
2840 PRINT "CTRL RVSON CTRL RED CT
      RL BLU CTRL BLK CTRL
      BLU
2850 PRINT "CTRL RVSON CTRL RED CTR
      L BLU CTRL BLK CTRL BLU CTR
      TRL BLK CTRL BLU CTRL RED
      CTRL BLU CTRL RED CTRL
      RED CTRL BLK CTRL BLU CTRL
      BLK CTRL BLU CTRL BLU C
      TRL YEL CTRL RED CTRL BLU
      CTRL BLU
2870 PRINT "CTRL RVSON CTRL BLU CT
      RL RED CTRL BLU CTRL BLK CTR
      L BLU CTRL BLK CTRL RED
      CTRL YEL CTRL RED CTRL YEL C
      TRL GRN CTRL YEL CTRL RED
2880 PRINT "CTRL RVSON CTRL BLU CT
      RL BLK CTRL BLU CTRL BLK
      CTRL RED CTRL BLK CTRL YEL
      CTRL GRN CTRL WHIT SHIFT W C
      TRL GRN CTRL YEL CTRL RED
```

*Continued*



EVACU-POD *Continued*

COMMODORE 64

```

2890 PRINT "CTRL RVSON CTRL RED CTRL
    L BLK BLU CTRL BLK CTRL BLK CTRL RED
    L CTRL BLU CTRL BLK CTRL CTRL RED
    CTRL CTRL YEL CTRL CTRL GRN CTRL CTRL YEL
2900 PRINT "CTRL RVSON CTRL BLU CTRL
    BLK BLU CTRL BLK CTRL BLU
    CTRL CTRL BLU CTRL BLK CTRL BLU
2910 PRINT "CTRL RVSON CTRL BLK CTRL
    CTRL BLU CTRL BLK CTRL
    L BLU
2920 RETURN
2930 REM SPRITE DATA FOR SHIPS
2940 DATA 129,000,000,066,000,060,00
    DATA 000,126,000,000,126,000,06
    DATA 000,000,066,000,000,129,000,00
    DATA 000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,00
    DATA 000,024,000,000,024,000,06
    DATA 000,000,060,000,060,000,00
    DATA 126,000,000,126,000,000,126,00
    DATA 000,126,000,000,255,000,000,25
    DATA 000,000,255,000,000,255,000,00
    DATA 255,000,001,231,128,001,195,12
    DATA 003,129,192,003,000,192,006,00
    DATA 096,006,000,096,004,000,032,00
    DATA 000,000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,000,00
    DATA 000,000,000,000,000,000,000,00

```

```

3140 DATA 000,000,000,000,000,000,00
3150 DATA 000,000,000,000,000,000,024,00
3160 DATA 000,060,000,000,060,000,000,02
3170 DATA 000,000,024,000,000,024,000,00
3180 DATA 066,000,000,008,128,000,080,00
3190 DATA 000,005,000,000,144,000,000,00
3200 DATA 128,000,064,000,000,008,128,00
3210 DATA 000,000,000,000,000,000,000,00
3220 DATA 000,000,000,000,000,000,000,00
3230 DATA 000,000,000,000,000,000,000,00
3240 DATA 000,000,000,000,000,000,000,00
3250 DATA 000,000,000,000,000,000,000,00
3260 REM SOUND DATA
3270 DATA 048,011,032,088,064,000
3280 DATA 240,024,002,128,008,128
3290 DATA 000,249,143,010,000,064
3300 DATA 128,000,249,000,034,002,031
3310 REM LANDING AND END OF GAME DATA
3320 DATA "YOUR SHIP HAS CRASHED.", "YOUR
    CREW DIED IN THE EXPLOSION.", "YOUR
    DATA "YOU CAME IN TOO FAST.", "YOU AN
    D YOUR CREW, "ARE STRANDED HERE."
3340 DATA "A ROUGH LANDING.", "HALF OF YOU
    R FUEL WAS LOST, "BECAUSE OF A LEAK."
3350 DATA "NOT A BAD LANDING. HOWEVER, "
    "THE LANDING GEAR IS DAMAGED.", "
    DATA "A PERFECT LANDING. ALL", "SYST
    EMS ARE GO FOR LIFT-OFF.", "
3370 DATA "A PERFECT LANDING.", "THE", "INJU
    RED MINER WILL SURVIVE."
3380 DATA "YOU HAVE MADE A LUCKY LANDING
    .", "YOU CAN REPAIR YOUR SHIP."
3390 DATA "YOUR MISSION HAS TAKEN TOO, "LO
    NG. THE INJURED MINER HAS", "DIED
    DATA SKILL LEVEL, 1] BEGINNER, 2] ADV
    ANCED, 3] EXPERT, 4] PROFESSIONAL
    DATA LOCATION, 1] EARTH'S MOON, 2] MA
    RS, 3] VENUS, 4] OTHER
3420 DATA YEAR OF MISSION, 1] 2485, 2] 224
    B, 3] 2157, 4] 1995

```

HCM

## EVACU-POD

IBM PC &amp; IBM PCjr

```

100 ** EVACU-POD **
110 ** **
120 ** **
130 COPYRIGHT 1985
140 EMERALD VALLEY PUBLISHING CO.
150 BY WILLIAM K. BALTHROP
160 AND THE HCM STAFF
170 HOME COMPUTER MAGAZINE
180 VERSION 5.2.1
190 IBM PCjr W/CARTRIDGE BASIC
    FROM DOS 2.1 or
    IBM PC W/BASICA and
    COLOR/GRAPHICS ADAPTER and
    COLOR MONITOR
240 SCREEN 0,0,0:KEY OFF:FALSE=0:TRUE=-
    1:CR=1:C=3:FOR I=1 TO 10:KEY I,"Z":
    NEXT I
250 DATA &H55,&H77,&HFF,&HBB,&HAA:FOR I
    =0 TO 4:READ T:T$(I)=CHR$(T):NEXT
    I
260 O2=INT(10*RND(1)+1)
270 DEF FN MIN(ARG,LIM)=LIM*-(ARG>LIM)+
    ARG*-(ARG<=LIM)
280 DEF FN MAX(ARG,LIM)=LIM*-(ARG<LIM)+
    ARG*-(ARG>=LIM)
290 RANDOMIZE TIMER
300 DIM CRASH%(20):CRASH%(0)=24:CRASH%(
    1)=16
310 FOR I=2 TO 20:J=-INT(RND(1)*2):CR
    ASH%(I)=INT(32767*RND(1))*(J-(J=0))
    :NEXT I
320 GOSUB 1700:SHIP.WEIGHT=5000:ENGINE.
    EFF=(5-ENGINE)/1.5:THRUST.ADJ=((SH
    IP.WEIGHT+FUEL.WEIGHT)*GRAVITY)/ENG
    INE.EFF)/20:FUEL.WEIGHT=GRAVITY*125
    00
330 DIM SHP%(25),TEST%(2),ALT1%(22),ALT
    2%(22),LV(15),P%(30),Z%(19),ZL%(19)
    ,TS%(25):CLS:DINIT=1:ALTSW=1
340 S:SCREEN 1:DRAW "bm100,100+SHIPS
    GET (97,97)-(103,103),SHP%:PUT (97,
    97),SHP%
360 T(0)=374:T(1)=315:T(2)=255:T(3)=480
    :T(4)=102:RESTORE 1510
370 FOR I=0 TO 15:READ LV(I):NEXT I

```

```

380 KEY OFF:SCREEN 1:COLOR 1,0:ALTIMETE
    R$="fd612u6e1d3fd2d3u3b14d3":ALTIMET
    ER2$="fd612u6e1d3fd2d3u3b14d3ubr2bd2
    d2"
390 DRAW "bm100,100"+ALTIMETER$:GET (95
    ,100)-(105,112),ALT1%:PUT (95,100),
    ALT1%
400 DRAW "bm100,100"+ALTIMETER2$:GET (9
    5,100)-(105,112),ALT2%:PUT (95,100)
    ,ALT2%
410 POKE 1052,PEEK(1050):LINE (0,0)-(29
    9,170),3,B:ON SKILL GOSUB 440,450,4
    20,430:GOTO 460
420 RESTORE 1310:RETURN
430 RESTORE 1210:RETURN
440 RESTORE 1580:RETURN
450 RESTORE 1640:RETURN
460 READ XS,Y:IF LEFT$(XS,1)="*" THEN P
    X=VAL(MID$(XS,2)):PY=Y:GOTO 460
470 X=VAL(X$):IF X$<>"@" THEN LINE (PX,
    PY)-(X,Y),C:PX=X:PY=Y:GOTO 460
480 IF C=0 THEN 560
490 READ N:FOR I=1 TO N:READ P%(I):NE
    XT I
500 FOR I=1 TO N:READ X,Y:PAINT (X,Y)
    ,TS(P%(I)),3:NEXT I
510 READ BX1,BY1,BX2,BY2:SX=BX1:SY=BY1
520 LINE (BX1-5,BY1-5)-(BX1+5,BY1+5),0,
    BF:LINE (BX1-5,BY1-5)-(BX1+5,BY1+5)
    ,0,B:LINE (BX2-5,BY2-5)-(BX2+5
    ,BY2+5),0,BF:LINE (BX2-5,BY2-5)-(BX
    2+5,BY2+5),0,B
530 IF C=3 THEN C=0:GOTO 410
540
550 GOSUB 1530
560 K$=INKEY$:IF K$<>" " THEN DEF SEG=0:
    POKE 1050,PEEK(1052):ON INSTR("jkl
    ioesdxt1JKLUIOESDXT1",K$) GOSUB 760
    780,790,800,810,820,830,840,850,86
    0,870,880,760,780,790,800,810,820,8
    30,840,850,860,870,880
570 IF FUEL.WEIGHT<=0 THEN FUEL.WEIGHT
    =0:POWER=0
580 IF POWER<0 THEN POWER=0
590 IF POWER=0 THEN ALTSW=1 ELSE ALTSW=
    2

```

Continued



```

600 ACCEL=(POWER*ENGINE.EFF)/(SHIP.WEIGHT+
    FUEL.WEIGHT)-GRAVITY:IF V.VELOCIT
    Y<0 AND REL.ALT<=0 THEN V.
    VELOCITY=0:ACCEL=0
610 DELTA.ALT=(PREV.VELOC+ACCEL/2):PREV
    .VEL=V.VELOCITY:ABS.ALT=ABS.ALT+
    DELTA.ALT
620 REL.ALT=ABS.ALT-(LEVEL*2000):FUEL.W
    EIGHT=FUEL.WEIGHT-INT(POWER/300
    0):IF ABS.ALT<0 THEN ABS.ALT=0:
    REL.ALT=0
630 IF REL.ALT<25 THEN GOSUB 1810:TX=7
    :GOSUB 1820:GOTO 2010
640 SX=SX+DX:SY=SY+DY:SY=FNMAX(SX,3):SX
    =FNMIN(SX,287):SY=FNMAX(SY,3):SY=FN
    MIN(SY,167):GOSUB 1530:IF ABS.ALT>0
    AND TR=0 THEN TR=1
650 TI=TI+1:IF TI=1000+250*O2 THEN
    2020
660 IF REL.ALT>0 THEN GOSUB 900:GOSUB
    1100:GOTO 560
670 IF DY+DX<0 THEN GOSUB 1810:TX=7:GO
    SUB 1820:GOTO 2010
680 DOCKED=FALSE:IF SX=BX2 AND SY=BY2 T
    HEN RESCUED=TRUE:DOCKED=TRUE:BASE=2
690 IF SX=BX1 AND SY=BY1 THEN DOCKED=TR
    UE:BASE=1
700 IF V.VELOCITY<-25 OR (V.VELOCITY
    <-4 AND NOT DOCKED) THEN CR=.2*TR*
    CR:GOSUB 1810:TX=7:GOSUB 1820:GOTO
    2010
710 IF V.VELOCITY<-15 THEN CR=CR*.4:T
    X=6:GOSUB 1820:GOTO 770:GOTO 1990
720 IF V.VELOCITY<-10 THEN CR=CR*.6:G
    OSUB 770:FUEL.WEIGHT=FUEL.WEIGHT/2:
    IF DOCKED AND BASE=2 THEN TR=2:TX=5
    :GOSUB 1820:GOTO 750 ELSE IF DOCKED
    AND BASE=1 THEN TR=4.1:TX=5:GOSUB
    1820:IF RESCUED THEN 1980 ELSE 750
730 IF V.VELOCITY<-4 THEN GOSUB 770:C
    R=CR*.8:SY=SY+1.2:IF DOCKED AND BASE=
    2 THEN TR=2:TX=4:GOSUB 1820:GOTO 75
    0 ELSE IF DOCKED AND BASE=1 THEN TX
    =4:GOSUB 1820:IF TR=3 THEN 1980 EL
    SE 750
740 IF V.VELOCITY<0 AND V.VELOCITY>4
    :.99 THEN GOSUB 770:CR=CR*.15:TR=4
    :.1:IF NOT DOCKED THEN TX=3:GOSUB 18
    20:CR=CR*.1:GOTO 750 ELSE IF BASE=
    2 THEN TR=2:TX=2:GOSUB 1820:CR=CR*.2
    :GOTO 750 ELSE IF RESCUED THEN TX=1
    :GOSUB 1820:GOTO 1980 ELSE 750
750 GOSUB 900:GOSUB 1100:GOTO 560
760 POWER=POWER+THRUST.ADJ:GOTO 880
770 V.VELOCITY=0:REL.ALT=0:ABS.ALT=(LEV
    EL*2000):POWER=0:DX=0:RETURN
780 POWER=POWER+THRUST.ADJ*.5:GOTO 880
790 POWER=POWER+THRUST.ADJ*.10:GOTO 880
800 POWER=POWER+THRUST.ADJ:GOTO 880
810 POWER=POWER+THRUST.ADJ*.5:GOTO 880
820 POWER=POWER+THRUST.ADJ*.10:GOTO 880
830 DY=DY-1:GOTO 880:
840 DX=DX-1:GOTO 880:
850 DX=DX+1:GOTO 880:
860 DY=DY+1:GOTO 880:
870 POWER=(SHIP.WEIGHT+FUEL.WEIGHT)*(GR
    AVITY+.5)/ENGINE.EFF:GOTO 880:
880 SOUND 2000,1:IF INSTR(K$, "ulot")
    )<>0 AND TR=0 THEN TR=1:RETURN ELSE
    RETURN
890 DEF SEG=0:POKE 1050,PEEK(1052):WHIL
    E INKEY$="" :GOTO 880:
900 IF REL.ALT<=200 THEN ELEVATION=REL.
    ALT ELSE ELEVATION=ABS.ALT
910 LINE (292,ELEV)-(302,ELEV+12),0,BF
920 IF ELEVATION>200 THEN 1000
930 IF PA<>1 THEN LINE (302,0)-(319,199
    ),0,BF:FOR I=0 TO 199 STEP 5:LINE
    (306,I)-(308,I),1:NEXT I:FOR I=0 T
    O 199 STEP 10:LINE (305,I)-(308,I):
    NEXT I:FOR I=0 TO 199 STEP 50:LI
    NE (302,I)-(308,I):NEXT I:LINE (302,
    199)-(308,199):LINE (309,0)-(309,19
    9),1
940 IF PA<>1 THEN LOCATE 1,40:PRINT "2"
    :LOCATE 13,40:PRINT "1":LOCATE 2,
    40:PRINT "0":LOCATE 3,40:PRINT "0"
    :LOCATE 14,40:PRINT "0":LOCATE 15
    ,40:PRINT "0"
950 ELEV=201-ELEVATION:IF ELEV>185 T
    HEN ELT=185
960 ON ELT SW GOTO 970,980
970 PUT (292,ELEV),ALT1%:GOTO 990
980 PUT (292,ELEV),ALT2%
990 PA=1:RETURN
1000 V=INT(200-(ELEVATION/60)):IF ELE
    V>185 THEN ELEV=185
1010 IF ELEV<0 THEN ELEV=0
1020 IF PA=2 THEN 1060

```

```

1030 LINE TO (302,0)-(319,199),0,BF:FOR I=
    0 TO 199 STEP 1:NEXT I:LINE (302,I)-
    (308,I),1:NEXT I:LINE (309,0)-(309,
    199),1:NEXT I:LINE (309,0)-(309,19
    9),0
1040 FOR I=0 TO 199 STEP 199/48:LINE (
    306,I)-(308,I),1:NEXT I
1050 ON ALTSW GOTO 1070,1080
1060 PUT (292,ELEV),ALT1%:GOTO 1090
1070 PUT (292,ELEV),ALT2%
1080 PA=2:RETURN
1090 LOCATE 23,1:PRINT USING "&####.
    #":BASE 24,1:PRINT USING "&####.
    #":H.VEL 24,1:PRINT USING "&####.
    #":LOCATE 1,1
1100 LOCATE 25,1:PRINT USING "&####.
    #":V.VELOC 25,1:PRINT USING "&####.
    #":SURF 23,20:PRINT USING "&####.
    #":LOCATE 24,20:PRINT USING "&####.
    #":FUEL 24,20:PRINT USING "&####.
    #":LOCATE 25,20:PRINT USING "&####.
    #":POWER 25,20:PRINT USING "&####.
    #":Y1(0)=166:Y2(0)=199:Y1(1)=133:Y2(1)
    =166:Y1(2)=100:Y2(2)=133:Y1(3)=66:Y
    2(3)=100:Y1(4)=33:Y2(4)=66
1110 FOR I=0 TO 4:LINE (311,Y1(I))-
    (319,Y2(I)),1,B:PAINT (318,Y1(I)+3),T8
    (I),1:LINE (311,Y1(I))-(319,Y2(I)),
    0,B:NEXT I
1120 RESTORE:RETURN
1130 REM TERRAIN
1140 DATA *229,88,243,94,261,117,268,121
    ,267,133,259,150,241,103,234,121,225,
    137,237,123,114,229,88,238,103,254,121,
    239,106,167,52,147,260,135,254,194,109,
    186,37,248,147,260,135,254,194,109,
    186,37,181,154,44,12,58,17,60,26,50,36
    ,37,55,24,58,16,52,22,38,35,21,44,1
    220 DATA *1,17,9,42,16,42,12,1,1,71,16
    ,73,19,81,35,67,51,70,48,53,80,21,7
    ,1,1,102,1,105,20,110,37,110,60,96
    ,1,1,99,82,1,12,93,130,90,127,77,117,
    6,115,34,122,14,141,13,146,21,143,
    1,125,49,117,41,151,1,154,2
    1230 DATA 161,1,139,36,160,26,*174,30,1
    ,60,44,144,48,135,48,127,59,144,85,1
    ,35,112,110,116,91,97,96,77,*75,28,9
    ,6,48,100,59,77,74,69,106,77,138,100
    ,161,160,144,177,107,170,80,148,58
    ,166,51,176,30,152,85,154,77,161,80
    ,164,93,158,116,128,137,123,132,128,12
    ,141,134,152,104,152,85,*105,122,10
    ,138,118,152,149,101,151,88,*136,82,1
    ,12,135,108,149,101,151,88,*136,82,1
    ,4,85,101,97,113,106,56,83,40,73,42,6
    ,103,67,82,72,70,86,56,83,40,73,42,6
    ,8,51,60,59,62,69,88,83,42,102,56,12
    ,2,*29,90,42,86,38,98,45,114,45,135
    1250 DATA 27,153,24,142,28,134,37,127,28
    ,115,31,100,29,91,*1,86,15,93,24,169
    ,21,114,11,124,1,160,13,169,*38,24,169
    ,52,155,49,145,62,132,68,136,72,145
    ,68,161,60,169,*135,169,145,159,181
    ,155,199,169,*181,155,228,118,213,8
    ,205,61,187
    1260 DATA 43,191,25,191,15,177,1,217,1
    ,230,18,237,37,233,65,265,97,272,134
    ,289,151,*289,95,276,78,263,87,263,
    ,86,251,75,248,61,252,46,264,32,282,
    ,30,289,37,*289,50,275,44,268,48,260
    ,58,264,72,278,68,289,71,*289,1,279
    ,9,256,14,238,11,234,1,204,11
    1270 DATA 199,33,205,14,216,64,230,74,22
    ,7,59,223,32,205,14,*1,1,1
    1280 DATA 27,2,1,0,3,2,3,3,3,3,4,1,2,4
    ,4,3,2,3,0,1,3,4,3,4,2,2,3
    1290 DATA 6,4,17,4,43,20,90,34,90,50,35
    ,99,18,100,2,165,55,158,55,91,117,83
    ,124,42,124,53,146,119,98,124,200,1
    ,24,223,98,223,63,239,118,231,128,26
    ,6,93,266,81,284,63,276,6,160,8,200,
    ,160,155,165,45,24,246,130
    1300 REM TERRAIN
    1310 DATA *1,62,36,84,40,93,49,91,52,61
    ,72,49,98,23,119,27,121,49,110,76,11
    ,9,119,116,127,94,146,55,117,24,105,
    ,9,86,1,92,*1,99,9,108,10,116,9,121
    1320 DATA 5,126,*1,126,1,127,19,123,23,1
    ,15,27,119,31,131,48,152,68,163,75,1
    ,49,83,145,90,156,99,162,140,142,146
    ,120,149,97,166,88,168,72,156,54,17
    ,7,41,212,63
    1330 DATA 222,65,236,62,245,60,265,75,28
    ,0,70,289,72,*94,169,90,165,86,158,8
    ,3,155,78,153,76,159,75,165,75,169,*
    ,68,139,51,124,38,122,52,142,60,143,
    ,68,139,*131,97
    1340 DATA 126,88,129,72,128,44,149,1,*19
    ,7,1,163,27,153,42,142,52,140,64,136
    ,77,132,97,*125,109,131,106,139,97,
    ,140,90,147,66,153,61,158,66,161,73,
    ,159,81,150,89

```

Continued



**Continued**

IBM PC &amp; IBM PCjr

# PROGRAM LISTING

1350	DATA	117	77	128	100	64	102	65	81	118	124	106	76	113	123	70	123	101	61	102
	193	94	69	45	82	70	102	65	81	118	124	106	76	113	123	70	123	101	61	102
1360	DATA	96	86	70	51	22	63	52	60	117	15	27	10	8	55	11	16	28	72	13
	85	45	82	70	51	22	63	52	60	117	15	27	10	8	55	11	16	28	72	13
1370	DATA	38	14	56	39	38	13	13	11	17	15	27	10	8	55	11	16	28	72	13
	1	25	1	45	38	13	13	11	17	15	27	10	8	55	11	16	28	72	13	13
1380	DATA	18	127	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
	84	16	19	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
1390	DATA	53	71	64	15	58	13	13	13	13	13	13	13	13	13	13	13	13	13	13
	74	21	64	15	58	13	13	13	13	13	13	13	13	13	13	13	13	13	13	13
1400	DATA	24	143	159	9	141	1	1	1	1	1	1	1	1	1	1	1	1	1	1
	7	177	89	185	74	118	195	24	143	159	9	141	1	1	1	1	1	1	1	1
1410	DATA	105	253	163	122	208	117	175	281	145	154	223	156	241	165	220	162	249	167	187
	40	170	142	153	174	206	125	180	275	131	169	223	156	241	165	220	162	249	167	187
1420	DATA	112	117	180	275	131	169	223	156	241	165	220	162	249	167	187	184	168	184	168
	128	269	81	275	131	169	223	156	241	165	220	162	249	167	187	184	168	184	168	184
1430	DATA	24	169	225	269	81	275	131	169	223	156	241	165	220	162	249	167	187	184	168
	45	215	169	225	269	81	275	131	169	223	156	241	165	220	162	249	167	187	184	168
1440	DATA	251	97	260	135	125	180	275	131	169	223	156	241	165	220	162	249	167	187	184
	4	258	121	194	161	236	166	17	197	179	12	12	12	12	12	12	12	12	12	12
1450	DATA	222	209	157	230	166	17	197	179	12	12	12	12	12	12	12	12	12	12	12
	6	226	122	194	161	236	166	17	197	179	12	12	12	12	12	12	12	12	12	12
1460	DATA	238	206	156	230	166	17	197	179	12	12	12	12	12	12	12	12	12	12	12
	3	238	206	156	230	166	17	197	179	12	12	12	12	12	12	12	12	12	12	12
1470	DATA	141	27	3	4	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
	3	141	27	3	4	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
1480	DATA	112	175																	

[illegible]

## HCM



```

100 | *****  

110 | ** EVACU-POD **  

120 | *****  

130 | COPYRIGHT 1985  

140 | EMERALD VALLEY PUBLISHING CO.  

150 | BY WILLIAM K. BALTHROP  

160 | HOME COMPUTER MAGAZINE  

170 | VERSION 5.2.1  

180 | TI EXTENDED BASIC  

190 |  

200 | CALL CLEAR :: DIM O(2) :: DISPLAY AT  

    | (12,9)::"EVACU-POD" :: DISPLAY AT(23  

    | )::PRESS ENTER TO CONTINUE"  

210 | CALL KEY(O,K,S) :: IF S=0 THEN 210  

220 | CALL CLEAR :: FOR X=1 TO 9 :: CALL  

    | COLOR(X,2,1) :: NEXT X :: CALL SCREE  

    | N(16)  

230 | CALL CHAR(120,"81423C3C3C3C4281026C  

    | 9E1EC2442420100CC86C300107C20183C3C7  

    | E7E7E66C3")  

240 | CALL CHAR(33,"FFFFFFFFFFFFFFFFF",97  

    | "00",98,"FFFFFFF",42,"FFFFF  

    | FFFFFFFFFF")  

250 | CALL CHAR(63,"FFFFFFFFFFFFFFFFF",104  

    | "5A5A5A185A",112,"010101030101010F  

    | ",113,"0101010301010107")  

260 | CALL CHAR(62,"FF818199998181FFFFFFF  

    | FFFFFFFFFF") :: CALL COLOR(1,4,1,2,  

    | 5,1)  

270 | TR,TI,D1,D2,V1,V2,F,H,B,H1,V=0 :: S  

    | =5000 :: DX,D3=25 :: DY,D4=41 :: CR  

    | =1  

280 | GOSUB 1270 :: CALL COLOR(1,4,1,2,5,  

    | 1,9,10,11) :: CALL SPRITE(#2,123,2,1  

    | 60,222,#3,32,7,168,222)  

290 | CALL SPRITE(#2,123,2,160,222,#3,32,  

    | 7,168,222)  

300 | GOSUB 510 :: IF F>1 THEN CALL SOUND  

    | (-4250,110,VO,220,VO,110,VO,-5,VO)  

310 | GOSUB 900 :: IF TR=0 AND H>0 THEN T  

    | R=1  

320 | IF (TR=1 OR TR=3) AND H1=0 THEN 1080  

330 | IF TR=2 AND H>0 THEN TR=3 :: CR=CR+  

    | 2  

340 | IF TR=4 AND H1>0 THEN TR=TI :: CR=C  

    | R+1  

350 | IF D=1 AND H1<200 THEN D=0 :: GOSUB  

    | 420 :: CALL SOUND(-4250,110,VO,220  

    | ,VO,110,VO,-5,VO) :: GOTO 370  

360 | IF D=0 AND H1>199 THEN D=1 :: GOSUB  

    | 390  

370 | TI=TI+1 :: IF TI=1000+250*O(0) THEN  

    | RESTORE 2400 :: GOSUB 2310 :: GOSUB  

    | 2320 :: GOTO 1020 ELSE 300  

380 | DISPLAY AT(22,1)::"BASE":TAB(14)::"SU  

    | RF":"H.VEL":TAB(14)::"FUEL":"V.VEL":  

    | TAB(14)::"POWER" :: CALL LOCATE(#1,D  

    | 3,D4) :: GOTO 460  

390 | CALL VCHAR(1,30,32,20) :: CALL VCHAR  

    | (6,30,63,4) :: CALL VCHAR(10,30,42,4  

    | ) :: CALL VCHAR(14,30,98,4)  

400 | CALL VCHAR(18,30,97,4) :: CALL HCHAR  

    | (22,28,33,2) :: IF H<9935 THEN CALL  

    | LOCATE(#2,160-H/(500/8),222,#3,168-  

    | H/(500/8),222)  

410 | RETURN  

420 | CALL SOUND(-4250,110,VO,220,VO,110,  

    | VO,-5,VO)  

430 | CALL VCHAR(6,30,32,16) :: RESTORE 24  

    | 10 :: FOR OH=1 TO 42 STEP 2 :: READ  

    | B,C :: CALL HCHAR(OH/2,29,B)  

440 | CALL HCHAR(OH/2,30,C) :: NEXT OH ::  

    | IF H1>200 THEN A=200 ELSE A=H1  

450 | CALL LOCATE(#2,161-A*.8,222,#3,169-  

    | A*.8,222)  

460 | IF H<201 THEN CALL HCHAR(22,28,33,2  

    | ) :: RETURN  

470 | IF H<2201 THEN CALL HCHAR(22,28,97,  

    | 2) :: RETURN  

480 | IF H<4201 THEN CALL HCHAR(22,28,98,  

    | 2) :: RETURN  

490 | IF H<6201 THEN CALL HCHAR(22,28,42,  

    | 2) :: RETURN  

500 | CALL HCHAR(22,28,63,2) :: RETURN  

510 | CALL KEY(1,K1,S1) :: CALL KEY(2,K2,S  

    | 2) :: IF (S1=0 AND S2=0) THEN 680  

520 | IF S1=0 THEN 600  

530 | CALL SOUND(-400,110,VO,220,VO,440,0  

    | -5,VO) :: CALL SOUND(-4250,110,VO,2  

    | 20,VO,110,VO,-5,VO)  

540 | IF K1=5 THEN D1=D1-.25 :: E=E-10*G  

    | :: GOTO 600  

550 | IF K1=0 THEN D1=D1+.25 :: E=E-10*G  

    | :: GOTO 600  

560 | IF K1=2 THEN D2=D2-.25 :: E=E-10*G  

    | :: GOTO 600  

570 | IF K1=3 THEN D2=D2+.25 :: E=E-10*G  

    | :: GOTO 600  

580 | IF K1=11 THEN F=((S+E)*(G+.5))/EF  

590 | IF K1=19 AND S1=1 THEN GOSUB 2320 :  

    | GOTO 510  

600 | IF S2=0 THEN 680  

610 | CALL SOUND(-400,110,VO,220,VO,660,0  

    | -5,VO) :: CALL SOUND(-4250,110,VO,2  

    | 20,VO,110,VO,-5,VO)  

620 | IF K2=12 THEN F=F-THA*10 :: GOTO 68  

    | 0  

630 | IF K2=3 THEN F=F-THA*5 :: GOTO 680

```

```

6400 IF K2=2 THEN F=F+THA :: GOTO 680
6500 IF K2=4 THEN F=F+THA :: GOTO 680
6600 IF K2=5 THEN F=F+THA*5 :: GOTO 680
6700 IF K2=6 THEN F=F+THA*10 :: GOTO 680
6800 VO=MAX((THA*200-F)/840,0) :: VO=MIN(
VO,30)
6900 IF E<=0 THEN E=0 :: F=0
7000 IF F<0 THEN F=0
7100 IF F=0 THEN CALL PATTERN(#3,32) ELSE
CALL PATTERN(#3,104)
7200 V2=(F*EF)/(S+E)-G :: V=V+V2 :: IF V
<0 AND H1<=0 THEN V,V2=0
7300 B=(V1+(V2/2)) :: V1=V :: H=H+B :: E=
E-INT(F/3000)
7400 IF H<=0 THEN H,H1=0 :: GOTO 820
7500 CALL GCHAR(ABS((D3+4)/8+.5),ABS((D4
+4)/8+.5),CC)
7600 IF CC=97 THEN H1=H-2000 :: GOTO 810
7700 IF CC=98 THEN H1=H-4000 :: GOTO 810
7800 IF CC=42 THEN H1=H-6000 :: GOTO 810
7900 IF CC=63 THEN H1=H-8000 :: GOTO 810
8000 H1=H
8100 IF H1<0 THEN H1=0
8200 IF (D1<>0 OR D2<>0) AND H1=0 THEN RE
STORE 2330 :: GOSUB 1000 :: GOSUB 2
310 :: CR=CR*.1 :: GOSUB 2320 :: GO
TO 1020
8300 IF H>9935 THEN 870
8400 IF D=0 AND H1<200 THEN GOTO 860
8500 CALL LOCATE(#2,160-(H/(500/8)),222,
#3,168-(H/(500/8)),222) :: GOTO 870
8600 CALL LOCATE(#2,161-H*.8,222,#3,169
-H1*.8,222)
8700 DX=DX+D1 :: IF DX<1 OR DX>160 THEN
DX=DX-D1 :: E=E*.95 ELSE D3=INT(DX)
8800 DY=DY+D2 :: IF DY<17 OR DY>208 THEN
DY=DY-D2 :: E=E*.95 ELSE D4=INT(DY
)
8900 CALL LOCATE(#1,D3,D4) :: RETURN
9000 DISPLAY AT(22,5) SIZE(8) : USING "####
#.#.#":H
9100 DISPLAY AT(22,18) SIZE(8) : USING "####
#.#.#":H1
9200 DISPLAY AT(23,7) SIZE(6) : USING "####
#.#.#":SQR(D1^2+D2^2)*62.5
9300 DISPLAY AT(23,20) SIZE(6) : USING "####
#.#.#":E
9400 DISPLAY AT(24,6) SIZE(7) : USING "####
#.#.#":V
9500 DISPLAY AT(24,19) SIZE(7) : USING "####
#.#.#":F/1000 :: RETURN
9600 TT=TR :: TR=4 :: IF V>-4 AND V<.01
THEN RESTORE 2390 :: GOSUB 2310 ::
GOSUB 2320 :: GOSUB 380 :: F,V,V1=0
:: GOTO 300
9700 RESTORE 2330 :: CALL DELSPRITE(#2,#
3) :: GOSUB 1000 :: GOSUB 2310 :: GO
SUB 2320
9800 CR=2*TR*CR
9900 GOTO 1020
10000 FOR C=1 TO 5 :: CALL SOUND(300,110,
0,110,0,110,0,-8,0) :: CALL PATTERN(
#1,121)
10100 CALL SOUND(400,110,0,110,0,220,0,-4
,0) :: CALL PATTERN(#1,122) :: NEXT C
:: RETURN
10200 DISPLAY AT(23,1) : "WANT TO PLAY AGAI
N." Y=N :: CALL DELSPRITE(ALL) ::
IF TR=4 THEN TR=TT
10300 IF TR=1 THEN TI=1000
10400 DISPLAY AT(22,1) : "YOUR SCORE IS:" I
NT(((2000-2*TI)+E/G+(O(0)*500))*CR)
*TR
10500 ACCEPT AT(24,1) VALIDATE("YN") SIZE(1
) BEEP:AS
10600 IF AS="N" THEN STOP
10700 IF AS<>"Y" THEN GOTO 1050 ELSE CALL
CLEAR :: GOTO 270
10800 IF (D3<>25 OR D4<>41) AND (D3<>137 OR
D4<>185) THEN GOTO 960
10900 IF V>0 OR V<-4 THEN 1130
11000 IF D3=25 AND D4=41 AND TR=3 THEN RE
STORE 2380 :: GOSUB 2310 :: GOSUB 2
320 :: CR=CR*1.5 :: TR=4.1 :: GOTO
1020
11100 IF D3=25 AND D4=41 AND TR=1 THEN RE
STORE 2370 :: GOSUB 2310 :: GOSUB 2
320 :: TR,F,V,V2=0 :: GOSUB 380
GOTO 300
11200 RESTORE 2370 :: GOSUB 2310 :: GOSUB
2320 :: GOSUB 380 :: F,V,V2=0 :: T
R=2 :: CR=CR*1.2 :: GOTO 300
11300 IF V<-10 THEN GOTO 1190
11400 RESTORE 2360 :: GOSUB 2310 :: GOSUB
2320 :: CR=CR*.8 :: S=S*1.2 :: GOS
UB 380
11500 IF (TR=1 OR TR=3) AND D3=137 AND D4=
185 THEN TR=2 :: GOTO 1180
11600 IF TR=3 AND D3=25 AND D4=41 THEN TR
=4.1 :: GOTO 1020
11700 TR=0
11800 F,V,V2=0 :: GOTO 300
11900 IF V<-15 THEN GOTO 1240
12000 RESTORE 2350 :: GOSUB 2310 :: GOSUB
2320 :: CR=CR*.6 :: E=E/2 :: GOSUB
380
12100 IF (TR=1 OR TR=3) AND D3=137 AND D4=
185 THEN TR=2 :: GOTO 1180

```

**Continued**







```

420  FL$ = " ": FOR IT = 1 TO 30: FL$ = FL$ + " ": NEXT
430  $FL$ = 0
440  DIM MN$(5)
450  BL$ = CHR$(5)
460  AS = "A": Z$ = "Z"
470  CM$ = "4": B2 = 5
480  B1 = 4
490  SP$ = "1"
500  DIM CZ(15)
510  DIM WX$(6)
520  REM INSERT TO BLANK WORD
530  FOR IT = 1 TO 10: READ X$: IZ$(IT) = CHR$(ASC(X$) - 64): NEXT: IZ$(11) = CHR$(ASC(27))
540  FOR IT = 1 TO 5: READ MN$(IT): NEXT
550  FOR IT = 1 TO 5: READ X$: WX$(IT) = CHR$(ASC(X$) - 64): NEXT: WX$(6) = CHR$(27)
560  RETURN
570  DATA "H", "M", "U", "K", "J", "Q", "Z", "B"
580  DATA "CREATE WORD LIST", "LOAD WORD LIST", "SAVE WORD LIST", "PRINT WORD LIST", "PLAY SWITCH 'N' SPELL"
590  DATA "E", "R", "H", "U", "M"
600  REM MAIN MENU
610  HOME
620  VTAB 1: HTAB 8: PRINT "SWITCH 'N' SPELL"
630  FOR IT = 1 TO MC(MN)
640  VTAB 4 + IT: HTAB 5: PRINT "1. EDIT THE PRESENT WORD LIST"
650  PRINT IT: MN$(IT): NEXT
660  VC = 3: GOSUB 3100
670  VC = 20: HC = 29
680  VTAB 20: HTAB 5: PRINT "ENTER A NUMBER (1 - (MC(MN)))"
690  VTAB 22: HTAB 7: PRINT "PRESS <ESC> TO LEAVE"
700  GOSUB 4600
710  IF (VAL(IN$) < 1 OR VAL(IN$) > MC(MN)) AND IN$ < > IZ$(11) THEN 710
720  CH = VAL(IN$)
730  RETURN
740  REM CREATE WORD LIST
750  HOME
760  VTAB 1: HTAB 10: PRINT "CREATE A WORD LIST"
770  IF FL = 0 THEN 880
780  VC = 3: GOSUB 3100
790  VTAB 7: HTAB 5: PRINT "1. EDIT THE PRESENT WORD LIST"
800  VTAB 10: HTAB 5: PRINT "2. CREATE A NEW WORD LIST"
810  VTAB 13: HTAB 7: PRINT "ENTER A NUMBER (1 - 2)"
820  VC = 13: HC = 31: GOSUB 4600
830  IF IN$ = IZ$(11) THEN RETURN
840  IF IN$ < > 1 AND IN$ < > 2 THEN 850
850  EN PRINT BL$: GOTO 830
860  FOR IT = 3 TO 13: VTAB IT: HTAB 1: CALL 868: NEXT
870  IF IN$ = "1" THEN 920
880  FL$ = " ": GOSUB 2880
890  IF IN$ = IZ$(11) THEN RETURN
900  FL = 1: WLS(FL) = SP$
910  HOME: VTAB 1: HTAB 10: PRINT "CREATE A WORD LIST"
920  VTAB 17: HTAB 1: PRINT "CTRL: "
930  INVERSE: PRINT CHR$(ASC(IZ$(1)) + 64): NORMAL: PRINT "LEFT"
940  HTAB 17: INVERSE: PRINT CHR$(ASC(IZ$(5)) + 64): NORMAL: PRINT "C"
950  VTAB 18: HTAB 8: INVERSE: PRINT "C"
960  HTAB 17: INVERSE: PRINT CHR$(ASC(IZ$(6)) + 64): NORMAL: PRINT "C"
970  VTAB 19: HTAB 8: INVERSE: PRINT "C"
980  HTAB 17: INVERSE: PRINT CHR$(ASC(IZ$(7)) + 64): NORMAL: PRINT "C"
990  VTAB 20: HTAB 8: INVERSE: PRINT "C"
1000  HTAB 17: INVERSE: PRINT CHR$(ASC(IZ$(8)) + 64): NORMAL: PRINT "C"
1010  VTAB 21: HTAB 17: INVERSE: PRINT CHR$(ASC(IZ$(9)) + 64): NORMAL: PRINT "C"
1020  VTAB 23: HTAB 10: PRINT "PRESS <ESC> TO LEAVE"
1030  VC = 3: GOSUB 3100
1040  VK = 1
1050  GOSUB 2600
1060  GOSUB 1700
1070  GOSUB 2690

```

```

1080  RETURN
1090  REM LOAD WORD LIST
1100  VTAB 1: HTAB 10: PRINT "LOAD A WORD LIST"
1110  SV$ = FL$: FL$ = " "
1120  GOSUB 2880
1130  IF IN$ = IZ$(11) THEN FL$ = SV$: RETURN
1140  TURN = "1"
1150  VTAB 14: HTAB 5: PRINT "FROM DRIVE NUMBER: 1"
1160  VC = 14: HC = 24
1170  GOSUB 4600
1180  IF IN$ = IZ$(11) THEN RETURN
1190  IF IN$ = IZ$(10) THEN IN$ = "1": GOTO 1220
1200  TO IF IN$ < > "1" AND IN$ < > "2" THEN 1210
1210  EN "D" + IN$
1220  DR$ = "D" + IN$
1230  HOME
1240  IF PD THEN GOSUB 4960
1250  VTAB 10: HTAB 5: PRINT "LOADING A WORD LIST"
1260  PRINT CHR$(4); "OPEN "FL$: DR$
1270  PRINT CHR$(4); "READ "FL$: DR$
1280  INPUT X$: FL = VAL(X$): IF FL = 0 THEN 1320
1290  IF FL > 50 THEN FL = 50
1300  FOR IT = 1 TO FL
1310  INPUT WLS(IT): NEXT
1320  PRINT CHR$(4); "CLOSE"
1330  RETURN
1340  REM SAVE WORD LIST
1350  IF FL = 0 THEN RETURN
1360  HOME: VTAB 10: HTAB 5: PRINT "SAVING THE PRESENT WORD LIST"
1370  VTAB 12: HTAB 7: DR$ = "1"
1380  PRINT "DRIVE NUMBER: 1"
1390  VC = 12: HC = 22: GOSUB 4600
1400  IF IN$ = IZ$(11) THEN RETURN
1410  IF IN$ = "1" OR IN$ = "2" THEN DR$ = "1"
1420  IF IN$ = IZ$(10) THEN 1440
1430  GOTO 1390
1440  DR$ = "D" + DR$
1450  IF PD THEN GOSUB 4960
1460  PRINT CHR$(4); "OPEN "FL$: DR$
1470  PRINT CHR$(4); "OPEN "FL$: DR$
1480  GOSUB 5030
1490  PRINT CHR$(4); "WRITE "FL$: DR$
1500  PRINT CHR$(4); "WRITE "FL$: DR$
1510  PRINT STR$(FL)
1520  FOR IT = 1 TO FL
1530  PRINT WLS(IT): NEXT
1540  PRINT CHR$(4); "CLOSE"
1550  RETURN
1560  REM PRINT WORD LIST
1570  IF FL = 0 THEN RETURN
1580  HOME: PRINT " "
1590  GOSUB 5060
1600  IF IN$ = IZ$(11) THEN RETURN
1610  IF IN$ = "1" THEN GOSUB 5150: RETURN
1620  PRINT CHR$(4); "PR#1"
1630  PRINT "SWITCH 'N' SPELL WORD LIST:"
1640  PRINT "FL$ = "FL$: PRINT " "
1650  FOR IT = 1 TO FL: NM$ = " "
1660  PRINT NM$: WLS(IT): NEXT
1670  PRINT CHR$(4); "PR#0"
1680  RETURN
1690  REM WORD ENTRY
1700  GOSUB 2530
1710  HX = 1
1720  VB = VK
1730  IF VB > 10 THEN VB = VB - 10: GOTO 1730
1740  HC = HX + B2: VC = VB + B1: GOSUB 5240
1750  IF IN$ = "-" OR (IN$ > "A" AND IN$ < "Z") THEN 1840
1760  X$ = " "
1770  FOR IT = 1 TO 11: IF IN$ = IZ$(IT) THEN X = IT
1780  NEXT
1790  IF X = 11 THEN GOSUB 2570: RETURN
1800  IF X = 0 THEN PRINT BL$: GOTO 1740
1810  ON X GOSUB 1890, 1920, 1950, 2020, 2100, 2410, 2480, 2220, 2310, 2180
1820  IF FL = 0 THEN FL$ = " ": RETURN
1830  GOTO 1720
1840  ET$(HX) = IN$: HX = HX + 1
1850  IF HX > 15 THEN HX = 15
1860  GOTO 1740
1870  RETURN
1880  REM CURSOR LEFT
1890  IF HX > 1 THEN HX = HX - 1
1900  RETURN
1910  REM CURSOR RIGHT
1920  IF HX < 15 THEN HX = HX + 1
1930  RETURN
1940  REM CURSOR UP
1950  GOSUB 2570
1960  IF VK = 1 THEN RETURN

```

*Continued*



```

119970 VK IF VK = VK THEN PG = PG + 1 THEN GOSUB 2600
119980 RETURN
119990 REM
120000 IF VK = VK THEN PG = PG + 1 THEN GOSUB 2600
120010 RETURN
120020 IF VK = VK THEN PG = PG + 1 THEN GOSUB 2600
120030 RETURN
120040 IF VK = VK THEN PG = PG + 1 THEN GOSUB 2600
120050 RETURN
206000 GOSUB 2550
206010 RETURN
206020 REM
206030 IF HX = 15 OR ET$(15) < > " " THEN
206040 FOR IT = 1 TO HX STEP - 1
206050 ET$(IT) = ET$(IT + 1)
206060 VTAB VB + B1: HTAB IT + B2 + 1: PRI
206070 NT ET$(IT + 1): NEXT
206080 ET$(HX) = VTAB VB + B1: HTAB H
206090 X + B2: PRINT ET$(HX);
206100 REM
206110 RETURN
206120 REM
206130 GOSUB 2020
206140 HX = 1
206150 RETURN
206160 REM
206170 INSERT WORD SPACE
206180 IF FL = 50 THEN RETURN
206190 GOSUB 2570
206200 FOR IT = FL TO VK STEP - 1
206210 WLS(IT + 1) = WLS(IT): NEXT
206220 WLS(VK) = SP$: GOSUB 2530
206230 FL = FL + 1
206240 GOSUB 2600
206250 RETURN
206260 REM
206270 DELETE WORD
206280 IF VK < FL THEN 2340
206290 VK = VK - 1: FL = FL - 1: IF FL = 0
206300 THEN RETURN
206310 GOSUB 2530: GOSUB 2600: RETURN
206320 FOR IT = VK TO FL - 1
206330 WLS(IT) = WLS(IT + 1): NEXT
206340 FL = FL - 1
206350 GOSUB 2530
206360 GOSUB 2600
206370 RETURN
206380 REM
206390 DELETE CHARACTER
206400 IF HX = 15 THEN 2450
206410 FOR IT = HX TO 14
206420 ET$(IT) = ET$(IT + 1)
206430 VTAB VB + B1: HTAB IT + B2: PRINT E
206440 TS(IT): NEXT
206450 ET$(15) = VTAB VB + B1: HTAB 1
206460 S + B2: PRINT
206470 RETURN
206480 REM
206490 BLANK FIELD
206500 HX = 1
206510 FOR IT = 1 TO 15: ET$(IT) = " "
206520 VTAB VB + B1: HTAB IT + B2: PRINT "
206530 : NEXT
206540 RETURN
206550 REM
206560 CONVERT STRING TO STRING ARR
206570 AY
206580 FOR IT = 1 TO 15: ET$(IT) = MIDS(W
206590 LS(VK), IT, 1): IF ET$(IT) = " THEN
206600 ET$(IT) = "
206610 NEXT
206620 RETURN
206630 REM
206640 CONVERT STRING ARRAY TO STR
206650 ING
206660 WLS(VK) = "": FOR IT = 1 TO 15: WLS(
206670 VK) = WLS(VK) + ET$(IT): NEXT
206680 RETURN
206690 REM
206700 PRESENT PARTIAL LIST OF WORDS
206710 PG = 41
206720 IF PG > VK THEN PG = PG - 10: GOTO
206730 2610
206740 FOR IT = 0 TO 9
206750 VTAB 5 + IT: HTAB 1: CALL - 868: I
206760 F PG + IT < 10 THEN HTAB 2
206770 PRINT PG + IT:
206780 IF PG + IT < FL THEN HTAB 6: PR
206790 INT WLS(PG + IT);
206800 NEXT
206810 RETURN
206820 REM
206830 LIST CLEANUP
206840 HOME: VTAB 10: HTAB 5
206850 PRINT "LIST CLEANUP"
206860 IF FL = 0 THEN RETURN
206870 FOR IT = 1 TO FL: WLS(IT) = LEFT$(
206880 (WLS(IT) + SP$, 15)
206890 FOR JI = 1 TO 15: P1 = JI: IF MIDS
206900 (WLS(IT), JI, 1) < > " " THEN 2750
206910 NEXT JI
206920 FOR JI = 15 TO 1 STEP - 1: P2 = JI:
206930 IF MIDS(WLS(IT), JI, 1) < > " " O
206940 R P2 < P1 THEN 2770
206950 NEXT JI
206960 IF P2 < P1 THEN WLS(IT) = SP$: GOTO
206970 2790
206980 WLS(IT) = MIDS(WLS(IT), P1, (P2 - P
206990 1 + 1))
207000 IF LEN(WLS(IT)) = 1 THEN WLS(IT)
207010 = WLS(IT) +
207020 NEXT IT

```

**Continued**



```

3600 RETURN
3610 REM BLANK PROMPTS
3620 FOR BX = 18 TO 23: VTAB BX: HTAB 1:
3630 CALL 868: NEXT
3640 RETURN
3650 REM SHOW ORIGINAL WORD
3660 GOSUB 3620
3670 VTAB 19: HTAB 1
3680 PRINT "ORIGINAL WORD: "WLS(KX)
3690 FOR DL = 1 TO 1500: NEXT
3700 GOSUB 3620: GOSUB 3490
3710 RETURN
3720 REM FINISH PROMPT
3730 GOSUB 3620
3740 GOSUB 3410
3750 VTAB 19: HTAB 1
3760 PRINT "DONE! PRESS ANY KEY TO CONT
3770 INUE: "
3780 VC = 19: HC = 36: GOSUB 4600
3790 REM GET REVERSAL PLAY KEYBOARD CH
3800 OICE
3810 HX = 1: NRS(1) = "": NRS(2) = ""
3820 VTAB 18: HTAB 34: PRINT
3830 IF WRS = WLS(KX) THEN RETURN
3840 VC = 18: HC = HX + 33: GOSUB 4600
3850 IF IN$ > "0" AND IN$ < "9" TH
3860 EN 3970
3870 X = 0
3880 FOR IT = 1 TO 6: IF IN$ = WX$(IT) T
3890 HEN X = IT: GOTO 3880
3900 NEXT
3910 PRINT BLS: GOTO 3810
3920 IF X = 6 THEN RETURN
3930 IF X = 5 THEN 3940
3940 ON X GOSUB 3650, 4070, 4010, 4030
3950 IF X > 3 THEN GOTO 3810
3960 IF X = 1 THEN 3790
3970 RETURN
3980 NR = VAL (NRS(1) + NRS(2))
3990 IF NR = 2 AND NR < CZ THEN G
4000 OSUB 4210: GOTO 3790
4010 PRINT BLS: GOTO 3790
4020 NRS(HX) = IN$: IF HX < 2 THEN HX =
4030 2
4040 GOTO 3810
4050 RETURN
4060 REM LEFT ARROW
4070 IF HX = 2 THEN HX = 1: RETURN
4080 RETURN
4090 REM RIGHT ARROW
4100 IF HX = 1 THEN HX = 2: RETURN
4110 RETURN
4120 REM AUTOSOLVING
4130 GOSUB 3620
4140 VTAB 19: HTAB 10
4150 PRINT "AUTOSOLVING "
4160 FOR SX = LEN (WLS(KX)) TO 2 STEP
4170 1
4180 IF ETS(SX) = MIDS (WLS(KX), SX, 1) T
4190 HEN 4170
4200 FOR SY = 1 TO SX - 1
4210 IF ETS(SY) < MIDS (WLS(KX), SX, 1)
4220 IF THEN 4160
4230 IF SY > 1 THEN NR = SY: GOSUB 4210
4240 NR = SX: GOSUB 4210: GOTO 4170
4250 NEXT SY
4260 NEXT SX
4270 GOSUB 3620
4280 RETURN
4290 REM PERFORM REVERSAL
4300 N1 = 1: N2 = NR
4310 IF N1 > N2 THEN GOSUB 4380: GOS
4320 UB 4400: GOTO 4320
4330 GOSUB 4380: GOSUB 4400
4340 WMS(3) = LEFT$(WBS, 2 * N1 + 1) +
4350 MIDS (WMS(3), 2 * (N1 + 1), 2 * (N2
4360 - 1) + 1) + RIGHT$(WMS(3),
4370 LEN (WBS) - 2 * N2)
4380 WMS(4) = LEFT$(LEFT$(WBS, 2 * N
4390 1 - 1) + ETS(N1) + WBS), LEN (WBS))
4400 : WMS(2) = LEFT$(LEFT$(WBS, 2 *
4410 N2 - 1) + ETS(N2) + WBS), LEN (WBS))
4420 GOSUB 4380: GOSUB 4400
4430 WMS(2) = WBS: WMS(4) = WBS: WMS(1) =
4440 LEFT$(LEFT$(WMS(1), 2 * N2 - 1)
4450 + ETS(N2) + WBS), LEN (WBS))
4460 WMS(5) = LEFT$(WMS(5), 2 * N1 - 1)
4470 + ETS(N1) + RIGHT$(WMS(5), LEN (
4480 WBS) - 2 * N1)
4490 GOSUB 4400
4500 FOR HX = 1 TO 2: GOSUB 4380: GOSUB
4510 4400: NEXT HX
4520 XS = ETS(N1): ETS(N1) = ETS(N2): ETS(
4530 N2) = XS: N1 = N1 + 1: N2 = N2 - 1: I
4540 F N1 = < N2 THEN 4220
4550 FOR GX = 1 TO 5: WMS(GX) = WBS: NEXT
4560
4570 WRS = "": WMS(3) = "": FOR GX = 1 T
4580 O CZ: WMS(3) = WMS(3) + ETS(GX) + "
4590 ": WRS = WRS + ETS(GX): NEXT
4600 WMS(3) = LEFT$(LEFT$(WMS(3) + WBS), LE
4610 N (WBS))
4620 GOSUB 4400
4630 RV = RV + 1: GOSUB 3410
4640 RETURN

```

```

4650 WMS(1) = RIGHT$(LEFT$(WMS(1) + " "
4660 EN (WBS)): WMS(5) = LEFT$(LEFT$(W
4670 MS(5)), LEN (WBS)): RETURN
4680 REM PRINT SWITCH 'N' SPELL SETUP
4690 FOR GX = 1 TO 5: VTAB 6 + GX: HTAB
4700 2
4710 PRINT WMS(GX): NEXT
4720 RETURN
4730 REM WORD RANDOMIZER
4740 VTAB 9: HTAB 4: PRINT "RANDOMIZING
4750 WORD
4760 FOR IT = 1 TO 15: CZ(IT) = 0: NEXT
4770 CZ = LEN (WLS(KX))
4780 FOR IT = 1 TO CZ
4790 P1 = 1 + INT ((CZ - IT + .5) * RN
4800 D (1))
4810 P2 = 1: P3 = 0
4820 IF CZ(P2) = 0 THEN P3 = P3 + 1
4830 IF P3 = P1 THEN ETS(IT) = MIDS (WL
4840 S(KX), P2, 1): CZ(P2) = 1: GOTO 4530
4850 P2 = P2 + 1: GOTO 4500
4860 NEXT IT
4870 FOR IT = 1 TO 5: WMS(IT) = WBS: NEXT
4880
4890 WRS = "": WMS(3) = "": FOR IT = 1 T
4900 O CZ: WMS(3) = WMS(3) + ETS(IT) + "
4910 ": WRS = WRS + ETS(IT): NEXT
4920 IF WRS = WLS(KX) THEN 4450
4930 WMS(3) = LEFT$(LEFT$(WMS(3) + WBS), LE
4940 N (WBS))
4950 RETURN
4960 REM GET CHARACTER
4970 VTAB VC: HTAB HC
4980 IN$ =
4990 GET IN$: IF IN$ = " " THEN 4620
5000 VTAB VC: HTAB HC: PRINT IN$:
5010 RETURN
5020 REM ERROR HANDLER
5030 HOME: VTAB 7: HTAB 1
5040 X = PEEK (222)
5050 IF X = 5 THEN 4760
5060 PRINT "ERROR NUMBER "X"
5070 PRINT "AT LINE "PEEK (218) + PEE
5080 K (219) * 256)
5090 VTAB 12: HTAB 1
5100 PRINT "PRESS ANY KEY TO CONTINUE: "
5110 GET IN$: IF IN$ = " " THEN 4730
5120 RUN
5130 RETURN
5140 PRINT " "FL$ "
5150 PRINT " "IS NOT ON DRIVE " RIGHTS
5160 (DR$, 1)
5170 GOSUB 5030: GOTO 4710
5180 REM GET PRODOS FILENAME
5190 LTS = CHR$(8): CRS = CHR$(13)
5200 MX = 15: TS = 12: HT = 4
5210 VTAB VT: HTAB HT: PRINT TS: SPC (MX
5220 - LEN (TS)): HTAB LEN (TS) + HT
5230 IF LEN (TS) = MX THEN HTAB MX
5240 IF TS = 1 THEN GOSUB 4910: GOTO 4
5250 860
5260 GOSUB 4930
5270 IF KS = CRS THEN 4950
5280 IF KS = LTS AND LEN (TS) < 2 THEN
5290 TS = "": GOTO 4820
5300 IF KS = LTS THEN TS = LEFT$(TS, L
5310 EN (TS) - 1): GOTO 4820
5320 IF LEN (TS) < MX THEN TS = TS + KS
5330 : GOTO 4820
5340 TS = LEFT$(TS, MX - 1) + KS: GOTO
5350 4820
5360 GET KS: IF NOT (KS = LTS OR KS = C
5370 RS OR (KS > "A" AND KS < "Z"))
5380 THEN PRINT CHR$(7): GOTO 4910
5390 RETURN
5400 GET KS: IF NOT (KS = LTS OR KS = C
5410 RS OR (KS > "A" AND KS < "Z"))
5420 OR (KS > "0" AND KS < "9") O
5430 R KS = " " THEN PRINT CHR$(7):
5440 GOTO 4930
5450 RETURN
5460 FL$ = TS: RETURN
5470 HTAB 1: VTAB 22: PRINT "PLACE DISK
5480 IN DRIVE "MIDS (DR$, 3, 1): " AND PR
5490 ESS RETURN
5500 GOSUB 5010: IF KB < > (141) THEN 4
5510 970
5520 HTAB 1: VTAB 22: CALL 868
5530 PRINT CHR$(4): "PREFIX": DR$
5540 RETURN
5550 KB = PEEK (-16384): IF KB > 127
5560 THEN POKE -16368, 0: RETURN
5570 GOTO 5010
5580 PRINT CHR$(4): "CLOSE "FL$
5590 PRINT CHR$(4): "DELETE "FL$: DR$: R
5600 ETURN
5610 REM PRINT TO SCREEN OR PRINTER
5620 HOME: VTAB 1: HTAB 10: PRINT "PRIN
5630 T WORD LIST
5640 VC = 3: GOSUB 3100: VTAB 7: HTAB 5:
5650 PRINT "1. PRINT TO SCREEN"
5660 VTAB 10: HTAB 5: PRINT "2. PRINT TO
5670 PRINTER"
5680 VTAB 13: HTAB 7: PRINT "ENTER A NUM
5690 BER (1
5700 2)
5710 VC = 13: HC = 31: GOSUB 4600

```

Continued



# SWITCH 'N' SPELL Continued

APPLE II Family

```

5110 IF IN$ = IZ$(1) THEN RETURN
5120 IF IN$ < IZ$(1) AND IN$ < "2" TH
EN PRINT BLS:: GOTO 5100
5130 RETURN
5140 REM PRINT TO SCREEN
5150 VE = 1
5160 GOSUB 2600
5170 VTAB = 17: HTAB 7: PRINT "PRESS <RETU
RN> TO CONTINUE"
5180 VTAB = 19: HTAB 10: PRINT "PRESS <ESC
> TO LEAVE"
5190 GET K$: IF K$ < IZ$(1) AND K$ <
IZ$(11) THEN 5190

```

```

5200 IF K$ = IZ$(11) THEN RETURN
5210 VK = VK + 10: IF FL > VK THEN 5160
5220 RETURN
5230 REM GET CHARACTER FOR EDITING
5240 VTAB VC: HTAB HC
5250 IN$ = " "
5260 GET IN$: IF IN$ = " " THEN 5260
5270 IF IN$ = "Z" THEN VTAB VC: HTAB HC
: PRINT IN$
5280 RETURN

```

HCM

# SWITCH 'N' SPELL

COMMODORE 64

```

100 REM *****
110 REM * SWITCH 'N' SPELL *
120 REM *****
130 REM COPYRIGHT 1984, 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM BY RANDY THOMPSON
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 5.2.1
180 REM C-64 BASIC
190 REM
200 REM INITIALIZE VARIABLES
210 POKE 53281,12:POKE 646,0:POKE 53272
,21:POKE 657,128:POKE 53284,3
220 POKE 53265,PEEK(53265) OR 64
230 MX=50: DIM WLS(MX+1): TS="SWITCH N SP
ELL"
240 FOR I=0 TO 6
250 READ ST$(I),BC(I),SD(I,1),SD(I,2)
260 NEXT
270 REM MENU
280 SC=0: R=2: PRINT:POKE 646,0:GOSUB 2710
: X=7
290 IF NR>0 THEN R=6
300 FOR I=1 TO R
310 Y=1.3+1:GOSUB 2690
320 PRINT "CHR$(I+48)" "ST$(I)
330 NEXT
340 X=2:Y=23:GOSUB 2690
350 PRINT "SHIFT I SHIFT N SHIFT P
SHIFT U SHIFT T SPACE SHIFT A S
PACE SHIFT N SHIFT U SHIFT M SHI
FT B SHIFT E SHIFT R SPACE M SHI
FT E SHIFT E SHIFT T SHIFT W SHI
FT N SHIFT N SPACE SHIFT O SHI
FT N SHIFT E SPACE SHIFT A SHI
FT N SHIFT D SPACE": IF R=6 THEN
PRINT "SHIFT S SHIFT I SHIFT X
SPACE"
360 IF R=2 THEN PRINT "SHIFT T SHIFT W
SHIFT O SPACE"
370 POKE 198,122:POKE 204,0:GET K$:IF
K$=" " THEN 370
380 IF ASC(K$)=13 AND SC<>0 THEN 520
390 IF ASC(K$)<49 OR ASC(K$)>48+R THEN 3
70
400 POKE 198,0:POKE 646,0:PRINT CHR$(AS
C(K$)+128)CHR$(157):
410 IF SC=0 THEN X=4:Y=24:GOSUB 2690:PRI
NT "SHIFT P SHIFT U SHIFT S SHI
FT H SPACE SHIFT R SHIFT E SHIF
T T SHIFT U SHIFT R SHIFT N SPA
CE SHIFT T SHIFT O"
420 IF SC<>0 THEN L=LEN(ST$(SC)):X=1075
+(SC*120):GOSUB 3120
430 SC=VAL(K$):V=1:H=SD(SC,1):L=SD(SC,2
):GOSUB 2950
440 X=19:Y=24:GOSUB 2690
450 POKE 646,1:PRINT CHR$(18)ST$(SC)CHR
$(146):POKE 646,0
460 FOR I=1 TO 18-LEN(ST$(SC))
470 PRINT CHR$(160);
480 NEXT
490 X=37:Y=23:GOSUB 2690
500 C=BC(SC):L=LEN(ST$(SC)):X=1075+(SC*
120):GOSUB 3060:V=1:GOSUB 3030
510 POKE 198,0:POKE 646,3:POKE 207,0:GO
TO 370
520 POKE 204,1:POKE 207,0:POKE 646,0
530 ON SC GOSUB 550,950,1200,1450,1950,2
450,2600
540 GOTO 270
550 REM CREATE WORD LIST
560 GOSUB 2710:POKE 198,0:IF NR=0 THEN 68
0
570 X=3:Y=4:GOSUB 2690
580 PRINT "DO YOU WANT TO ...":X=4:Y=9:G
OSUB 2690
590 PRINT "[1] ADD TO THE CURRENT WORD
LIST"
600 X=18:Y=11:GOSUB 2690:PRINT "OR":X=4:
Y=13:GOSUB 2690
610 PRINT "[2] CREATE A NEW WORD LIST"
620 X=0:Y=23:GOSUB 2690
630 PRINT "2SPACE SHIFT I SHIFT N SHI
FT P SHIFT U SHIFT T SPACE SHI
FT A SPACE SHIFT N SHIFT U SHIFT
T M SHIFT B SHIFT E SHIFT R SPA
CE SHIFT B SHIFT E SHIFT T SHIF
T W SHIFT E SHIFT N SPACE SHIF
T O SHIFT N SPACE SHIFT A SHIF
T A SHIFT N SHIFT D SPACE SHIF
T N SHIFT W SHIFT O 3SPACE":POKE
1980,122

```

```

640 POKE 2023,96
650 X=37:Y=23:GOSUB 2690
660 GOSUB 3580:IF K$<"1" AND K$<"2" TH
EN 660
670 IF K$="2" THEN FOR I=1 TO NR:WLS(I)
="":NEXT:NR=0:SS=0
680 POKE 198,0:POKE 204,1:POKE 207,0:GO
SUB 2600
690 X=10:Y=24:GOSUB 2690
700 MS="F7-TO RETURN TO MENU":GOSUB 3170
710 IF NR=0 THEN S=1:W=S:GOTO 730
720 W=NR+1:S=ABS(INT((W-1)/10)*10)+1:IF
W>MX THEN W=MX:S=MX-9
730 GOSUB 2600:IF S=10<0 THEN 770
740 X=1:Y=23:GOSUB 2690
750 MS="F1-FOR WORDS"
760 MS=MS+STR$(S-10)+"-MID$(STR$(S-1)
,LEN(STR$(S-1))-1)+"":GOSUB 3170
770 IF S>NR OR S=MX-9 THEN 810
780 X=21:Y=23:GOSUB 2690
790 MS="F3-FOR WORDS"
800 MS=MS+STR$(S+10)+"-MID$(STR$(S+19
),LEN(STR$(S+19))-1)+"":GOSUB 3170
810 GOSUB 3610
820 X=15:Y=(W-S)*2+2:GOSUB 2690
830 POKE 198,0:L=15:B=65:T=90:GOSUB 3240
:IF S<>WLS(W) THEN SS=0
840 WLS(W)=S:IF LEN(SS)=1 THEN WLS(W)=
SS+
850 IF W>NR AND S<>" " THEN NR=W
860 IF S$=" " AND NR=W THEN NR=NR-1
870 IF ASC(K$)=136 THEN RETURN
880 IF ASC(K$)=133 THEN S=S-10:W=S:GOTO
730
890 IF ASC(K$)=134 THEN S=S+10:W=S:GOSU
B 730
900 IF ASC(K$)<>145 THEN 930
910 W=W-1:IF W<S THEN S=S-10:GOTO 730
920 GOTO 820
930 IF W<MX THEN W=W+1:IF W=S+10 THEN
S=S+10:GOTO 730
940 GOTO 820
950 REM LOAD WORD LIST
960 SS=1:WB=WW:WW=0:RB=NR
970 PRINT "SHIFT CLR":POKE 53265,PEEK
(53265) AND 191:POKE 53280,BC(SC):P
OKE 646,BC(SC)
980 POKE 53281,0:PRINT "CTRL RVSON"
LOAD WORD LIST
990 X=0:Y=24:GOSUB 2690
1000 PRINT "CTRL RVSON"
CTRL RVSON
:POKE 2023,160
CTRL RVSONOFF
1010 POKE 56295,BC(SC):POKE 646,7
1020 X=0:Y=2:GOSUB 2690
1030 PRINT "PLEASE ENTER FILE NAME :":L=
14:B=32:T=90:GOSUB 3240:IF S$=" " THE
N 1150
1040 FL$=S$:GOSUB 3730:IF K$="T" THEN 1060
1050 OPEN 1,8,8,"0:"+FL$+".W"+",S,R":GOT
O 1070
1060 OPEN 1,1,0,FL$+".W"
1070 INPUT#1,NR:IF NR<1 OR NR>50 THEN GO
TO 1110
1080 FOR I=1 TO NR
1090 INPUT#1,WLS(I)
1100 NEXT
1110 IF K$="T" THEN 1140
1120 OPEN 15,8,15:INPUT#15,V,S$:CLOSE 15:I
F V=0 THEN 1140
1130 SS=0:NR=RB:WW=WB:PRINT:PRINT "..."
S$
1140 CLOSE 1
1150 X=10:Y=24:GOSUB 2690
1160 POKE 646,BC(SC):PRINT "CTRL RVSON"
F7-TO RETURN TO MENU:CTRL RVSON";
GET K$:IF K$=" " THEN 1170
1170 IF ASC(K$)<>136 THEN 1170
1180 POKE 53265,PEEK(53265) OR 64:POKE 5
3283,12:RETURN
1190 REM SAVE WORD LIST
1200 SS=1
1210 PRINT "SHIFT CLR":POKE 53265,PEEK
(53265) AND 191:POKE 53280,BC(SC):P
OKE 646,BC(SC)
1230 POKE 53281,0:PRINT "CTRL RVSON"
SAVE WORD LIST
1240 X=0:Y=24:GOSUB 2690

```

Continued



```

1250 PRINT "CTRL RVSON"
1260 POKE 56295,BC(SC):POKE 646,7
1270 X=0:Y=2:GOSUB2690
1280 PRINT "PLEASE ENTER FILE NAME";:L=
14:B=32:T=90:GOSUB3240:IFSS="":THEN
1400
1290 FLS=SS:GOSUB3730:IF KS="T" THEN1310
1300 OPEN 1,8,8,"@0:"+FLS+"W"+",S,W":GO
TO1320
1310 OPEN 1,1,1,FLS+"W"
1320 PRINT#1,NR
1330 FOR I=1 TO NR
1340 PRINT#1,WL$(I)
1350 NEXT
1360 IF KS="T" THEN1140
1370 OPEN15,8,15:INPUT#15,V,SS:CLOSE15:1
FV=0 THEN1140
SS=0:PRINT:PRINT "SS"
1380 CLOSE1
1390 X=10:Y=24:GOSUB2690
1400 POKE 646,BC(SC):PRINT "CTRL RVSON";
1410 F7 TO RETURN TO MENU:CTRL RVSON";
1420 GET KS:IF KS="T" THEN1420
1430 IF ASC(KS)<>136 THEN1420
1440 POKE 53265,PEEK(53265) OR 64:POKE 5
3283,12:RETURN
1450 REM PRINT WORD LIST
1460 GOSUB2710:POKE 198,0
1470 X=2:Y=5:GOSUB2690
1480 PRINT "DO YOU WANT TO ...":X=3:Y=10:
GOSUB2690
1490 PRINT "[1] PRINT WORD LIST ON THE
SCREEN":PRINT TAB(18)"OR"
1500 PRINT TAB(3)"[2] PRINT WORD LIST O
N THE PRINTER"
1510 X=0:Y=23:GOSUB2690
1520 PRINT "2SPACE"SHIFT I"SHIFT N"SH
IFT P"SHIFT U"SHIFT T"SPACE"SHI
FT A"SPACE"SHIFT N"SHIFT U"SHIF
T M"SHIFT B"SHIFT E"SHIFT R"SPA
CE"SHIFT B"SHIFT E"SHIFT T"SHIF
T W"2SHIFT E"SHIFT N"SPACE"SHIF
T O"SHIFT N"SHIFT E"SPACE"SHIF
T A"SHIFT N"SHIFT D"SPACE"SHIF
T T"SHIFT W"SHIFT O"3SPACE":POKE
1980,122
1530 POKE 2023,96
1540 X=37:Y=23:GOSUB2690
1550 GOSUB3580:IF KS<>"1" AND KS<>"2" TH
EN1550
1560 POKE 204,1:POKE 207,0:GOSUB2600
1570 IF KS="2" THEN1760
1580 REM PRINT WORD LIST ON THE SCREEN
1590 S=1:X=10:Y=24:GOSUB2690
1600 MS=F7 TO RETURN TO MENU:GOSUB3170
1610 GOSUB2600:IF S-10<0 THEN1650
1620 X=1:Y=23:GOSUB2690
1630 MS=F1 FOR WORDS
1640 MS=MS+STR$(S-10)+" "+MID$(STR$(S-1)
,LEN(STR$(S-1))-1)+"":GOSUB3170
1650 IF S+10>NR OR S=MX-9 THEN1690
1660 X=21:Y=23:GOSUB2690
1670 MS=F3 FOR WORDS
1680 MS=MS+STR$(S+10)+" "+MID$(STR$(S+19)
,LEN(STR$(S+19))-1)+"":GOSUB3170
1690 POKE 198,0:GOSUB3610
1700 GET KS:IF KS="T" THEN1700
1710 IF ASC(KS)<>133 AND ASC(KS)<>134 AN
D ASC(KS)<>136 THEN1700
1720 IF ASC(KS)=136 THEN RETURN
1730 IF ASC(KS)=133 AND PEEK(1945)=70 TH
EN S=S-10:GOTO1610
1740 IF ASC(KS)=134 AND PEEK(1965)=70 TH
EN S=S+10:GOTO1610
1750 GOTO1700
1760 REM PRINT WORD LIST ON THE PRINTER
1770 POKE 53283,PEEK(53281):GOSUB2600:X=
14:Y=12:GOSUB2690
1780 PRINT "CTRL RVSON PRINTING"
1790 OPEN 4,4:CMD4
1800 FOR I=1 TO NR
1810 IF I AND 2 THEN POKE 53283,PEEK(532
81)
1820 PRINT "WORD #";:IF I<10 T
HEN PRINT " ";
1830 PRINT I" - WL$(I)
1840 IF I AND 1 THEN POKE 53283,BC(SC)
1850 PRINT
1860 NEXT
1870 POKE 53283,PEEK(53281)
1880 PRINT#4:CLOSE4
1890 X=14:Y=12:GOSUB2690:PRINT
1900 X=10:Y=24:GOSUB2690
1910 MS=F7 TO RETURN TO MENU:GOSUB3170
1920 GET KS:IF KS="T" THEN1920
1930 IF ASC(KS)<>136 THEN1920
1940 RETURN
1950 REM PLAY SWITCH 'N' SPELL
1960 A=0:GOSUB2710
1970 X=1
1980 FOR Y=1 TO 9
1990 GOSUB2690
2000 PRINT "SPACE"
2010 NEXT

```

```

2020 POKE 646,BC(SC):X=1
2030 FOR Y=10 TO 12
2040 GOSUB2690
2050 PRINT "CTRL RVSON"
CTRL RVSOFF
2060 NEXT:POKE 646,0
2070 X=1
2080 FOR Y=13 TO 21
2090 GOSUB2690
2100 PRINT "SPACE"CTRL RVSON"36SPACE"
CTRL RVSOFF"SPACE"
2110 NEXT:GOSUB2690:PRINT "38SPACE"
2120 X=17:Y=14:GOSUB2690
2130 PRINT "CTRL RVSON"SHIFT P"SHIFT
R"SHIFT E"2SHIFT S"
2140 X=7:Y=16:GOSUB2690
2150 PRINT "CTRL RVSON"SHIFT F"4SPACE
"SHIFT T"SHIFT O"SPACE"SHIFT H"
2SHIFT E"SPACE"SHIFT T"SHIFT O"
SHIFT E"SPACE"SHIFT W"SHIFT O"
SHIFT R"SHIFT D"SPACE"SHIFT A"O"
SHIFT G"SHIFT A"SHIFT I"SHIFT N"
:POKE 1672,241:POKE 55944,0
:POKE 1674,237:POKE 55946,0
2160 X=10:Y=18:GOSUB2690
2170 PRINT "CTRL RVSON"SHIFT F"4SPACE
"SHIFT A"SHIFT U"SHIFT T"SHIFT O
SPACE"SHIFT S"SHIFT O"SHIFT L
SHIFT V"SHIFT E":POKE 1755,243:P
OKE 56027,0
2190 POKE 1757,237:POKE 56029,0
2200 X=9:Y=20:GOSUB2690
2210 PRINT "CTRL RVSON"SHIFT F"4SPACE
"SHIFT T"SHIFT O"SPACE"SHIFT R
SHIFT E"SHIFT T"SHIFT U"SHIFT R
SHIFT N"SPACE"SHIFT T"SHIFT O
SPACE"SHIFT M"SHIFT E"SHIFT N"
SHIFT U":POKE 1834,247:POKE 56106,
0
2220 POKE 1836,237:POKE 56108,0
2230 X=1:Y=23:GOSUB2690
2240 PRINT "SHIFT H"SHIFT O"SHIFT W
SPACE"SHIFT M"SHIFT A"SHIFT N"
SHIFT Y"SPACE"SHIFT L"SHIFT E"2S
HIFT T"SHIFT E"SHIFT R"SHIFT S"
SPACE"SHIFT D"SHIFT O"SPACE"SHI
FT Y"SHIFT O"SHIFT U"SPACE"SHIF
T W"SHIFT A"SHIFT N"SHIFT T"SPA
CE"SHIFT R"SHIFT E"SHIFT V"SHIF
T E"SHIFT R"SHIFT S"SHIFT E"SHI
FT D":POKE 1982,127:POKE 56254,0
2250 X=3:Y=Y+1:GOSUB2690
2260 PRINT "SHIFT I"SHIFT N"SHIFT P
SHIFT U"SHIFT T"SPACE"SHIFT A"
SPACE"SHIFT N"SHIFT U"SHIFT M"SH
IFT B"SHIFT E"SHIFT R"SPACE"SHI
FT B"SHIFT E"SHIFT T"SHIFT W"2S
HIFT N"SHIFT N"3SPACE"SHIFT A"
SHIFT N"SHIFT D":POKE 2018,122:PO
KE 56250,0
2270 POKE 2010,114:POKE 56282,0
2280 WW=WW+1:IF WW>NR THEN WW=1
2290 NT=0
2300 GOSUB3910:GOSUB4500
2310 FOR I=1 TO LEN(WL$(WW))*200:NEXT
2320 Y=11:GOSUB3950:GOSUB4000:GOSUB4070
2330 X=32:GOSUB4130:GOSUB4170
2340 X=35:GOSUB4130:GOSUB2690:L=2:B=48:T
=57:GOSUB3240
2350 IF ASC(KS)<>133 THEN2370
2360 Y=11:GOSUB3910:FOR I=1 TO LEN(WL$(W
W))*300:NEXT:GOSUB3950:GOTO2340
2370 IF ASC(KS)=134 THEN4840
2380 IF ASC(KS)=136 THEN RETURN
2390 IF VAL(SS)>LEN(WL$(WW)) OR VAL(SS)<
2 THEN GOTO2340
2400 R=VAL(SS)
2410 NT=NT+1:GOSUB4210:GOSUB4290
2420 IF SW=WL$(WW) THEN GOSUB4610:GOTO2
280
2430 IF A=1 THEN GOTO4840
2440 GOTO2340
2450 REM EXIT PROGRAM
2460 IF NR=0 OR SS=1 THEN2560
2470 GOSUB2710
2480 X=6:Y=8:GOSUB2690
2490 PRINT "DO YOU WANT TO EXIT WITHOUT"
2500 X=9:Y=10:GOSUB2690
2510 PRINT "SAVING THE WORD LIST?"
2520 X=17:Y=12:GOSUB2690
2530 PRINT "(Y/N)"
2540 GET KS:IF KS="Y" THEN2540
2550 IF KS<>"Y" THEN RETURN
2560 PRINT "SHIFT CLR":POKE 53265,PEEK(
53265) AND 191
POKE 53280,14:POKE 53281,6:L=7:R=L
WL$(WW)="GOODBYE":SW="EYBDOOG":GOS
UB4000:GOSUB4210:GOSUB4290
2590 POKE 657,0:POKE 646,14:END
2600 REM CLEAR BORDERED SCREEN
2610 X=1
2620 FOR I=1 TO 22
2630 Y=1:GOSUB2690
2640 PRINT " "
2650 NEXT

```

Continued



```

2660 Y=I:GOSUB2690
2670 PRINT "38SPACE";
2680 RETURN
2690 REM PLOT CURSOR AT X,Y
2700 POKE 781,Y:POKE 782,X:POKE 783,0:SY
S 65520:RETURN
2710 REM DRAW SCREEN BORDER
2720 PRINT "SHIFT CLR";
2730 POKE 53281,12:POKE 53280,BC(SC):POK
E 53282,BC(SC)
2740 PRINT "40SPACE";
2750 FOR I=1 TO 22
2760 PRINT "SPACE";
NEXT
2770 NEXT
2780 PRINT "40SPACE";
2790 PRINT "39SPACE";:POKE 2023,96
2800 L=LEN(ST$(SC)):GOSUB4960
2810 FOR I=1 TO L
2820 IF ASC(MID$(ST$(SC),I,1))=32 THEN28
50
2830 POKE 1024+X+I,ASC(MID$(ST$(SC),I,1)
)
2840 POKE 55296+X+I,PEEK(646)
2850 NEXT
2860 L=LEN(T$):GOSUB4960:X=X-8
2870 FOR I=1 TO L
2880 IF ASC(MID$(T$,I,1))=32 THEN2930
2890 POKE 1024+(I*40)+(X*40),ASC(MID$(T$
,I,1))
2900 POKE 55296+(I*40)+(X*40),PEEK(646)
2910 POKE 1063+(I*40)+(X*40),ASC(MID$(T$
,I,1))
2920 POKE 55335+(I*40)+(X*40),PEEK(646)
2930 NEXT
2940 RETURN
2950 REM START NOTE
2960 V=(V-1)*7
2970 FOR I=54272 TO 54296
2980 POKE I,0
2990 NEXT
3000 POKE 54296,15:POKE 54277+V,129:POKE
54278+V,68
3010 POKE 54273+V,H:POKE 54272+V,L
3020 POKE 54276+V,17:RETURN
3030 REM END NOTE
3040 V=(V-1)*7
3050 POKE 54276+V,16:POKE 54296,0:RETURN
3060 REM HILIGHT A PART OF THE SCREEN
3070 POKE 53283,C
3080 FOR I=1 TO L
3090 POKE X+I,PEEK(X+I) OR 128:POKE X+54
272+I,1
3100 NEXT
3110 RETURN
3120 REM DE-HILIGHT PART OF THE SCREEN
3130 FOR I=1 TO L
3140 POKE X+I,PEEK(X+I) AND 127:POKE X+5
4272+I,0
3150 NEXT
3160 RETURN
3170 REM PRINT A MESSAGE AT BOTTOM OF SC
REEN
3180 SS=" "
3190 FOR I=1 TO LEN(M$)
3200 X=ASC(MID$(M$,I,1)):X=X+32:IF X<96
THEN X=X+96
3210 SS=SS+CHR$(X)
3220 NEXT
3230 PRINT SS:RETURN
3240 REM INPUT ROUTINE
3250 Z=0
3260 SS=W$(W):V=PEEK(214)*40+PEEK(211)+
1024:POKE 213,L+PEEK(211)
3270 GOSUB3580
3280 IF (ASC(K$)=13 AND SS<>"") OR (ASC(
K$)=13 AND (SC=2 OR SC=3)) THEN3530
3290 IF (SC=2 OR SC=3) AND (ASC(K$)>33 A
ND ASC(K$)<45) THEN3270
3300 IF (SC=2 OR SC=3) AND (ASC(K$)>57 A
ND ASC(K$)<65) THEN3270
3310 IF SC=5 AND (ASC(K$)=133 OR ASC(K$)
=134 OR ASC(K$)=136) THEN3530
3320 IF SC<>1 OR (SS=" " AND W$(W+1)<>"
") THEN3380
3330 IF ASC(K$)=145 AND W>1 THEN3530
3340 IF ASC(K$)=17 AND SS<>" " THEN3530
3350 IF ASC(K$)=136 THEN3570
3360 IF ASC(K$)=133 AND PEEK(1945)=70 TH
EN3530
3370 IF ASC(K$)=134 AND PEEK(1965)=70 TH
EN3530
3380 GOSUB3390:GOTO3270
3390 IF Z=0 THEN3420
3400 IF ASC(K$)=20 THEN K$="":I=-1:GOTO3
490
3410 IF ASC(K$)=157 AND SC<>5 THEN I=-1:
GOTO3520
3420 IF Z=L THEN3570
3430 IF Z=LEN(SS) THEN3460
3440 IF ASC(K$)=29 THEN I=1:GOTO3520
3450 IF ASC(K$)=148 AND LEN(SS)<L THEN K
$="+MID$(SS,Z+1,1):GOTO3480
3460 IF (ASC(K$)<B OR ASC(K$)>T) AND K$<>"
" THEN3570
3470 I=0
3480 Z=Z+1
3490 SS=LEFT$(SS,Z-1)+K$+MID$(SS,Z+1)

```

Continued



```

4280 RETURN
4290 REM ANIMATE THE REVERSAL
4300 L=LEN(WL$(WW))
4310 C=4:SP=INT((40-(L*2))/2+1223)
4320 FOR I=1 TO R STEP 2
4330 POKE SP+40+I,PEEK(SP+I)
4340 POKE SP+1+32,POKE(SP+40+I+54272,0)
4350 POKE SP+40+(R*2)-1,PEEK(SP+(R*2)-1)
4360 POKE SP+1+(R*2)-1,32:POKE(SP+40+(R*2)-1)
4370 IF I>R-2 THEN C=2:IF R AND 1 THEN 4450
4380 FOR J=1 TO C
4390 X=2:Y=6:GOSUB 2690
4400 POKE 213,35:PRINT CHR$(148)
4410 POKE 216,0
4420 X=3:Y=4:GOSUB 2690
4430 POKE 213,35:PRINT CHR$(20)
4440 NEXT J
4450 NEXT I
4460 Y=4:GOSUB 3950
4470 Y=6:GOSUB 3950
4480 GOSUB 4000
4490 RETURN
4500 REM SCRAMBLE THE WORD'S LETTERS
4510 SW$="":RS=WL$(WW)
4520 FOR I=1 TO LEN(RS)-1
4530 L=LEN(RS)
4540 R=RND(0)*L+1
4550 SW$=SW$+MID$(RS,R,1)
4560 RS=LEFT$(RS,R-1)+MID$(RS,R+1)
4570 NEXT I
4580 SW$=RS+SW$:IF SW$=WL$(WW) THEN 4510
4590 RETURN
4600 RETURN
4610 REM WORD IS CORRECTLY REVERSED
4620 S=54272
4630 POKE S+5,9:POKE S+8,50
4640 POKE S+6,240
4650 POKE S+4,17
4660 POKE S+24,15
4670 FOR I=10 TO 200 STEP 10:POKE S+1,I

```

```

4680 NEXT I=200 TO 75 STEP -10:POKE S+1,I
4690 NEXT I=75 TO 50 STEP -7:POKE S+1,I
4700 NEXT I=10 TO 210 STEP 20:POKE S+1,I
4710 NEXT I=10 TO 24:POKE S+1,0:NEXT:A=0
4720 POKE S+4,16:POKE S+24,15
4730 FOR I=1 TO 24:POKE S+1,0:NEXT:A=0
4740 IF ME THEN MS="YOU'VE DONE IT IN"+STR$(NT)+REVERSAL
4750 IF ME=0 THEN MS="REVERSAL"
4760 IF NT>1 THEN MS=MS+"S"
4770 X=4:Y=11:GOSUB 2690:GOSUB 3170
4780 FOR I=1 TO 2200:NEXT:Y=11:GOSUB 3950
4790 FOR Y=5 TO 9:GOSUB 3950:NEXT
4800 ME=0:RETURN
4810 REM AUTO SOLVE
4820 ME=1:X=35:GOSUB 4130
4830 A=1:R=0:L=LEN(SW$)
4840 IF MID$(WL$(WW),L)=MID$(SW$,L) THEN L=L-1:GOTO 4870
4850 S$=MID$(WL$(WW),L,1)
4860 FOR I=1 TO L
4870 IF R<V0 THEN 4920
4880 IF MID$(SW$,I,1)=S$ THEN R=I
4890 NEXT I
4900 IF R=1 THEN R=L
4910 L=R:GOSUB 4170
4920 S$="I'VE":GOTO 2410
4930 REM FIND SCREEN CENTER
4940 X=INT((37-L)/2):RETURN
4950 REM DATA FOR SCREEN-TITLE,COLOR,NOT E
4960 DATA MENU,3,0,0,CREATE WORD LIST,6,34,75,LOAD WORD LIST,5,38,126
4970 DATA SAVE WORD LIST,9,43,52,PRINT WORD LIST,4,45,198
4980 DATA PLAY SWITCH N SPELL,14,51,97,E
4990 XIT PROGRAM,2,57,172

```

HCM

## SWITCH 'N' SPELL

IBM PC &amp; IBM PCjr

```

1000 * * * * *
1010 * SWITCH 'N' SPELL *
1020 * * * * *
1030 COPYRIGHT 1985
1040 EMERALD VALLEY PUBLISHING CO.
1050 BY RANDY THOMPSON
1060 AND THE HCM STAFF
1070 HOME COMPUTER MAGAZINE
1080 VERSION 5.2.1
1090 IBM PCjr, W/CARTRIDGE BASIC
1100 FROM DOS 2.1 or
1110 IBM PC W/BASICA and
1120 COLOR/GRAPHICS ADAPTER and
1130 COLOR MONITOR
1140
1150 DIM C(3,16):LOGO$="SWITCH 'N' SPELL"
1160
1170 CLS:SCREEN 1:RANDOMIZE TIMER
1180 KEY 9,"SCREEN 2:CLS:"+CHR$(13):KEY
1190 10,"X"+CHR$(13)
1200 BX=79:BY=14:CB=10:DIM ARRAY(1000),W
1210 DS(50)
1220 KEY 1,CHR$(1):KEY 2,CHR$(2):KEY 3,C
1230 HR$(3)
1240 TRUE=-1:FALSE=0:MAXWORD=16
1250 TITLES$="SWITCH 'N' SPELL":SCREEN 1,
1260 0,0:CLS:KEY OFF
1270 CLS:LOCATE 1,10:PRINT TITLES:POKE 1
1280 050,PEEK(1052)
1290 LOCATE 3,16:PRINT "MENU"
1300 IF WW.MAX=0 THEN MENU.NUM=2 ELSE ME
1310 NU.NUM=6
1320 RESTORE:FOR I=1 TO MENU.NUM:READ
1330 MENU.OPTION$:LOCATE I*2+6,8:K$=READ
1340 ":RSET K$=STR$(I)+":PRINT K$;"
1350 ":MENU.OPTION$:NEXT I
1360 LOCATE 23,9:PRINT "SELECTION?"
1370 K$="":WHILE K$<"1" OR K$>"RIGHT$(STR
1380 $(MENU.NUM),1):K$=INKEY$:WEND
1390 ON VAL(K$) GOSUB 910,1050,1010,1830
1400 1240,1090:GOTO 310
1410 REM DRAW WORD EDIT PAD
1420 CLS:FOR I=1 TO S.RANGE TO S.RANGE+9:LI
1430 NE(21,(I-1)*16+10)-(298,(I-1)*16+1
1440 0):3:NEXT I
1450 GOSUB 1480
1460 FOR I=1 TO S.RANGE TO S.RANGE+9:X$="
1470 ":RSET X$=STR$(I)+":LOCATE ((I
1480 -1) MOD 10)+1)*2+1,4:PRINT X$:LOCAT
1490 E(((I-1) MOD 10)+1)*2+1,CB+1:PRINT
1500 WDS(I):SPACES$(MAXWORD-LEN(WDS(I))+
1510 1):NEXT I
1520 RETURN
1530 REM INPUT ROUTINE
1540 CX=1:CY=1:GOSUB 700:SR=S.RANGE-1
1550 K$="":WHILE K$="":K$=INKEY$:WEND:L=
1560 LEN(K$):K=ASC(K$):WDL=LEN(WDS(CY+SR
1570 ))
1580 IF LEN(K$)=2 THEN ON INSTR("KMPHRSG
1590 O",RIGHT$(K$,1)) GOSUB 600,610,620,
1600 630,640,650,660,680:GOTO 690

```

```

1610 IF K>65 THEN 530
1620 IF K=13 THEN IF CY=10 THEN GOSUB 71
1630 0:CY=1:CX=1:GOSUB 700:INSERTING=FA
1640 SE:GOTO 690 ELSE GOSUB 710:GOSUB 62
1650 0:INSERTING=FALSE:GOTO 690
1660 IF K=8 AND WDL>0 THEN J=CY+SR:IF CX
1670 >1 THEN WDS(J)=LEFT$(WDS(J),CX-2)+M
1680 ID$(WDS(J),CX):LOCATE CY+2+1,CB+1:P
1690 RINT WDS(J):SPACES$(MAXWORD-WDL+2):
1700 GOSUB 710:CX=CX-1:GOSUB 700:GOTO 69
1710 0 ELSE GOSUB 660:GOTO 690
1720 IF K=1 THEN RETURN
1730 IF K=2 AND S.RANGE < 41 THEN S.RANG
1740 E=S.RANGE+10:GOSUB 720:GOSUB 710:GO
1750 SUB 410:CX=1:CY=1:GOSUB 700:SR=S.RA
1760 NGE=1:GOTO 690
1770 IF K=3 AND S.RANGE > 1 THEN S.RANGE
1780 =S.RANGE-10:GOSUB 720:GOSUB 710:GOS
1790 UB 410:CX=1:CY=1:GOSUB 700:SR=S.RAN
1800 GE=1:GOTO 690
1810 IF K$>"a" AND K$<"z" THEN K$=CHR$
1820 (65+(ASC(K$)-97)) ELSE IF (K$="A" O
1830 R K$>"Z") AND K$<"-" THEN 690
1840 IF NOT INSERTING OR CX>WDL THEN INS
1850 ERTING=FALSE:GOTO 580 ELSE X$=WDS(C
1860 Y+SR)
1870 IF WDL>MAXWORD THEN BEEP:GOTO 690
1880 IF CX=1 THEN WDS(CY+SR)=K$+X$ ELSE
1890 WDS(CY+SR)=LEFT$(X$,CX-1)+K$+MID$(X
1900 $,CX)
1910 GOSUB 710:LOCATE CY*2+1,CB+1:PRINT
1920 WDS(CY):CX=CX+1:GOSUB 700:GOTO 690
1930 IF CX<MAXWORD THEN LOCATE CY+2+1,C
1940 X+CB:PRINT K$:GOSUB 710:CX=CX+1:GO
1950 SUB 700:IF CX<WDL+1 THEN MID$(WDS(
1960 CY+SR),CX-1)=K$ ELSE WDS(CY+SR)=WDS
1970 (CY+SR)+K$
1980 GOTO 690
1990 IF CX>1 THEN GOSUB 710:CX=CX-1:GOSU
2000 B 700:RETURN ELSE RETURN
2010 IF CX<LEN(WDS(CY+SR))+1 THEN GOSUB
2020 710:CX=CX+1:GOSUB 700:RETURN ELSE R
2030 ETURN
2040 IF CY<10 THEN GOSUB 710:CY=CY+1:GOS
2050 UB 1570:CX=1:GOSUB 700:INSERTING=FA
2060 LSE:RETURN ELSE RETURN
2070 IF CY>1 THEN GOSUB 710:CY=CY-1:CX=1
2080 :GOSUB 700:INSERTING=FALSE:RETURN E
2090 LSE RETURN
2100 IF INSERTING=FALSE THEN INSERTING=T
2110 RUE:RETURN ELSE INSERTING=FALSE:RET
2120 URN
2130 DEL
2140 FLAG=FALSE:IF LEN(WDS(CY))>0 THEN F
2150 LAG=TRUE:IF CX>1 THEN WDS(CY+SR)=LE
2160 FT$(WDS(CY+SR),CX-1)+MID$(WDS(CY+SR
2170 ),CX+1) ELSE WDS(CY+SR)=MID$(WDS(CY
2180 +SR),CX+1)
2190 IF FLAG THEN LOCATE CY*2+1,CB+1:PRI
2200 NT WDS(CY+SR):SPACES$(MAXWORD-WDL+2)
2210 :GOSUB 700:RETURN

```

Continued



```

680 GOSUB 710: CX=LEN(WDS(CY))+1: GOSUB 7
690 RETURN
700 GOTO 450
710 XS=BX+(CX-1)*8: YS=BY+(CY-1)*16: LINE
(XS,YS)-(XS+8,YS+16),3,B: RETURN
720 XS=BX+(CX-1)*8: YS=BY+(CY-1)*16: LINE
(XS,YS)-(XS+8,YS+16),0,B: RETURN
730 REM LAY UP FUNCTION ON KEY MENU
I=S.RANGE-10: J=I+9: K=S.RANGE+10: L=K
+9
740 LINE (0,170)-(319,199),0,BF
750 FUNC2$=""
760 IS=MIDS(STR$(I),2): JS=MIDS(STR$(J),
2): KS=MIDS(STR$(K),2): LS=MIDS(STR$(
L),2)
770 IF S.RANGE>41 THEN 890
780 FUNC$="(F1) RETURN TO MENU (F2)
Words="+KS+"-"+LS
790 IF S.RANGE>10 THEN FUNC2$=""
(F3) Words="+IS+"-"+JS
800 LOCATE 23,1: PRINT FUNC$: LOCATE 25,
1: PRINT FUNC2$:
810 GET (0,170)-(319,199),ARRAY: PUT (0,
170),ARRAY,PRESET
820 RETURN
830 FUNC$="" (F1) RETURN TO MEN
U"
840 FUNC2$="" (F2) AUTO-SOLVE"
850 FUNC3$="" (F3) VIEW UNSCRAM
BLED WORD"
860 LOCATE 23,1: PRINT FUNC$: LOCATE 24,
1: PRINT FUNC2$: LOCATE 25,1: PRINT F
UNC3$:
870 GET (0,170)-(319,199),ARRAY: PUT (0,
170),ARRAY,PRESET
880 RETURN
890 FUNC$="" (F1) RETURN TO MENU (F3)
WORDS="+IS+"-"+JS: GOTO 800
900 DATA "CREATE WORD LIST", "LOAD WORD
LIST", "SAVE WORD LIST", "PRINT WORD
LIST", "PLAY SWITCH 'N' SPELL", "EXIT
PROGRAM"
910 REM CREATE WORD LIST
920 CLS
930 NEW WORDS=TRUE
940 LOCATE 1,10: PRINT TITLES
950 IF WW.MAX=0 THEN KS="2": GOTO 990
960 LOCATE 5,1: PRINT "DO YOU WANT TO
...": LOCATE 10,5: PRINT "[1] UPDATE T
HE CURRENT WORD LIST": LOCATE 13,18:
PRINT "OR": LOCATE 16,5: PRINT "[2] C
REATE A NEW WORD LIST"
970 KS="": WHILE KS<>"1" AND KS<>"2": KS=
INKEY$: WEND
980 IF KS="1" THEN MODE$="EDIT WORD LIS
T": IF WW.MAX=0 THEN KS="2": GOTO 990
ELSE GOTO 770
990 IF KS="2" THEN MODE$="CREATE WORD L
IST": FOR I=1 TO 50: WDS(I)="" : NEXT
I
1000 S.RANGE=1: E.RANGE=10: GOSUB 380: GOSU
B 720: GOSUB 430: GOSUB 2200: RETURN
1010 REM
1020 MODE$="SAVE WORD LIST": GOSUB 1470: G
OSUB 1540
1030 GOSUB 1110: ON ERROR GOTO 1590
1040 OPEN FILENAMES FOR OUTPUT AS #1: FOR
I=1 TO 50: PRINT #1, WDS(I): NEXT I
: PRINT #1, WW.MAX: CLOSE #1: NEW WORDS
=FALSE: ON ERROR GOTO 0: RETURN
1050 REM
1060 LOCATE 5,1: MODE$="LOAD WORD LIST": G
OSUB 1470: GOSUB 1540: LOCATE 5,10: PR
INT SPACES(20)
1070 GOSUB 1110: ON ERROR GOTO 1630
1080 OPEN FILENAMES FOR INPUT AS #1: FOR
I=1 TO 50: INPUT #1, WDS(I): NEXT I
: INPUT #1, WW.MAX: CLOSE #1: NEW WORDS=
FALSE: ON ERROR GOTO 0: RETURN
1090 CLS: IF NEW WORDS THEN PRINT: PRINT "
DO YOU WANT TO EXIT WITHOUT": INPUT
"SAVING THE WORD LIST (Y/N)": YNS:
IF YNS<>"Y" AND YNS<>"Y" THEN RETUR
N
1100 CLS: PRINT "EXITING SWITCH 'N' SPELL
...": PRINT: PRINT "GOODBYE": END
1110 K=7: GOSUB 1170: POKE 1050, PEEK(1052)
: FILES$="" : INPUT "FILENAME": FILES: IF
FILES$="X" OR FILES$="" THEN ON ERRO
R GOTO 0: RETURN 1200 ELSE CLN=INSTR
(FILES,"")
1120 IF CLN<>0 THEN DEVICES$=LEFT$(F$,CLN
): FILES$=MIDS(FILES,CLN+1)
1130 DOT=INSTR(FILES,"."): IF DOT>1 THEN
FILES$=LEFT$(FILES,DOT-1)
1140 IF LEN(FILES)>9 THEN K=10: GOSUB 117
0: PRINT "ERROR: FILENAME TOO LONG. I
...": LOCATE 11,5: PRINT "9 CHARACTERS M
AXIMUM": LOCATE 12,5: PRINT "PRESS A
KEY TO CONTINUE": GOSUB 1190: GOSUB
1180: GOTO 1110
1150 XS=DEVICES$: GOSUB 1220: DEVICES$=XS: XS
=FILES$: GOSUB 1220: FILES$=XS: IF DEVIC
ES<>"B" AND DEVICES<>"A": AND DEVIC
ES<>"B" AND DEVICES<>"E": THEN K=10
: GOSUB 1170: PRINT "ILLEGAL DEVICE."
: LOCATE 11,5: PRINT "PRESS A KEY TO
CONTINUE": GOSUB 1190: GOSUB 1180: G
OTO

```

```

1160 FILENAMES$=DEVICES$+FILES$+".SWS": K=10
: GOSUB 1170: PRINT "VALIDATED AS: "
: FILENAMES$="": LOCATE 11,5: PRINT "O
KAY? (Y/N)": GOSUB 2090: IF YNS<>"Y"
AND YNS<>"Y" THEN GOSUB 1180: GOTO
1110 ELSE RETURN
1170 LOCATE K,5: PRINT SPACES(33): LOCATE
K+1,5: PRINT SPACES(33): LOCATE K,5
: RETURN
1180 FOR I=8 TO 20: LOCATE I,5: PRINT S
PACES(33): NEXT I: RETURN
1190 KS$="" : WHILE KS$="" : KS$=INKEY$: WEND: IF
KS$="X" THEN RETURN 1210 ELSE RETUR
N
1200 RETURN
1210 FOR I=1 TO LEN(XS): YS=MIDS(XS,I,1):
IF ASC(MIDS(XS,I,1))>=97 THEN MIDS(
XS,I,1)=CHR$(65+ASC(YS)-97)
1230 NEXT I: RETURN
1240 IY$="YOU": ON ERROR GOTO 0: SCREEN 1
: CLS: MODE$="PLAY SWITCH 'N' SPELL":
GOSUB 1470: GOSUB 830
1250 DONE=FALSE: COUNT=0: WW=WW+1: IF WW>WW
.MAX THEN WW=1
1260 GOSUB 2080: SWS$="" : WORDS$=WDS(WW): ORI
GINALS$=WORDS$: INIT=TRUE
1270 FOR I=1 TO LEN(WORDS)-1: L=LEN(WOR
DS): R=INT(RND(0)*L+1): SWS$=SWS+MIDS(
WORDS,R,1): WORDS$=LEFT$(WORDS,R-1)+M
IDS(WORDS,R+1): NEXT SWS=WORDS+SWS: I
F SWS=ORIGINALS THEN 1260 ELSE WORD
S$=SWS
1280 T=INT((42-LEN(WORDS)*2)/2): L1=4: L2=
L1+2: L3=L2+2
1290 GOSUB 2060
1300 K=0: FOR I=1 TO LEN(B$) STEP 2: LOC
ATE L3+1, T+1-1: K=K+1: PRINT CHR$(24)
: LOCATE L3+2, T+1-2: IF K<10 THEN PR1
NT K ELSE PRINT INT(K/10): LOCATE L3
+3, T+1-2: PRINT K MOD 10
1310 NEXT I: LOCATE L2, T: PRINT B$
1320 IF INIT THEN LOCATE L3+7, 10: PRINT "
THE WORD IS:
...": LOCATE
L3+8, 10: PRINT ORIGINALS: FOR I=1
TO 30: NEXT I: INIT=FALSE
1330 LOCATE L3+7, 5: PRINT SPACES(33): LOCA
TE L3+8, 5: PRINT SPACES(33)
1340 LOCATE L3+7, 10: PRINT "NUMBER TO REV
ERSE?": GOSUB 1710: IF KA=-1 THEN G
OSUB 1960: GOTO 1250 ELSE IF KA=-3 T
HEN INIT=TRUE: GOTO 1320: ELSE IF K
A=-2 THEN RETURN ELSE SN=D: LOCATE
L3+7, 5: PRINT SPACES(33)
1350 IF SN<2 OR SN>K THEN LOCATE L3+5, 5:
PRINT "RANGE IS 2 THRU": K: LOCATE L3
+9, 5: PRINT "PRESS A KEY TO CONTINUE
...": WHILE INKEY$="" : WEND: LOCATE L3
+9, 5: PRINT SPACES(33): LOCATE L3+5, 5
: PRINT SPACES(33): GOTO 1340
1360 GOSUB 1370: IF DONE THEN 1250 ELSE G
OTO 1280
1370 TEMPS$="" : FOR I=SN TO 1 STEP -1: TE
MPS$=TEMPS+MIDS(WORDS,I,1): NEXT I: WO
RDS$=TEMPS+MIDS(WORDS,SN+1)
1380 BS$=LEFT$(BS,SN*2-1): WL=LEN(BS): A=LE
N(BS)-1: B=0: C=0
1390 AS$=LEFT$(BS,1)+AS: CS$=CS+RIGHT$(BS,1
): BS$=MIDS(BS,2,LEN(BS)-2): GOTO 1400
1400 B=B+1: GOSUB 1410: A=A-1: C=C+1: GOSUB
1410: A=A-1: C=C+1: GOSUB 1410: C=C-1: I
F LEN(BS)>2 THEN 1390 ELSE AS$=BS+A
$: GOTO 1430
1410 LOCATE L1, T+A+LEN(C$): PRINT "": : L
OCATE L2, T+B+LEN(B$): PRINT "": : LOC
ATE L2, T+B-1: PRINT "": : LOCATE L3, T
+C-1: PRINT "":
1420 LOCATE L1, T+A: PRINT C$: : LOCATE L2, T
+B: PRINT B$: : LOCATE L3, T+C: PRINT AS
: RETURN
1430 FOR I=1 TO 100: NEXT I
1440 FOR I=1 TO LEN(C$): LOCATE L1, T+I-
1: PRINT "": LOCATE L2, T+I-1: PRINT M
IDS(C$,I,1): FOR J=1 TO 50: NEXT J
: I: FOR I=1 TO LEN(AS): LOCATE L3, T+
C+I-1: PRINT "": LOCATE L2, T+C+I-1: P
RINT MIDS(AS,I,1): FOR J=1 TO 50: N
EXT J, I
1450 COUNT=COUNT+1: GOSUB 2250: IF WORDS$=O
RIGINALS THEN GOSUB 1700: LOCATE 15,
5: PRINT IY$+"GOT THE WORD IN": LOCAT
E 16, 5: PRINT COUNT: TS: FOR J=1 TO
3000: NEXT J: GOSUB 1580: DONE=TRUE: IY
$="YOU"
1460 RETURN
1470 CLS
1480 LOCATE 1, INT((40-LEN(MODES))/2): PRI
NT MODE$: XS$=LEFT$(MODES,4): IF XS$="L
OAD" OR XS$="SAVE" THEN RETURN
1490 GET (0,0)-(319,8),ARRAY: PUT (0,0),A
RRAY: PUT (0,2),ARRAY: GET (0,0)-(319
,9),ARRAY: PUT (0,1),ARRAY: PRESET
LINE (0,11)-(21,169),0,BF: LINE (21,
11)-(21,169),3: LINE (298,11)-(319,1
69),0,BF: LINE (298,11)-(298,169),3
1510 FOR I=1 TO LEN(LOGO$): LOCATE I+3,
2: PRINT MIDS(LOGO$,I,1): LOCATE I+3,
59: PRINT MIDS(LOGO$,I,1): NEXT I

```

*Continued*



# SWITCH 'N' SPELL *Continued*

```

1520 GET (0,11)-(21,168),ARRAY:PUT (0,11
9),ARRAY:PRESET:GET (298,11)-(319,16
1530 RETURN PUT (298,11),ARRAY:PRESET
1540 LOCATE 23,10:PRINT "(F10) RETURN TO
MENU"
1550 GET (0,170)-(319,190),ARRAY:PUT (0,
170),ARRAY:PRESET:RETURN
1560 LINE (0,170)-(319,190),3,BF:RETURN
1570 IF WW.MAX<CY+SR THEN WW.MAX=CY+SR
:RETURN ELSE RETURN
1580 LINE (22,12)-(297,168),0,BF:RETURN
1590 LOCATE 14,5:PRINT "ERROR SAVING ";
FILENAMES:
1600 LOCATE 16,5:PRINT "ERR: ";ERR;" PRE
SS A KEY"
1610 LOCATE 17,5:PRINT " " TO CON
TINUE...:GOSUB 1670
1620 LOCATE 14,5:PRINT SPACES(33):LOCATE
16,5:PRINT SPACES(33):LOCATE 17,5:
PRINT SPACES(33):GOSUB 1580:RESUME
1030
1630 LOCATE 14,5:PRINT "ERROR LOADING "
:FILENAMES:
1640 LOCATE 16,5:PRINT "ERR: ";ERR;" PRE
SS A KEY"
1650 LOCATE 17,5:PRINT " " TO CON
TINUE...:GOSUB 1670
1660 LOCATE 14,5:PRINT SPACES(33):LOCATE
16,5:PRINT SPACES(33):LOCATE 17,5:
PRINT SPACES(33):GOSUB 1580:RESUME
1070
1670 K$="":WHILE K$="" :K$=INKEY$:WEND:IF
K$="X" THEN K$=INKEY$:RESUME 1680
ELSE RETURN
1680 RETURN 1690
1690 RETURN
1700 PLAY "T3216402AO3C#EO2A03C#e116e164
03ec#02ao3c#02ag116a":RETURN
1710 TTX=POS(0):TTY=CSRLIN
1720 NUMERIC=TRUE:LOCATE TTY,TTX:PRINT S
PACES(6):LOCATE TTY,TTX:GOSUB 1730:
IF KA<0 THEN RETURN ELSE IF D<1 THE
N 1710 ELSE RETURN
TTY=CSRLIN:TX=POS(0):K$="":D$=""
1730 K$="":WHILE K$="" :K$=INKEY$:WEND:KA
=ASC(K$)
1750 IF KA=3 THEN KA=-3:RETURN
1760 IF KA=2 THEN KA=-2:RETURN
1770 IF KA=1 THEN KA=-1:RETURN
1780 IF KA=13 THEN D=VAL(D$):NUMERIC=FAL
SE:RETURN
1790 IF KA=8 THEN IF LEN(D$)>0 THEN D$=L
EFT$(D$,LEN(D$)-1):TX=TX-1:LOCATE T
Y,TX:PRINT
740 ELSE GOTO 1740
IF LEN(D$)>5 THEN 1740
IF NUMERIC AND K$<"0" OR K$>"9" THE
N 1740
1820 PRINT K$;:TX=TX+1:D$=D$+K$:GOTO 174
0
1830 CLS:MODE$="PRINT WORD LIST":GOSUB 1
480
1840 LOCATE 5,4:PRINT "DO YOU WANT TO...
":LOCATE 10,4:PRINT "[1] PRINT WORD
LIST TO THE SCREEN:LOCATE 13,18:P
RINT "OR":LOCATE 16,4:PRINT "[2] PR
INT WORD LIST TO THE PRINTER"[2] PR
K$="":WHILE K$<>"1" AND K$<>"2":K$=
INKEY$:WEND
1860 IF K$="1" THEN GOSUB 2110:RETURN

```

```

1870 LOCATE 5,4:PRINT SPACES(18):LOCATE
10,4:PRINT SPACES(34):LOCATE 13,18:
PRINT "LOCATE 16,4:PRINT SPACES
(34)"
1880 LOCATE 5,4:PRINT "PRESS RETURN WHEN
PRINTER:LOCATE 7,5:PRINT "IS READ
Y:":WHILE INKEY$="" :WEND
1890 OPEN "LPT1:" FOR OUTPUT AS #1
1900 LOCATE 9,5:PRINT "PRINTING..."
1910 FOR I=1 TO 50:IF LEN(WDS(I))>1 TH
EN PRINT #1, USING "WORD I##:
NEXT I
1920 CLOSE #1:LOCATE 22,5:PRINT "PRESS A
KEY TO CONTINUE"
1930 WHILE INKEY$="" :WEND:RETURN
1940 RETURN
1950 REM AUTO-SOLVE
1960 LOCATE L3+7,5:PRINT SPACES(33):LOCA
TE L3+6,5:PRINT "AUTO-SOLVE":IY$="I
"
1980 FOR Z=LEN(ORIGINALS) TO 1 STEP -1
:VS=MIDS(ORIGINALS,Z,1)
1990 IF VS=MIDS(WORDS,Z,1) THEN 2040
2000 J=INSTR(WORDS,VS):IF J=1 THEN 2030
2010 IF J>Z THEN 2040
2020 SN=J:GOSUB 1370:GOSUB 2060:IF DONE=
TRUE THEN 2050
2030 SN=Z:GOSUB 1370:GOSUB 2060:IF DONE=
TRUE THEN 2050
2040 NEXT Z
2050 LOCATE L3+6,5:PRINT SPACES(33):RETU
RN 1250
2060 NS="":AS=NS:BS=NS:CS=NS:X$=NS:FOR I
=1 TO LEN(WORDS):NS=NS+MIDS(WORDS
,I,1):
NEXT I:BS=LEFT$(NS,LEN(NS
)-1)
2070 RETURN
2080 PLAY "t44132o1bo218c132d18d#132o1bo
2116c53132d116d#132g18g":RETURN
2090 YN$="":WHILE YN$="" :YN$=INKEY$:WEND
:IF ASC(YN$)<>13 AND YN$<>"Y" AND YN
$<>"N" THEN 2090
EN 2090
IF ASC(YN$)=13 THEN YN$="Y":RETURN
ELSE RETURN
2110 PRINT WORD LIST TO SCREEN
2120 S.RANGE=1:E.RANGE=10:GOSUB 390:GOSU
B 720
2130 K$="":WHILE K$="" :K$=INKEY$:WEND
IF K$<>CHR$(1) AND K$<>CHR$(2) AND
K$<>CHR$(3) THEN 2130
IF K$=CHR$(1) THEN RETURN
IF K$=CHR$(2) AND S.RANGE<40 THEN S
.RANGE=S.RANGE+10:E.RANGE=E.RANGE+1
0:GOSUB 720:GOSUB 410:GOTO 2130
IF K$=CHR$(3) AND S.RANGE>10 THEN S
.RANGE=S.RANGE-10:GOSUB 720:GOSUB 4
10:GOTO 2130
2180 GOTO 2130
2190 LIST CLEANUP
2200 J=1:FOR I=1 TO 50:IF WDS(I)>" " THEN
WDS(J)=WDS(I):J=J+1
2210 IF LEN(WDS(I))=1 THEN WDS(I)=WDS(I)
+" "
2220 IF J=1 THEN WW.MAX=0
IF J-1<>1 THEN WDS(I)=WDS(I)
2230 NEXT J:WW.MAX=J-1:RETURN
2240 IF COUNT=1 THEN TS="REVERSAL" ELSE
TS="REVERSALS"
2250 RETURN
2260 RETURN

```

HCM

## SWITCH 'N' SPELL

TI-99/4A

```

100 REM *****
110 REM * SWITCH 'N' SPELL *
120 REM *****
130 REM COPYRIGHT 1985
140 REM EMERALD VALLEY PUBLISHING CO.
150 REM BY RANDY THOMPSON
160 REM AND THE HCM STAFF
170 REM HOME COMPUTER MAGAZINE
180 REM VERSION 5.2.1
190 REM TI BASIC OR
200 REM TI EXTENDED BASIC
210 REM
220 RANDOMIZE
230 CALL CLEAR
240 PRINT TAB(6);"SWITCH 'N' SPELL":
:PRESS ENTER TO CONTINUE"
250 GOSUB 3220
260 DIM WDS(50),MENUS(6)
270 RESTORE 3260
280 FOR Z=0 TO 6
290 READ MENUS(Z)
300 NEXT Z
310 CALL CLEAR
320 NM=6
330 IF NW>0 THEN 350
340 NM=2
350 FOR Z=0 TO NM
360 PRINT MENUS(Z):
370 NEXT Z
380 FOR Z=1 TO 9-NM
390 PRINT
400 NEXT Z

```

```

410 PRINT "ENTER YOUR CHOICE";
420 GOSUB 3220
430 IF (K<49)+(K>48+NM) THEN 420
440 ON K-48 GOSUB 460,1310,1310,1530,17
90,3130
450 GOTO 310
460 CALL CLEAR
470 IF NW=0 THEN 520
480 PRINT "CREATE WORD LIST":("1) A
DD TO CURRENT LIST":("2) BUILD NEW
LIST":
:ENTER YOUR CHOICE"
490 GOSUB 3220
500 IF (K<49)+(K>50) THEN 490
510 ON K-48 GOTO 560,520
520 FOR Z=1 TO 50
530 WDS(Z)=" "
540 NEXT Z
550 NW=0
560 TOP=1
570 CALL CLEAR
580 FOR Z=TOP TO TOP+9
590 PRINT STR$(Z);")";WDS(Z)
600 NEXT Z
610 PRINT
620 PRINT "A) ADD":("B) CHANGE":("C) NEXT
PAGE":("D) PREVIOUS PAGE":("E) RETUR
N TO MENU"
630 GOSUB 3220
640 IF (K<65)+(K>69) THEN 630
650 ON K-64 GOTO 660,990,1240,1270,1300
660 PRINT "ENTER WORD #":NW+1
670 FLG=0

```

Continued



```

680 INPUT AS="":AS
690 IF AS=" " THEN 570
700 FOR Z=1 TO LEN(AS)
710 TS=SEG$(AS,Z,1)
720 IF ((TS<"A")+(TS>"Z"))*(TS<>"-")THE
N 750
730 NEXT Z
740 GOTO 790
750 PRINT "ILLEGAL CHARACTERS ENTERED":
: "REDO ENTRY"
760 FOR TD=1 TO 300
770 NEXT TD
780 GOTO 570
790 P=POS(AS," ",1)
800 IF P=0 THEN 820
810 AS=SEG$(AS,1,P-1)
820 IF LEN(AS)<=15 THEN 890
830 CALL SOUND(100,110,0)
840 AS=SEG$(AS,1,15)
850 PRINT "WORD TRUNCATED TO:AS::"IS
THIS OK (Y/N)?";
860 GOSUB 3220
870 IF CHR$(K)="Y" THEN 890
880 GOTO 570
890 IF LEN(AS)>1 THEN 940
900 PRINT "MUST BE LONGER THAN ONE": "CH
ARACTER WORD":
910 FOR TD=1 TO 300
920 NEXT TD
930 GOTO 570
940 IF FLG=1 THEN 1220
950 NW=NW+1
960 WDS(NW)=AS
970 TOP=INT(NW/10)*10+1
980 GOTO 570
990 IF NW=0 THEN 570
1000 PRINT "CHANGE WHICH ONE?"
1010 INPUT " ":BS
1020 IF BS=" " THEN 570
1030 FOR Z=1 TO LEN(BS)
1040 IF (SEG$(BS,Z,1)>="0")*(SEG$(BS,Z,1)
<="9") THEN 1110
1050 NEXT Z
1060 CALL SOUND(100,110,0)
1070 PRINT "MUST ENTER A NUMBER"
1080 FOR TD=1 TO 200
1090 NEXT TD
1100 GOTO 570
1110 IF (VAL(BS)>0)*(VAL(BS)<=NW) THEN 11
70
1120 CALL SOUND(100,110,0)
1130 PRINT "MUST BE BETWEEN 1 AND";NW
1140 FOR TD=1 TO 300
1150 NEXT TD
1160 GOTO 570
1170 PRINT "OLD WORD: ";WDS(VAL(BS))
1180 PRINT "NEW WORD: ";
1190 FLG=1
1200 TOP=INT(VAL(BS)/10)*10+1
1210 GOTO 680
1220 WDS(VAL(BS))=AS
1230 GOTO 570
1240 IF TOP=41 THEN 570
1250 TOP=TOP+10
1260 GOTO 570
1270 IF TOP=1 THEN 570
1280 TOP=TOP-10
1290 GOTO 570
1300 RETURN
1310 CALL CLEAR
1320 INPUT "DEVICE&FILE NAME: ":FS
1330 IF FS="CS1" THEN 1380
1340 IF SEG$(FS,1,3)<>"DSK" THEN 1450
1350 IF (SEG$(FS,4,1)<"1")+(SEG$(FS,4,1)
>"3") THEN 1450
1360 IF SEG$(FS,5,1)<>" " THEN 1450
1370 FS=SEG$(FS,1,15)
1380 IF K=51 THEN 1460
1390 OPEN #1:FS,INPUT, FIXED 64
1400 INPUT #1:NW
1410 FOR Z=1 TO NW
1420 INPUT #1:WDS(Z)
1430 NEXT Z
1440 CLOSE #1
1450 RETURN
1460 OPEN #1:FS,OUTPUT, FIXED 64
1470 PRINT #1:NW
1480 FOR Z=1 TO NW
1490 PRINT #1:WDS(Z)
1500 NEXT Z
1510 CLOSE #1
1520 RETURN
1530 CALL CLEAR
1540 PRINT "ENTER: "::1) LIST TO SCREEN"
::2) FOR PRINTOUT"
1550 GOSUB 3220
1560 IF (K<49)+(K>50) THEN 1550
1570 IF K=49 THEN 1640
1580 INPUT "ENTER PRINTER PARAMETER:
":PS
1590 OPEN #1:PS
1600 FOR Z=1 TO NW
1610 PRINT #1:"WORD #";Z;" " ;WDS(Z)
1620 NEXT Z
1630 RETURN
1640 TOP=1
1650 CALL CLEAR
1660 FOR Z=TOP TO TOP+9

```

```

1670 PRINT "WORD #";Z;" " ;WDS(Z)
1680 IF Z=NW THEN 1700
1690 NEXT Z
1700 PRINT "MORE (Y/N)?"
1710 GOSUB 3220
1720 IF CHR$(K)="Y" THEN 1750
1730 IF CHR$(K)<>"N" THEN 1710
1740 RETURN
1750 IF TOP+9>NW THEN 1640
1760 IF TOP=41 THEN 1640
1770 TOP=TOP+10
1780 GOTO 1650
1790 WORD=1
1800 CALL CLEAR
1810 PRINT "GETTING A WORD..."
1820 NT=0
1830 WS=WDS(WORD)
1840 SWS=" "
1850 FOR Z=1 TO LEN(WS)
1860 P=INT(RND*LEN(WS)+1)
1870 SWS=SWS&SEG$(WS,P,1)
1880 IF P>1 THEN 1910
1890 WS=SEG$(WS,2,LEN(WS)-1)
1900 GOTO 1950
1910 IF P<>LEN(WS) THEN 1940
1920 WS=SEG$(WS,1,LEN(WS)-1)
1930 GOTO 1950
1940 WS=SEG$(WS,1,P-1)&SEG$(WS,P+1,LEN(W
S)-P)
1950 NEXT Z
1960 WS=WDS(WORD)
1970 IF WS=SWS THEN 1840
1980 CALL CLEAR
1990 X=5
2000 RESTORE 3280
2010 FOR Y=12 TO 15
2020 READ AS
2030 AS=SEG$(AS,1,LEN(SWS))
2040 GOSUB 3090
2050 NEXT Y
2060 FOR Y=19 TO 21
2070 READ AS
2080 GOSUB 3090
2090 NEXT Y
2100 AS=SWS
2110 Y=10
2120 GOSUB 3090
2130 AS="ENTER NUMBER 1 TO "&STR$(LEN(SW
S))
Y=1
2140 GOSUB 3090
2150 AS="OR PRESS A, B OR C"
2160 Y=3
2170 GOSUB 3090
2180 GOSUB 3220
2190 IF (K>=49)*(K<=57) THEN 2450
2200 IF (K<65)+(K>67) THEN 2190
2210 ON K-64 GOTO 2230,2300,3120
2220 AS=WS
2230 Y=17
2240 GOSUB 3090
2250 FOR TD=1 TO 200
2260 NEXT TD
2270 CALL HCHAR(17,1,32,32)
2280 GOTO 2190
2290 A=1
2300 R=0
2310 L=LEN(SWS)
2320 IF SEG$(SWS,L,1)<>SEG$(WS,L,1) THEN
2360
2340 L=L-1
2350 GOTO 2330
2360 S=SEG$(WS,L,1)
2370 FOR I=1 TO L
2380 IF SEG$(SWS,I,1)=S THEN 2400
2390 NEXT I
2400 IF I>1 THEN 2420
2410 I=L
2420 L=I
2430 GOSUB 2600
2440 GOTO 2900
2450 K=CHR$(K)
2460 CALL HCHAR(3,25,K)
2470 IF LEN(SWS)<10 THEN 2530
2480 GOSUB 3220
2490 IF K=13 THEN 2530
2500 IF (K<49)+(K>57) THEN 2480
2510 K=K&CHR$(K)
2520 CALL HCHAR(3,26,K)
2530 I=VAL(K)
2540 IF (I<=LEN(SWS))*(I>1) THEN 2570
2550 CALL HCHAR(3,23,32,2)
2560 GOTO 2190
2570 A=0
2580 GOSUB 2600
2590 GOTO 2900
2600 WL=" "
2610 NWS=" "
2620 FOR Z=1 TO 1 STEP -1
2630 CH=ASC(SEG$(SWS,Z,1))
2640 CALL HCHAR(10,4+Z,32)
2650 IF Z-1<>WL THEN 2690
2660 ZZ=WL-1
2670 OS=1
2680 GOTO 2780
2690 FOR ZZ=Z-1 TO WL STEP SGN(WL-(Z-1))
2700 CALL HCHAR(9,ZZ+6,32)
2710 CALL HCHAR(9,ZZ+4,32)

```

*Continued*



```

2720 CALL HCHAR(9,ZZ+5,CH)
2730 NEXT ZZ
2740 IF WL>Z-1 THEN 2770
2750 OS=1
2760 GOTO 2780
2770 OS=-1
2780 CALL HCHAR(9,ZZ+5+OS,32)
2790 CALL HCHAR(8,ZZ+5+OS,CH)
2800 WL=WL+1
2810 NWS=NWS&CHR$(CH)
2820 NEXT Z
2830 SW=NWS&SEG$(SW$,I+1,LEN(SW$)-1)
2840 CALL HCHAR(8,1,32,32)
2850 AS=NWS
2860 X=5
2870 Y=10
2880 GOSUB 3090
2890 RETURN
2900 NT=NT+1
2910 IF SW$=W$ THEN 2930
2920 IF A=1 THEN 2300 ELSE 2190
2930 Y=2
2940 IF A=0 THEN 2970
2950 AS="I DID IT IN "
2960 GOTO 2980
2970 AS="YOU DID IT IN "
2980 AS=AS&"REVERSAL #"&STR$(NT)
2990 CALL HCHAR(1,1,32,96)
3000 GOSUB 3090
3010 IF WORD<NW THEN 3040
3020 WORD=1
3030 GOTO 3050
3040 WORD=WORD+1

```

```

3050 FOR TD=1 TO 50
3060 CALL SOUND(-100,INT(RND*TD)*60+440,0)
3070 NEXT TD
3080 GOTO 1800
3090 FOR Z=0 TO LEN(A$)-1
3100 CALL HCHAR(Y,X+Z,ASC(SEGS(A$,Z+1,1)))
3110 NEXT Z
3120 RETURN
3130 CALL CLEAR
3140 PRINT "ARE YOU SURE (Y/N)?"
3150 GOSUB 3220
3160 IF CHR$(K)="Y" THEN 3190
3170 IF CHR$(K)<>"N" THEN 3150
3180 RETURN
3190 CALL CLEAR
3200 PRINT "GOODBYE"::::::::::::::::::
3210 END
3220 CALL KEY(0,K,S)
3230 IF S=0 THEN 3220
3240 CALL SOUND(1,880,0)
3250 RETURN
3260 DATA "MENU",1) CREATE WORD
RD LIST,2) LOAD WORD LIST,3) SAVE WORD LIST,4) PRINT WORD LIST
3270 DATA 5) PLAY SWITCH 'N' SPELL,6) EXIT PROGRAM
3280 DATA "1111"
3290 DATA A) SEE WORD,B) AUTO SOLVE,C) RETURN TO MENU

```

## HOW

## LASERITHMETIC

## APPENDIX E // Family

[illegible][illegible]

**Continued**



# LASERITHMETIC

© Home Computer Magazine 1985      Volume 5, No. 2

**Continued**



```

940 GOSUB 1230: SC=SC+1: PRINT "HOME" TAB (20) "
950 SCORE="SC: CNT=0: GOSUB 1130: NEXT Z, F,
L
960 POKES 53269, PEEK (53269) AND NOT 32
970 PRINT "SHIFT CLR 6 CRSR DOWN 5 CRSR
IGHT GREAT JOB, CAPTAIN": PRINT "3 CR
SR DOWN YOUR SCORE IS SC
980 FOR W=1 TO 200: NEXT W: GOTO 1040
990 PRINT "SHIFT CLR": POKES 53269, PEEK (5
3269) AND NOT 32
1000 FOR W=1 TO 5: POKES 53281, 1: POKES 53280, 2: P
OKES 53281, 2: POKES 53280, 1: NEXT W
1010 FOR X=1 TO 350: NEXT X: POKES 53281, 1: POKES
3280, 0
1020 PRINT "6 CRSR DOWN 10 CRSR RIGHT CTRL
RED SPACE STATION DESTROYED"
1030 PRINT "3 CRSR DOWN 8 CRSR RIGHT YOUR S
CORE="SC: FOR W=1 TO 2000: NEXT W
1040 POKES 53269, PEEK (53269) AND NOT 32
1050 PRINT "3 CRSR DOWN 3 CRSR RIGHT DO YOU
WANT TO PLAY AGAIN? (Y/N)"
1060 GET X$: IF X$="Y" THEN 1060
1070 GET X$: IF X$="N" THEN 1070
1080 IF X$="Y" THEN SP=32: MA=254: GOTO 290
1090 IF X$="N" THEN SP=32: MA=254: GOTO 290
1100 POKES 53281, 6: POKES 53280, 14: PRINT "CMD
R BLUE SHIFT CLR": END
1110 PRINT "SHIFT CLR": POKES 53281, 0: POKES
53280, 0
1120 FOR X=1 TO 25: N=INT (RND (1) * (999) + 1024)
: POKEN, 46: POKEN+54272, 14: NEXT X
1130 PRINT "CMDR BLK HOME 4 CRSR DOWN 4
CRSR RIGHT CTRL RVSON CMDR V
CMDR CTRL RVSON OFF
1140 PRINT "CMDR RED 3 CRSR RIGHT CTRL R
VSON CMDR V
CMDR CTRL RVSON OFF
1150 PRINT "CMDR RED 2 CRSR RIGHT CTRL R
VSON CMDR W
SHIFT W
SHIFT W
SHIFT W
1160 PRINT "CMDR RED 3 CRSR RIGHT CTRL R
VSON CMDR F
CMDR D
CT
RL RVSON OFF
1170 PRINT "CMDR BLK 4 CRSR RIGHT CTRL R
VSON CMDR F
CMDR D
CTRL
RVSON OFF
1180 PRINT "8 CRSR RIGHT CTRL PUR CMDR L
1190 PRINT "8 CRSR RIGHT CTRL PUR CMDR L
(: RETURN
1200 POKES 53269, PEEK (53269) OR 32: POKES 53258
, MA: POKES 53259, 125: RETURN
1210 FOR A=1 TO 20: POKES 54296, 15: FOR T=1 TO 3: N
EXT: POKES 54296, 0: FOR T=1 TO 3: NEXT
1220 NEXT: RETURN

```

```

1230 LZS="": FORK=1 TO SP-11: LZS=LZS+FLS: NE
XTK
1240 BL$="": FORK=1 TO SP-11: BL$=BL$+CLS: NE
XTK
1250 POKES 54296, 15: POKES 54283, 129: POKES 5428
4, 16: POKES 54285, 240: POKES 54279, 243
1260 POKES 54280, 200: POKES 54279, 243
1270 FOR W=1 TO 20: PRINT "HOME" TAB (10) CRSR DOWN
CTRL WH
TAB (10) LZS: NEXT W
1280 PRINT "HOME" TAB (10) CRSR DOWN TAB (10) BL$
: POKES 54283, 0: POKES 54284, 0: GOSUB 1430:
RETURN
1290 FOR LL=54272 TO 45296: POKELL, 0: NEXT: PO
KES 54296, 15
1300 POKES 54284, 124: POKES 54285, 248: FOR A=1 T
O 57: STEP 3
1310 POKES 54280, M(A): POKES 54279, M(A+1): POK
ES 54283, 17: FOR T=1 TO M(A+2): NEXT
1320 POKES 54283, 16: FOR T=1 TO 25: NEXT
1330 GET W$: IF W$="CHR$(13)" THEN RETURN
1340 NEXT A: GOTO 1300
1350 DATA 18, 209, 100, 18, 209, 50, 18, 209, 100
, 25, 30, 400
1360 DATA 37, 162, 450, 33, 135, 50, 31, 165, 50,
28, 49, 50
1370 DATA 50, 60, 400, 37, 162, 450, 33, 135, 100
, 31, 165, 50
1380 DATA 28, 49, 50, 50, 60, 400, 37, 162, 450, 3
3, 135, 50
1390 DATA 31, 165, 100, 33, 135, 100, 28, 49, 550
1400 REM MOVE ALIEN
1410 POKES 53269, PEEK (53269) AND NOT 32: POKES
3258, MA: POKES 53259, 125
1420 POKES 53269, PEEK (53269) OR 32: RETURN
1430 POKES 53269, PEEK (53269) AND NOT 32
1440 POKES 53260, MA: POKES 53261, 125
1450 FOR WW=1 TO 5: POKES 53269, PEEK (53269) OR 6
4: FOR NN=1 TO 10: NEXT NN
1460 POKES 53269, PEEK (53269) AND NOT 64
1470 FOR NN=1 TO 10: NEXT: NEXT: GOSUB 1200: RET
URN
1480 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
1490 DATA 0, 0, 0, 7, 255, 224, 15, 255, 240, 15, 129, 24
1500 DATA 14, 60, 112, 14, 60, 112, 14, 60, 112
1510 DATA 15, 255, 240, 15, 255, 240, 15, 129, 24
0
1520 DATA 15, 129, 240, 15, 129, 240, 15, 129, 24
0
1530 DATA 15, 129, 240, 0, 0, 0, 0, 0, 0, 0, 0
1540 DATA 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
1550 DATA 0, 0, 6, 33, 12, 4, 49, 0, 96
1560 DATA 0, 0, 64, 0, 0, 0, 0, 0, 8, 0, 8, 0, 8
1570 DATA 16, 24, 0, 0, 56, 0, 4, 8, 0, 8
1580 DATA 0, 2, 16, 0, 0, 0, 0, 0, 2, 0, 8
1590 DATA 0, 0, 0, 48, 16, 32, 16, 112, 48
1600 DATA 16, 32, 0, 0, 0, 0, 4, 2, 8
1610 DATA 24, 12, 8, 1, 128, 12, 0, 0, 0, 0

```

HCM

## LASERITHMETIC

IBM PC &amp; IBM PCjr

```

100 ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** ** 
110 ** LASERITHMETIC **
120 ** ** ** 
130 COPYRIGHT 1985
140 EMERALD VALLEY PUBLISHING CO.
150 BY R. G. CHRISTENSEN
160 AND THE HCM STAFF
170 HOME COMPUTER MAGAZINE
180 VERSION 5.2.1
190 IBM PCjr W/CARTRIDGE BASIC
200 FROM DOS 2.1 or
210 IBM PC W/BASICA and
220 COLOR/GRAPHICS ADAPTER and
230 COLOR MONITOR
240
250 CLS: KEY OFF: SCREEN 1: COLOR 0,0: DEFIN
T A-Z: DIM M(60): LOCATE 12,12: PRINT
"LASERITHMETIC": LOCATE 22,7: PRINT
"PRESS ANY KEY TO START": PLAY "MF"
: GOSUB 620
260 CLS: LOCATE 5,1: PRINT "ENTER PROBLEM
TYPE": PRINT: PRINT "1) ADDITION"
: PRINT: PRINT "2) SUBTRACTION": PRIN
T: PRINT "3) MULTIPLICATION": PRIN
T: PRINT "4) DIVISION" AS <"1" OR AS >"4" THEN
270 GOSUB 640: IF AS <"1" OR AS >"4" THEN
280 CLS: LOCATE 5,1: PRINT "ENTER SKILL L
EVEL": PRINT: PRINT "1) EASY": PRIN
T: PRINT "2) HARD": PRINT: PRINT
3) HARDER": PRINT: PRINT
ARD
290 GOSUB 640: IF AS <"1" OR AS >"4" THEN
300 CLS: LOCATE 1,1: PRINT "OK CAPTAIN": P
RINT: PRINT "YOU'RE THE COMMANDER OF
THE SPACE STATION COLUMBIA.
: PRINT: PRINT "DESTROY THE ATTACKING
ALIENS BY LOADING YOUR LASER CANNO
N WITH THE CORRECT ANSWERS TO
THE PROBLEMS.
310 PRINT: PRINT "CAUTION! DON'T LET AN
ALIEN GET TOO CLOSE!": LOCAT
E 22,1: PRINT "PRESS ANY KEY TO CONT
INUE": SCORE=0

```

```

320 DRAW "BM300,155C2F2ND2RND2UENR3DR2N
D2RND2R2ND2FNE2D2RD3DRDRDLDLDLDBUSE
URNL7GRDGNL2GLHLUEL3FLDLDLDRDRDRDR
: GET (289,155)-(319,167), M: GOSUB 64
0
330 RANDOMIZE TIMER: FOR L=1 TO 4: J=1: CL
S: LOCATE 12,16: PRINT "ROUND": L: LOCA
TE 22,9: PRINT "PRESS ANY KEY TO STA
RT": GOSUB 620: MP=299
340 GOSUB 570: PUT (MP,55), M, PSET: FOR N=
1 TO 12: FOR REP=1 TO 4: N1=N
350 N2=INT (RND * SKILL * 4): IF N1 < N2 THEN S
WAP N1, N2
360 ON TYPE GOTO 370, 380, 390, 400
370 ANSWER=N1+N2: OF=43: GOTO 410
380 ANSWER=N1-N2: OF=45: GOTO 410
390 ANSWER=N1*N2: OF=42: GOTO 410
400 IF N1=0 OR N2=0 THEN 350 ELSE ANSWE
R=N1-N1=N1-N2: OF=47
410 LOCATE 4,4: PROBS=RIGHT$(STR$(N1), LE
N (STR$(N1))-1)+CHR$(OF)+RIGHT$(STR$(
N2), LEN (STR$(N2))-1)+": LINE(22,2
4)-(96,31), 1, BF: PRINT PROBS: UN$=
TIME=0: ANSWERS=RIGHT$(STR$(ANSWER)
, LEN (STR$(ANSWER))-1)
420 DEF SEG=0: POKE 1050, PEEK (1052)
430 AS=INKEY$: IF AS=CHR$(27) THEN 260 E
LSE TIME=TIME+1: IF TIME=300-(SKILL
30) AND UN$= THEN GOSUB 490: IF MP
<68 THEN 530
440 IF AS <"0" OR AS >"9" THEN 430
450 UN$=UN$+AS: PRINT AS: IF UN$ <> LEFT$(
ANSWER$, LEN (UN$)) THEN 480
460 IF UN$=ANSWER$ THEN GOSUB 500 ELSE
470 NEXT: NEXT: NEXT: CLS: LOCATE 12,1: PRIN
T "GREAT JOB CAPTAIN. MISSION IS A
SUCCESS": GOTO 560
480 SOUND 110, 10: LINE(22,24)-(96,31), 1,
BF: GOSUB 490: IF MP < 68 THEN 530 ELSE
490 TIME=0: FOR MM=1 TO 4: SOUND 220 * MM, 2
: NEXT: PUT (MP,55), M: MP=MP-12: PUT (MP
,55), M, PSET: RETURN
500 LINE(64,51)-(MP+4,59), 1: LINE(64,60)
-(MP+4,61), 2, B: LINE(64,70)-(MP+4,62
), 1

```

Continued



```

510 FOR Z=440 TO 1000 STEP 60: SOUND Z, LINE
520 NEXT Z: LINE (64,51)-(MP+4,59): LINE (64,70)-(
530 MP+4,62): LINE (MP,55): M:PSET: TIME=0: PRINT
540 "SCORE": RIT: RIT: RETURN TO 33: T=(Z-15)*20
550 PLAY MB: FOR Z=1 TO 33: T=(Z-15)*20
560 DRAW BM60,48: S=Z: TO 33: T=(Z-15)*20
570 SOUND RND*100+110,2: NEXT: PLAY "MF":
580 CAP=CAP+1: IF >15 THEN 550 ELSE M
590 P=299: GOSUB 570: GOTO 470
600 CLS: LOCATE 12,1: PRINT "YOUR SPACE S
610 TATION HAS BEEN DESTROYED"
620 LOCATE 20,1: PRINT "YOUR SCORE IS: "
630 RIT: LOCATE 22,1: PRINT "PRESS: CHR
640 S(17): CHR$(217): PRINT CONTINUE": GOSU
650 B 620: GOTO 260
660 CLS: FOR Z=1 TO 250: PSET (RND*319,RN
670 D*199),1: NEXT Z

```

```

580 AS="C3BM12,26:ERERERER2ER100FR2FRFRF2
590 DG2LGLGL2GL100HL2HLHL2UBM+16,-6E+0
600 BS="BM60,42C1DG4LGL2DGD2DGD2FD3R4C2
610 D3RD3F3RBU19LG3D3F3DGD2NU4LU5HD3LC1L
620 SULULELULELULELELERULERE2RERURURURU
630 L"
640 DRAW "XAS:XBS": PAINT (40,25),1,3:P
650 AINT (40,18),CHRS(&HDD)+CHRS(&H77),
660 5: PAINT (40,35),CHRS(&HDD)+CHRS(&H7
670 7),3
680 PUT (MP,55),M,PSET: RETURN
690 FOR ZZ=1 TO 2: RESTORE 650: FOR Z=1 T
700 O 19: READ F,D: SOUND F,D: SOUND 30000
710 ,1: AS=INKEYS: IF AS>" THEN RETURN
720 NEXT: NEXT: RETURN
730 AS=INKEYS: IF AS=" THEN 640 ELSE RE
740 TURN
750 DATA 294,3,294,3,294,3,392,6,587,6,
760 523,3,494,3,440,3,784,6,587,6,523,3,
770 494,3,440,3,784,6,587,6,523,3,494,
780 3,523,3,440,12

```

HCM

## LASERITHMETIC

TI-99/4A

```

100 REM *****
110 REM ***** LASERITHMETIC *****
120 REM *****
130 REM ***** COPYRIGHT 1985 *****
140 REM ***** EMERALD VALLEY PUBLISHING CO. *****
150 REM ***** BY R.G. CHRISTENSEN *****
160 REM ***** HOME COMPUTER MAGAZINE *****
170 REM ***** VERSION 5.2.1 *****
180 REM ***** TI BASIC OR *****
190 REM ***** TI EXTENDED BASIC *****
200 REM *****
210 REM *****
220 CALL CLEAR
230 CALL SCREEN(5): "55AA55AA55AA55AA"
240 CALL CHAR(112,16,14)
250 CALL COLOR(11,16,14)
260 CALL HCHAR(13,7,112,20)
270 CALL VCHAR(8,7,112,5)
280 CALL VCHAR(8,26,112,5)
290 CALL VCHAR(8,26,112,5)
300 MS="LASERITHMETIC"
310 R=10
320 C=8
330 GOSUB 1680
340 GOSUB 360
350 GOTO 500
360 MS="PRESS ANY KEY TO CONTINUE"
370 R=22
380 C=4
390 GOSUB 1680
400 FOR I=1 TO 2
410 RESTORE 2180
420 FOR Z=1 TO 19
430 READ LL,F1,V1,F2,V2,F3,V3
440 CALL SOUND(-LL*150,F1,V1,F2,V2,F3,V
450 3)
460 CALL KEY(0,K,S)
470 IF S<>0 THEN 490
480 NEXT I
490 RETURN
500 FOR Z=1 TO 14
510 CALL COLOR(Z,16,1)
520 NEXT Z
530 CALL COLOR(9,7,1)
540 CALL COLOR(10,11,1)
550 CALL COLOR(11,14,1)
560 CALL COLOR(12,5,1)
570 CALL CLEAR
580 PRINT "ENTER PROBLEM TYPE: " "1) AD
590 DITION "2) SUBTRACTION "3) MULT
600 IPLICATION "4) DIVISION"
610 CALL KEY(0,K,S)
620 IF S<>0 THEN 590
630 TYPE=K-48
640 CALL CLEAR
650 PRINT "ENTER SKILL LEVEL: " "1) EA
660 SY "2) HARD "3) HARDER "4) RE
670 AL HARD
680 CALL KEY(0,K,S)
690 IF S<>0 THEN 650
700 IF (K<49)+(K>52) THEN 650
710 SKILL=K-48
720 CALL CLEAR
730 GOSUB 1930
740 CALL SCREEN(3)
750 PRINT "OK: CAPTAIN " "YOU'RE TH
760 E COMMANDER OF THE SPACE STATION CO
770 LUMBIA.
780 PRINT "DESTROY THE ATTACKING ALIENS
790 BY LOADING YOUR LASER CANNON WITH TH
800 E CORRECT ANSWERS TO THE:
810 PRINT "PROBLEMS. " "CAUTION! DON'T
820 LET AN ALIEN GET TOO CLOSE! " "PR
830 ESS [FCTN] 9 TO END PLAY.
840 PRINT "PRESS ANY KEY TO START "
850 CALL KEY(0,K,S)
860 IF S<>0 THEN 760
870 SCORE=0

```

Continued



[illegible]

## HCM

## APPLE // Family

```

580 PR$(8) = "PRINT TEXT HEADERS (Y/N)?
590 PR$(9) = "FORMAT TEXT (Y/N)? "
600 PR$(10) = "PRINTOUT WIDTH (20 TO 13
610 DIM FB$(2):FB$(1) = " : FOR IT = 1
620 TO 132:FB$(1) FB$(1) : + : NEXT
630 DIM PUS$(3):PUS$(0) = " : PUS$(1) = ?
640 DIM ET$(38):DIM DX(2) : A : Z$ = "Z":B
BL$ = CHR$(7) : A$ = CHR$(69) + CHR(27)
D$(27) + CHR$(27) : + CHR$(71) : BOS$ = CHR$(27) + CHR(27)
(72) + CHR$(70) + CHR$(27) + CHR(27)
650 LF$ = CHR$(8):RT$ = CHR$(21):RB
$ = CHR$(13):ESC$ = CHR$(27)
660 QZ$ = "0000"
670 DIM KB(2):KB(1) = - 16384:KB(2) =
- 16388
680 DIM FP%(6)
690 DS% = 50
700 DIM GS%(7)
710 PL% = 60: DIM PG$(PL%)
720 DIM BF$(2)
730 DIM AR%(7)
740 FOR IT = 1 TO 19: READ XS: IF IT =
6 OR IT = 7 THEN 760
750 LES(IT) = CHR$(ASC(XS) - 64): G
OTO 770
760 LES(IT) = XS
770 NEXT
780 GOTO 800
790 DATA "I","D","M","F","E","V","Z","S","B","L","J","G","K","C","<","P",">","H"
800 RETURN
810 REM
820 VTAB 1: HTAB 1: INVERSE : PRINT " T
EE ORGANIZER : NORMAL
830 VTAB 1: HTAB 18: PRINT "REPORTS"
840 VTAB 24: HTAB 5
850 PRINT "PRESS <ESC> TO LEAVE";
860 RETURN
870 REM REPORT MENU
880 GOSUB 3020
890 VTAB 3: HTAB 1: PRINT "FILE: ";: IN
VERSE : PRINT FL$: NORMAL : SL = 1: S
R = 4
900 ON SL GOSUB 940,960,990,990,990
,1090,1090,1200,1210
910 IF INS = ESC$ THEN RETURN
920 IF SL > 10 THEN RETURN
930 GOTO 900

```

**Continued**



```

940  VTAB 1: HTAB 1: PRINT PR$(SL); RA$(SL);
    RATIO = 9999: SR = 1: RETURN
    IF RA$(1) = "N" THEN GOTO 1220: GOSUB 124
950  GOSUB 1220: SR = SR + 1: RETURN
960  IF RA$(3) = "N" THEN GOTO 1240: IF RA$(4) = "N" THEN S
970  $ (SL) SR = 1: RETURN
    SR = 1: RETURN
980  SR = 1: RETURN
990  SR = 1: RETURN
1000  L = 7: RETURN
    IF IN$ = 5: GOSUB 1220: RETURN
1010  IF IN$ = 5: GOSUB 1220: RETURN
1020  IF IN$ = 5: GOSUB 1220: RETURN
1030  O = 1050: SR = SR + 1: GOSUB 1220: GOS
1040  SL = 6: SR = SR + 1: GOSUB 1220: GOS
1050  IF RA$(6) = "Y" THEN TS$ = BD$ + MO
    TS$ = TS$ + MO: RETURN
1060  SR = 1: RETURN
1070  SR = 1: RETURN
1080  GOSUB 1220: RETURN
1090  C = 7: RETURN
    IF IN$ = 7: RETURN
1100  IF IN$ = 7: RETURN
    IF IN$ = 7: RETURN
1110  IF RA$(4) = "Y" OR RA$(7) = "N" THEN
    SR = SR + 1: RETURN
1120  SR = 1: RETURN
    IF IN$ = 8: GOSUB 1220: GOS
    IF IN$ = 8: GOSUB 1220: GOS
1130  IF RA$(SL) = "N" THEN SR = SR + 1: S
    L = 9: GOTO 1200
1140  SL = 5: SR = SR + 1: GOSUB 1220: GOS
    IF IN$ = 5: GOSUB 1220: GOS
1150  IF RA$(5) = "Y" THEN MO$ = "-": GOT
    O = 1170
1160  SL = 6: SR = SR + 1: GOSUB 1220: GOS
    IF IN$ = 6: SR = SR + 1: IF IN$
1170  IF ESC$ THEN RETURN
    IF RA$(6) = "Y" THEN TS$ = BD$ + MO
    TS$ = TS$ + MO: RETURN
1180  SR = 1: RETURN
1190  GOSUB 1220: GOSUB 1380: SR = SR + 1:
    GOSUB 1220: GOSUB 1240: SL = SL + 1:
1200  GOSUB 1220: GOSUB 1240: SL = SL + 1:
    GOSUB 1220: GOSUB 1240: SL = SL + 1:
1210  VTAB 1: HTAB 1: PRINT PR$(SL); RA$(SL);
    REM = 1: RETURN
1220  REM = 1: RETURN
1230  REM = 1: RETURN
1240  HX = 1: RA$(SL) = LEFT$( (RA$(SL) +
    ), 4): FOR IT = 1 TO 4: ET$(IT) =
    MID$(RA$(SL), IT, 1): NEXT VC
1250  HB = 1: LEN (PR$(SL))
    IF HX = 1 THEN HX = 1
1260  IF HX = 1 THEN HX = 1
1270  IF HX = 1 THEN HX = 1
1280  IF HX = 1 THEN HX = 1
1290  IF HX = 1 THEN HX = 1
1300  IF HX = 1 THEN HX = 1
1310  IF HX = 1 THEN HX = 1
1320  IF HX = 1 THEN HX = 1
1330  IF HX = 1 THEN HX = 1
1340  IF HX = 1 THEN HX = 1
1350  IF HX = 1 THEN HX = 1
1360  IF HX = 1 THEN HX = 1
1370  IF HX = 1 THEN HX = 1
1380  IF HX = 1 THEN HX = 1
1390  IF HX = 1 THEN HX = 1
1400  IF HX = 1 THEN HX = 1
1410  IF HX = 1 THEN HX = 1
1420  IF HX = 1 THEN HX = 1
1430  IF HX = 1 THEN HX = 1
1440  IF HX = 1 THEN HX = 1
1450  IF HX = 1 THEN HX = 1
1460  IF HX = 1 THEN HX = 1
1470  IF HX = 1 THEN HX = 1
1480  IF HX = 1 THEN HX = 1
1490  IF HX = 1 THEN HX = 1
1500  IF HX = 1 THEN HX = 1
1510  IF HX = 1 THEN HX = 1
1520  IF HX = 1 THEN HX = 1
1530  IF HX = 1 THEN HX = 1
1540  IF HX = 1 THEN HX = 1
1550  IF HX = 1 THEN HX = 1
1560  IF HX = 1 THEN HX = 1
1570  IF HX = 1 THEN HX = 1
1580  IF HX = 1 THEN HX = 1
1590  IF HX = 1 THEN HX = 1
1600  IF HX = 1 THEN HX = 1
1610  IF HX = 1 THEN HX = 1
1620  IF HX = 1 THEN HX = 1
1630  IF HX = 1 THEN HX = 1
1640  IF HX = 1 THEN HX = 1
1650  IF HX = 1 THEN HX = 1
1660  IF HX = 1 THEN HX = 1
1670  IF HX = 1 THEN HX = 1
1680  IF HX = 1 THEN HX = 1
1690  IF HX = 1 THEN HX = 1
1700  IF HX = 1 THEN HX = 1
1710  IF HX = 1 THEN HX = 1
1720  IF HX = 1 THEN HX = 1
1730  IF HX = 1 THEN HX = 1
1740  IF HX = 1 THEN HX = 1
1750  IF HX = 1 THEN HX = 1
1760  IF HX = 1 THEN HX = 1
1770  IF HX = 1 THEN HX = 1
1780  IF HX = 1 THEN HX = 1
1790  IF HX = 1 THEN HX = 1
1800  IF HX = 1 THEN HX = 1
1810  IF HX = 1 THEN HX = 1
1820  IF HX = 1 THEN HX = 1
1830  IF HX = 1 THEN HX = 1
1840  IF HX = 1 THEN HX = 1
1850  IF HX = 1 THEN HX = 1
1860  IF HX = 1 THEN HX = 1
1870  IF HX = 1 THEN HX = 1
1880  IF HX = 1 THEN HX = 1
1890  IF HX = 1 THEN HX = 1
1900  IF HX = 1 THEN HX = 1
1910  IF HX = 1 THEN HX = 1
1920  IF HX = 1 THEN HX = 1
1930  IF HX = 1 THEN HX = 1
1940  IF HX = 1 THEN HX = 1
1950  IF HX = 1 THEN HX = 1
1960  IF HX = 1 THEN HX = 1
1970  IF HX = 1 THEN HX = 1
1980  IF HX = 1 THEN HX = 1
1990  IF HX = 1 THEN HX = 1
2000  IF HX = 1 THEN HX = 1
2010  IF HX = 1 THEN HX = 1
2020  IF HX = 1 THEN HX = 1
2030  IF HX = 1 THEN HX = 1
2040  IF HX = 1 THEN HX = 1
2050  IF HX = 1 THEN HX = 1
2060  IF HX = 1 THEN HX = 1
2070  IF HX = 1 THEN HX = 1
2080  IF HX = 1 THEN HX = 1
2090  IF HX = 1 THEN HX = 1
2100  IF HX = 1 THEN HX = 1
2110  IF HX = 1 THEN HX = 1
2120  IF HX = 1 THEN HX = 1
2130  IF HX = 1 THEN HX = 1
2140  IF HX = 1 THEN HX = 1
2150  IF HX = 1 THEN HX = 1
2160  IF HX = 1 THEN HX = 1
2170  IF HX = 1 THEN HX = 1
2180  IF HX = 1 THEN HX = 1
2190  IF HX = 1 THEN HX = 1
2200  IF HX = 1 THEN HX = 1
2210  IF HX = 1 THEN HX = 1
2220  IF HX = 1 THEN HX = 1
2230  IF HX = 1 THEN HX = 1
2240  IF HX = 1 THEN HX = 1
2250  IF HX = 1 THEN HX = 1
2260  IF HX = 1 THEN HX = 1
2270  IF HX = 1 THEN HX = 1
2280  IF HX = 1 THEN HX = 1
2290  IF HX = 1 THEN HX = 1

```

Continued



```

23300 RETURN
23310 IF RA$(2) = "RETURN" THEN FB$(2) = PS$:P
23320 SS = VAL(RA$(10)) - 20
23330 X = VAL(RA$(10)) - VAL(RA$(3)) > X TH
23340 EN 2360
23350 X = (LV% + 1) * VAL(RA$(3))
23360 FB$(2) = LEFT$(FB$(1), X) + PS$:PS
23370 IF TF = 1 THEN FB$(2) = FB$(2) + TF
23380 $: RETURN
23390 REM TEXT TRANSFER
23400 IF BF$(DY) = " " THEN RETURN
23410 LC% = LC% + 1
23420 PG$(LC%) = FB$(2) + BF$(DY)
23430 IF LEN(PG$(LC%)) > VAL(RA$(10))
23440 THEN 2460
23450 BF$(DY) = " "
23460 GOTO 2540
23470 FOR IT = VAL(RA$(10)) TO VAL(RA$(10))
23480 $ (10) = LEN(BF$(DY)) STEP 1
23490 IF MIDS(PG$(LC%), IT, 1) = " " THEN
23500 X = IT: GOTO 2500
23510 NEXT X
23520 VAL = (RA$(10))
23530 BF$(DY) = RIGHT$(PG$(LC%), (LEN(PG$(LC%)) - X))
23540 PG$(LC%) = LEFT$(PG$(LC%), X)
23550 IF BF$(DY) = " " THEN BF$(DY) = " "
23560 IF LEFT$(BF$(DY), 1) = " " THEN BF$(DY) = " "
23570 IF RIGHT$(BF$(DY), 1) = " " THEN BF$(DY) = " "
23580 IF LC% = 1 THEN GOTO 2520
23590 IF LC% = PL% THEN GOSUB 2030
23600 RETURN
23610 REM FILE NAME ENTRY
23620 VTAB 10: HTAB 1: VC = 12
23630 PRINT ENTER FILE NAME: "
23640 GOSUB 3450
23650 IF K$ = ESC$ THEN IN$ = K$: RETURN
23660 IF FL$ = " " THEN 2590
23670 DR$ = "1": VTAB 10: HTAB 20
23680 PRINT "IN DRIVE NUMBER: "DR$
23690 VTAB 10: HTAB 37: PRINT DR$: VTAB 1
23700 HTAB 37
23710 GET IN$: IF IN$ = " " THEN 2650
23720 IF IN$ = CHR$(13) OR IN$ = ESC$ THEN
23730 RETURN
23740 IF IN$ = "1" OR IN$ = "2" THEN DR$
23750 IN$: GOTO 2640
23760 PRINT BL$: GOTO 2640
23770 RETURN
23780 REM OPEN A FILE
23790 VTAB 23: PRINT CHR$(4): VTAB 23
23800 IF PD THEN GOSUB 3650
23810 PRINT CHR$(4): "OPEN"; FL$; ",L"; ST
23820 RS(DS%)",D"DR$
23830 RETURN
23840 REM CLOSE A FILE
23850 VTAB 23: PRINT CHR$(4): VTAB 23
23860 PRINT CHR$(4): "CLOSE"
23870 RETURN
23880 REM FILE READ
23890 VTAB 23: PRINT CHR$(4): VTAB 23
23900 PRINT CHR$(4); "READ"; FL$; ",R"; S
23910 TR$(DX(1))
23920 TX$ = (GX + 1)
23930 FOR GX = 1 TO FD: GET IN$: TX$ = TX$
23940 + IN$: NEXT
23950 REM DECODE POINTERS
23960 FOR ZX = 1 TO 9 STEP 2
23970 GS%((ZX + 1) / 2) = 100 * (ASC(M
23980 IDS(EX$, ZX, 1) - 28) + ASC(MIDS
23990 (EX$, (ZX + 1), 1)) - 28
24000 NEXT
24010 REM GET FILE PARAMETERS
24020 DX(1) = 0: FD = 24
24030 GOSUB 2800
24040 FOR GX = 0 TO 5 VAL(MIDS(TX$, 4 *
24050 GX + 1, 4)): NEXT
24060 REM GET AN OCCUPIED NODE
24070 DX(1) = GS%(7): FD = 49: GOSUB 2800
24080 EX$ = LEFT$(TX$, 10): GS%(6) = VAL
24090 (MIDS(TX$, 11, 1))
24100 GOSUB 2860: GS% = RIGHT$(TX$, 38)
24110 RETURN
24120 REM CLEAR PAGE 1
24130 VC = 3: HC = 20
24140 GOSUB 3050: RETURN

```

```

30400 REM BLANK A PORTION OF PAGE
30500 FOR GX = VC TO VC + HC - 1
30600 VTAB GX: HTAB 1
30700 CALL 868: NEXT GX
30800 RETURN
30900 REM GET CHARACTER
31000 VTAB VC: HTAB HC
31100 IN$ = GET
31200 IF IN$ = " " THEN 3120
31300 IF ASC(IN$) > 31 THEN VTAB VC: H
31400 TAB HC: PRINT IN$
31500 RETURN
31600 REM RETURN TO ORGANIZER
31700 ZQ = 1: HOME = VTAB 7: HTAB 1
31800 HOME = VTAB 7: HTAB 1
31900 PRINT "INSERT THE PROGRAM DISK INTO
32000 DRIVE 1"
32100 PRINT "THEN PRESS ANY KEY: "
32200 VC = 8: HC = 21: GOSUB 3100
32300 HOME = PRINT
32400 PRINT CHR$(4) "RUN ORGANIZER, D1"
32500 RETURN
32600 REM ERROR HANDLER
32700 HOME = PRINT CHR$(4) "CLOSE"
32800 HOME = VTAB 7: HTAB 1
32900 X = PEEK(222)
33000 ON ((X = 6 OR X = 5) AND ZQ = 0) +
33100 2 * ((X = 13) + 3 * ((X = 6 OR X = 5
33200 3370, 3410, 3420, 3430, 3440) GOTO 5
33300 PRINT "ERROR NUMBER "X": "
33400 PRINT "AT LINE "PEEK(218) + PEE
33500 K(219) * 256)
33600 VTAB 12: HTAB 1
33700 PRINT "PRESS ANY KEY TO CONTINUE: "
33800 GET IN$: IF IN$ = " " THEN 3330
33900 RUN
34000 RETURN
34100 REM DETAIL ERROR EXPLANATIONS
34200 PRINT "FILE "; FL$: " ": PRINT "
34300 PRINT "NOT FOUND ON DISKETTE."
34400 PRINT " ": PRINT CHR$(4) "DELETE "
34500 FL$
34600 GOTO 3310
34700 PRINT "FILE "; FL$: " ": PRINT " ": I
34800 S NOT AGOODFILE: GOTO 60180
34900 PRINT "ORGANIZER" PROGRAM NOT ON DI
35000 SK: GOTO 3310
35100 PRINT "CONTROL 'C' KEY WAS PRESSED"
35200 TO PRINT "STOPPING THE PROGRAM." : GO
35300 TO 3310
35400 REM GET FILENAME ROUTINE
35500 CR$ = CHR$(13)
35600 TS = " ": VT = VC: HT = 1
35700 IF PD THEN MX = 15: GOTO 3490
35800 MX = 30
35900 VTAB VT: HTAB HT: PRINT TS: SPC(MX
36000 - LEN(TS)): HTAB LEN(TS) + HT
36100 IF LEN(TS) = MX THEN HTAB MX
36200 IF TS = " " THEN GOSUB 3590: GOTO 3
36300 GOSUB 3610
36400 IF K$ = ESC$ THEN 3640
36500 IF K$ = CR$ THEN 3640
36600 IF K$ = LF$ AND LEN(TS) < 2 THEN
36700 TS = " ": GOTO 3490
36800 IF K$ = LF$ THEN TS = LEFT$(TS, L
36900 EN(TS) - 1): GOTO 3490
37000 IF LEN(TS) < MX THEN TS = TS + K$
37100 : GOTO 3490
37200 TS = LEFT$(TS, MX - 1) + K$: GOTO
37300 3490
37400 GET K$: IF NOT (K$ = LF$ OR K$ = E
37500 SC$ OR K$ = CR$ OR (K$ > "A" AND
37600 K$ < "Z")) THEN PRINT CHR$(7)
37700 : GOTO 3590
37800 RETURN
37900 IF PD THEN GET K$: IF NOT (K$ = L
38000 F$ OR K$ = CR$ OR K$ = "Z") OR (K$ >
38100 "A" AND K$ < "Z") OR K$ = " "
38200 ) THEN PRINT CHR$(7): GOTO 3610
38300 IF PD = 0 THEN GET K$: IF NOT (K$
38400 = LF$ OR K$ = CR$ OR K$ = ESC$ OR
38500 (K$ > "A" AND K$ < "Z") OR (K$ >
38600 " " AND K$ < " ")) THEN
38700 PRINT CHR$(7): GOTO 3620
38800 RETURN
38900 FL$ = TS: RETURN
39000 REM GET PREFIX
39100 PRINT CHR$(4): "PREFIX,D"; DR$: RET
39200 URN

```

PROGRAM LISTING

HCM

## THE ORGANIZER REPORTS

COMMODORE 64

```

100 REM *****
110 REM * THE ORGANIZER *
120 REM * REPORTS *
130 REM *****
140 REM COPYRIGHT 1985
150 REM EMERALD VALLEY PUBLISHING CO.
160 REM BY WILLIAM K. BALTHROP
170 REM AND THE HCM STAFF
180 REM HOME COMPUTER MAGAZINE
190 REM VERSION 5.2.1
200 REM C-64 BASIC

```

```

210 REM SAVE THIS PROGRAM WITH
220 REM THE FILE NAME "REPORTS"
230 REM REPLACING "DUMMY REPORTS"
240 REM FROM VOL. 5, NO. 1
250 REM "THE ORGANIZER" PROGRAM
260 GOSUB 1830: GOSUB 1780: GOSUB 1890
270 GOSUB 1050
280 IF OP(4) OR OP(7) = 4 THEN 310
290 Y=Y+2: X=X+1: GOSUB 1720: PRINT "... YOU M
300 UST PRINT OUTLINE OR TEXT"

```

Continued



```

300 FOR I=1 TO 5000: NEXT: GOTO 270
310 Y=Y+1: IF Y=2: X=1: TO 5000: NEXT: GOTO 270
320 GOSUB 1720: PRINT "IS THIS OK (Y/N)":
330 L=1: B=32: T=99: IF S$<>"Y" AND S$<>"N" AN
340 GOSUB 1720: PRINT "SHIFT CRSRLEFT"
350 IF S$="N" THEN 270
360 IF S$="Y" THEN PRINT "SHIFT CRSRLEFT"
370 GOSUB 1720: GOSUB 1740: X=1: Y=6
380 GOSUB 1720: PRINT "PRINT ANOTHER ONE (
390 Y/N)": IF S$="N" THEN 270: IF S$="Y"
400 B=32: T=99: L=1: GOSUB 1590: IF S$<>"Y"
410 AND S$<>"N" THEN 270
420 IF S$="Y" THEN PRINT "SHIFT CRSRLEFT"
430 GOTO 2170
440 IF S$="N" THEN PRINT "SHIFT CRSRLEFT"
450 REM PRINT ROUTINE
460 TX=OP(7): LV=0: LN=1: RP=HF%(8): MHS=LE
470 FTS(" ", OP(5)): BHS=LEFTS(EDS, OP(6))
480 IF BHS=" " THEN BHS=NO$
490 Y=20: X=15: GOSUB 1720: PRINT "CTRL RVS
500 ON THINKING"
510 IF OP(4) THEN RN=RP: GOSUB 560
520 IF (NOT OP(4)) AND OP(8) AND LNK%(R
530 P, TX)>0 THEN RN=RP: GOSUB 560
540 IF LNK%(RP, TX)>0 THEN TP=LNK%(RP, TX)
550 : GOSUB 720
560 IF LNK%(RP, 1)>0 AND (LV<OP(1)) THEN
570 RP=LNK%(RP, 1): LV=LV+1: GOTO 480
580 IF LNK%(RP, 3)>0 THEN RP=LNK%(RP, 3):
590 GOTO 480
600 IF LNK%(RP, 0)>0 THEN RP=LNK%(RP, 0):
610 LV=LV-1: GOTO 520
620 IF LN<>1 THEN GOSUB 680
630 RETURN
640 REM SET PAGE LINE
650 GOSUB 2120
660 GOSUB 590: GOSUB 660: RETURN
670 T=OP(2): LV: IF OP(10)-T<20 THEN T=OP
680 (10)-20
690 TZS="": IF (LEN(TX$)+T)>OP(10) THEN G
700 OSUB 2190
710 PS(LN)=PS(LN)+LEFTS(SP$, T)
720 IF TT THEN PS(LN)=PS(LN)+NO$+TX$: GO
730 TO 640
740 PS(LN)=PS(LN)+BHS+MHS+TX$
750 IF TZS>PS(LN) THEN GOSUB 660: TX$=TZS: GOT
760 OSUB 590
770 RETURN
780 LN=LN+1: IF LN=61 THEN GOSUB 680
790 RETURN
800 REM PRINT A PAGE
810 Y=20: X=15: GOSUB 1720: PRINT "CTRL RVS
820 ON THINKING"
830 OPEN 4, 4: FOR I=1 TO 60: PRINT#4, PS(I)
840 : PS(I)="": NEXT: LN=1: PRINT#4, FFS: CLO
850 SE4
860 RETURN
870 REM HANDLE TEXT OUTPUT
880 TT=-1: F=1
890 RN=TP: GOSUB 2120: GOSUB 990
900 IF TX$="P" THEN PS(LN)="": GOTO
910 810
920 IF TX$="B" THEN 800
930 TX$=MIDS(TX$, F): F=1
940 GOSUB 590
950 IF OP(9) THEN GOSUB 840: IF TX$="P"
960 THEN 810
970 GOSUB 660
980 IF LNK%(TP, 3)>0 THEN TP=LNK%(TP, 3):
990 GOTO 740
1000 TT=0
1010 RETURN
1020 REM FORMAT TEXT OUTPUT
1030 FP=TP
1040 IF LNK%(FP, 3)=0 THEN RETURN
1050 FP=LNK%(FP, 3)
1060 RN=FP: GOSUB 2120: GOSUB 990
1070 IF TX$="P" OR TX$="B" THEN F=1:
1080 RETURN
1090 I=1
1100 IF MIDS(TX$, I, 1)=" " THEN 930
1110 I=1+1: IF I<LEN(TX$) THEN 910
1120 MX=OP(10)-(OP(2)*LV)-LEN(PS(LN))-I+
1130 2
1140 IF MX<=0 THEN RETURN
1150 PS(LN)=PS(LN)+LEFTS(TX$, I): TX$=MIDS
1160 (TX$, I+1)
1170 F=F+1: IF I=1 OR I=LEN(TX$) THEN F=1
1180 IF LEN(TX$)>0 THEN 900
1190 TP=FP: F=1: GOTO 860
1200 REM STRIP OFF TRAILING SPACES
1210 I=LEN(TX$)
1220 IF MIDS(TX$, I, 1)<>" " THEN 1030
1230 I=I-1: IF I>0 THEN 1010
1240 TX$=LEFTS(TX$, I): REM EXCEPT ONE
1250 RETURN
1260 REM INPUT OPTIONS
1270 GOSUB 1740: X=1: Y=6: B=32: T=99: L=1
1280 GOSUB 1720: PRINT "SELECT MAX GENERATI
1290 ON (Y/N)": IF S$="Y" AND S$<>"N" AN
1300 D S$<>"N" THEN 1070
1310 IF S$="N" THEN V=HF%(6): GOTO 1130
1320 Y=Y+1: B=48: T=57: L=3: IF S$=" " THEN P
1330 RINT "SHIFT CRSRLEFT"
1340 GOSUB 1720: PRINT "MAXIMUM REPORT GE
1350 NERATION?": IF S$="Y" AND S$<>"N" AN
1360 RINT "SHIFT CRSRLEFT"
1370 OP(1)=V: Y=Y+1: L=1: B=32: T=99
1380 GOSUB 1720: PRINT "INDENT GENERATIONS
1390 (Y/N)": IF S$="Y" AND S$<>"N" AN
1400 D S$<>"N" THEN 1140
1410 IF S$="N" THEN 4=0: GOTO 1200
1420 Y=Y+1: B=49: T=53: L=1: IF S$=" " THEN P
1430 RINT "SHIFT CRSRLEFT"
1440 GOSUB 1720: PRINT "INDENT WIDTH (1 T
1450 O 5)": IF S$="Y" AND S$<>"N" AN
1460 D S$<>"N" THEN 1190
1470 GOSUB 1590: V=VAL(S$): IF S$=" " THEN P
1480 RINT "SHIFT CRSRLEFT"
1490 OP(2)=V: Y=Y+1: B=32: T=99
1500 GOSUB 1720: PRINT "PRINT OUTLINE (Y/N)
1510 : GOSUB 1590: IF S$="Y" AND S$<>"N" AN
1520 D S$<>"N" THEN 1210
1530 IF S$="N" THEN OP(4)=0: GOTO 1260
1540 OP(4)=1: Y=Y+1: IF S$=" " THEN PRINT "
1550 SHIFT CRSRLEFT"
1560 GOSUB 1490
1570 Y=Y+1
1580 GOSUB 1720: PRINT "PRINT TEXT (Y/N)":
1590 D S$<>"N" THEN 1270
1600 IF S$="Y" AND S$<>"N" AN
1610 D S$<>"N" THEN 1270
1620 IF S$="N" THEN OP(7)=5: GOTO 1380
1630 IF S$=" " THEN PRINT "SHIFT CRSRLEFT"
1640 Y=Y+1: IF OP(4) THEN 1380
1650 Y=Y+1
1660 GOSUB 1720: PRINT "PRINT TEXT HEADER
1670 S (Y/N)": IF S$="Y" AND S$<>"N" AN
1680 D S$<>"N" THEN 1330
1690 IF S$="N" THEN OP(8)=0: GOTO 1380
1700 OP(8)=1: IF S$=" " THEN PRINT "SHIFT
1710 CRSRLEFT"
1720 Y=Y+1: GOSUB 1490
1730 Y=Y+1
1740 GOSUB 1720: PRINT "FORMAT TEXT (Y/N)":
1750 D S$<>"N" THEN 1400
1760 IF S$="Y" AND S$<>"N" AN
1770 D S$<>"N" THEN 1390
1780 IF S$="N" THEN OP(9)=0: GOTO 1430
1790 OP(9)=1: IF S$=" " THEN PRINT "SHIFT
1800 CRSRLEFT"
1810 Y=Y+1: L=3: B=48: T=57
1820 GOSUB 1720: PRINT "PRINT OUT WIDTH (20
1830 TO 132)": IF S$="Y" AND S$<>"N" AN
1840 D S$<>"N" THEN 1450
1850 GOSUB 1590: V=VAL(S$): IF (V<20 OR V>1
1860 32) AND S$<>" " THEN 1440
1870 IF S$=" " THEN PRINT "SHIFT CRSRLEFT"
1880 : 4=0: GOTO 1400
1890 OP(10)=V
1900 RETURN
1910 GOSUB 1720: PRINT "MARK HEADERS (Y/N
1920 )": IF S$="Y" AND S$<>"N" AN
1930 D S$<>"N" THEN 1500
1940 IF S$="N" THEN OP(5)=0: GOTO 1530
1950 OP(5)=1: IF S$=" " THEN PRINT "SHIFT
1960 CRSRLEFT"
1970 Y=Y+1
1980 GOSUB 1720: PRINT "BOLD HEADERS (Y/N
1990 )": IF S$="Y" AND S$<>"N" AN
2000 D S$<>"N" THEN 1550
2010 IF S$="N" THEN OP(6)=0: GOTO 1580
2020 OP(6)=1: IF S$=" " THEN PRINT "SHIFT
2030 CRSRLEFT"
2040 RETURN
2050 REM INPUT ROUTINE
2060 POKE 198, 0: S$=" "
2070 PRINT "CMDR @": 2SHIFT CRSRLEFT "":
2080 GOSUB 1680
2090 IF K=13 THEN PRINT " ": GOTO 1710
2100 IF K=20 AND LEN(S$)>0 THEN S$=LEFTS
2110 (S$, LEN(S$)-1): PRINT "SHIFT CRSRLEF
2120 T": GOTO 1610
2130 IF K<B OR K>T OR K=34 OR LEN(S$)=L
2140 THEN 1610
2150 IF K$="S" OR K$=" " OR K$="," OR K$
2160 " " OR K$="?" OR K$="@": THEN 1610
2170 PRINT K$: S$=S$+K$: GOTO 1610
2180 RETURN
2190 GET K$: IF K$=" " THEN 1680
2200 K=ASC(K$)
2210 IF K$=ESS THEN 2170
2220 RETURN
2230 REM PLACE CURSOR AT X,Y
2240 POKE 781, Y: POKE 782, X: POKE 783, 0: SY
2250 S 65520: RETURN
2260 REM INITIALIZE THE SCREEN
2270 POKE 53280, 12: POKE 53281, 12: POKE 64
2280 6, 1: PRINT "SHIFT CLR TAB(13) THE O
2290 RANIZER"
2300 PRINT TAB(16): "CRSRDOWN REPORTS": X=
2310 4: Y=24: GOSUB 1720
2320 PRINT "CTRL RVSON PRESS CTRL RVSONTO
2330 SPACE" ESS "SPACE CTRL RVSONTO
2340 RETURN TO MAIN MENU CTRL RVSONTO
2350 RETURN

```

Continued



# THE ORGANIZER REPORTS *Continued*

COMMODORE 64

```

1780 REM INPUT FILE NAME
1790 GOSUB 1740: LINE 12: B=32: T=99: X=1: Y=6
1800 GOSUB 1720: PRINT "ORGANIZER" FILE NAME
1810 FS=SS
1820 RETURN
1830 REM INITIALIZE VARIABLES
1840 ED$=CHR$(27)+"E": NOS=CHR$(27)+"F": R
EM CHANGE THESE TO MATCH YOUR PRINT
ER
1850 PL$=CHR$(112): CR$=CHR$(13): ES$="-":
PT=1: DIM HF%(8), OP$(10), PS(60)
1860 FC$=CHR$(2): TP$=CHR$(1): SP$=""
1870 FOR I=1 TO 5: FF$=FF$+CR$: NEXT
1880 REM FILE ROUTINES
1890 PRINT: PRINT "CRSR DOWN" ***GETTING F
ILE: FS=PRINT: ***PLEASE WAIT
1910 OPEN 15,8,15,10: OPEN 3,8,3,15: OR
D,S,R: GOSUB 2040: IF EN<>64 THEN 1930
1920 CLOSE 3: OPEN 2,8,2,15: ORD: GOSUB 20
40
1930 IF EN>19 THEN CLOSE 2: CLOSE 3: GOTO 206
0
1940 RC=1: GOSUB 1990: PT$=TP$: GOSUB 2000: IN
PUT #2, HF$: GOSUB 2030
1950 DIM LNK$(HF%(0),5)
1960 PT$=PL$: FOR I=1 TO 2 TO HF%(0): RC=I: GOSU
B 1990: GOSUB 2000
1970 INPUT #2, LNK$: GOSUB 2010: NEXT I
1980 RETURN
1990 HB%=INT(RC/256): LB%=RC-(256*HB%): RC
$=CHR$(LB%)+CHR$(HB%): RETURN

```

```

2000 PRINT #15: "P"+FC$+RC$+PT$: RETURN
2010 X=2: Y=12: GOSUB 1720: PRINT "READING R
ECORD #": I=1: FOR J=0 TO 4
2020 LNK$(I,J)=VAL(MID$(LNK$,J*3+1,3)): N
EXT J: RETURN
2030 FOR J=0 TO 8: HF$(J)=VAL(MID$(HF$,J*
3+1,3)): NEXT J: RETURN
2040 INPUT #15, EN, EMS, ET, ES: RETURN
2050 REM DISK ERROR - RERUN PROGRAM
2060 CLOSE 15: PRINT "SHIFT CLR CRSR DOWN
N: DISK ERROR #": EN: CTRL RVSO
N: EMS
2070 IF CHAIN THEN PRINT "CRSR DOWN"
IF TRYING TO GET MAIN MENU
2080 PRINT "CRSR DOWN" PLEASE CORR
ECT AND TRY AGAIN
2090 PRINT "CRSR DOWN" PRESS [RETU
RN] TO CONTINUE
2100 IF PEEK(197)<>1 THEN 2100
2110 POKE 198,0: RUN
2120 TX$="": RC=RN: GOSUB 1990: I=1
2130 PT$=CHR$(I): GOSUB 2000: GET #2, T$
2140 IF T$=CR$ THEN RETURN
2150 TX$=TX$+T$: I=I+1: IF I<40 THEN 2130
2160 RETURN
2170 REM RETURN TO MAIN MENU
2180 CLOSE 15: PRINT "SHIFT CLR CRSR DOWN"
GETTING MAIN MENU: LOAD "ORGANIZE",8
TZ=OP(10)-T: FOR ZZ=TZ TO 1 STEP -1
2190 IF MID$(TX$,ZZ,1) THEN 2220
2200 NEXT TZ: TZ$=RIGHT$(TX$,LEN(TX$)-TZ): TX
$=LEFT$(TX$,TZ): RETURN
2210 TZ$=RIGHT$(TX$,LEN(TX$)-ZZ): TX$=LEF
T$(TX$,ZZ-1): RETURN
2220

```

HCM

# THE ORGANIZER REPORTS

IBM PC & IBM PCjr

```

100 ***
110 THE ORGANIZER
120 REPORTS
130 ***
140 COPYRIGHT 1985
150 EMERALD VALLEY PUBLISHING CO.
160 BY WILLIAM K. BALTHROP
170 HOME COMPUTER MAGAZINE
180 VERSION 5.2.1
190 IBM PCjr W/CARTRIDGE BASIC or
200 IBM PC W/BASIC
210 SAVE THIS PROGRAM WITH THE
220 FILE NAME "REPORTS"
230 REPLACING "DUMMY REPORTS"
240 FROM VOL. 5, NO. 1
250 "THE ORGANIZER" PROGRAM
260 ON ERROR GOTO 1410
270 CLS: SCREEN 0: DEFINT A-Z: GOSUB 1290:
LOCATE 12,1: PRINT "PLACE DATA DISK
IN DRIVE": PRINT: PRINT "PRESS
CHR$(17): CHR$(217): WHEN READY.
GOSUB 1330
IF AS=ESC$ THEN 1370
GET FILE LINKS
CLS: LOCATE 1,13,0: PRINT "THE ORGANI
ZER": LOCATE 2,16: PRINT "REPORTS": LO
CATE 4,1: PRINT "ORGANIZER FILE NAME
LOCATE 12,1: PRINT "TO EXIT TO MA
IN MENU: PRINT "PRESS": CHR$(17): CH
RS(217): WITH NO FILE NAME
330 LOCATE 4,21,1: INPUT FS: IF FS=""
THEN 1370 ELSE IF MID$(FS,2,1)=""
THEN FS=LEFT$(FS,10) ELSE FS=LEFT$(
FS,8)
P=INSTR(FS,""): IF P>0 THEN FS=LEFT
$(FS,P-1)
350 F1$=FS+LNK: F2$=FS+ORD: LOCATE
6,1,0: PRINT "LOADING": FS: PRINT: PRI
NT: PLEASE WAIT
360 LOCATE 9,1,0: PRINT "READING RECORD
#": OPEN F1$ AS 1 LEN=10: OPEN F2$ A
S 2 LEN=38: GOSUB 1200: IF NUMREC<3 T
HEN ERROR 53 ELSE DIM LNK(NUMREC,4)
GOSUB 1240: FOR REC=2 TO NUMREC: GO
SUB 1210: LOCATE 9,17,0: PRINT REC-1:
NEXT
370
380 MAIN CONTROL ROUTINE
390
400 CLS: LOCATE 1,16: PRINT "REPORTS": LOC
ATE 2,(40-LEN(FS))/2: PRINT FS
410 GOSUB 460: LOCATE 23,1: PRINT "IS THI
S OK (Y/N)?": PR=23: GOSUB 690: IF AS=
"Y" OR AS="" OR AS=" " THEN 400 EL
E GOSUB 790
420 CLS: LOCATE 12,1: PRINT "PRINT ANOTHE
R ONE (Y/N)?": PR=12: GOSUB 690: IF AS=
"Y" OR AS="" OR AS=" " THEN FOR Z=
1 TO 60: PAGE$(Z)="" : NEXT: LNC=1: GOTO
400 ELSE 1370
430
440 GET OPTIONS
450
460 PR=4: LOCATE PR,1: PRINT "SELECT MAX
GENERATION (Y/N)?": PR=4: GOSUB 690: IF AS=
ESC$ THEN RETURN ELSE IF AS=CHR$(1
3) THEN SMAX$="N" ELSE SMAX$=AS

```

```

470 IF SMAX$="N" OR SMAX$=" " THEN OPT$
(1)=STR$(NUMREC): GOTO 490
480 PR=PR+1: PL=1: GOSUB 750: GOSUB 710: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(1)=N$
490 PR=PR+1: PL=2: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(2)=A$
500 IF OPT$(2)="N" OR OPT$(2)=" " THEN
520
510 PR=PR+1: PL=3: GOSUB 750: GOSUB 710: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
AND (VAL(N$)<1 OR VAL(N$)>5) THEN P
R=PR-1: GOTO 510 ELSE IF N$>" " THEN
OPT$(3)=N$
520 PR=PR+1: PL=4: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(4)=A$
530 IF OPT$(4)="N" OR OPT$(4)=" " THEN
610
540 GOSUB 550: IF AS=ESC$ THEN RETURN EL
SE GOSUB 580: IF AS=ESC$ THEN RETURN
ELSE 610
550 PR=PR+1: PL=5: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(5)=A$
560 IF OPT$(5)="Y" OR OPT$(5)=" " THEN
SP.ON$=" " ELSE SP.ON$="Y" THEN
RETURN
570 PR=PR+1: PL=6: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(6)=A$
580 IF OPT$(6)="Y" OR OPT$(6)=" " THEN
SP.ON$=BOLD.ON$+SP.ON$ ELSE SP.ON$=
BOLD.OFF$+SP.ON$
600 RETURN
610 PR=PR+1: PL=7: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(7)=A$
620 IF (OPT$(7)="Y" OR OPT$(7)=" ") AND
(OPT$(4)="N" OR OPT$(4)=" ") THEN
GOSUB 640: IF AS=ESC$ THEN RETURN
630 GOTO 660
640 PR=PR+1: PL=8: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(8)=A$
650 IF OPT$(8)="Y" OR OPT$(8)=" " THEN
GOSUB 550: IF AS=ESC$ THEN RETURN EL
SE GOSUB 580: RETURN ELSE RETURN
660 PR=PR+1: PL=9: GOSUB 750: GOSUB 690: IF
AS=ESC$ THEN RETURN ELSE IF AS>" "
THEN OPT$(9)=A$
670 PR=PR+1: PL=10: GOSUB 750: GOSUB 710: I
F AS=ESC$ THEN RETURN ELSE IF AS>" "
AND (VAL(N$)<20 OR VAL(N$)>132) T
HEN PR=PR-1: GOTO 670 ELSE IF N$>" "
THEN OPT$(10)=N$
680 RETURN
690 LOCATE PR,29,1
700 AS=INKEY$: IF AS="" THEN 690 ELSE IF
AS<>"Y" AND AS<>" " AND AS<>"N" AN
D AS<>" " AND AS<>ESC$ AND AS<>CHR$
(13) THEN 690 ELSE IF AS=ESC$ THEN
RETURN ELSE PRINT AS: RETURN
710 N$="": LOCATE PR,29,0: PRINT OPT$(PL)
: SPACES(4): LOCATE PR,29,1
720 AS=INKEY$: IF AS="" THEN 720 ELSE IF
AS<>"0" OR AS>"9" AND AS<>CHR$(13)
AND AS<>ESC$ THEN 720

```

Continued



```

730 IF AS=ESC$ OR AS=CHR$(13) THEN RETURN
740 IF ELSE IF AS=CHR$(8) THEN RETURN
750 NS=NS+AS:LOCATE PR,29,1:PRINT NS;:1
760 F LEN(NS)=4 THEN RETURN ELSE 720
770 LOCATE PR,1:PRINT OPTTEXT$(PL);TAB(2
780 9);OPT$(PL):RETURN
790 PRINT REPORT
800 FRMT=0:LEV=0:LNC=1:RP=FPFC
810 IF (OPT$(4)=" " OR OPT$(4)="N") AND
820 (OPT$(8)=" " OR OPT$(8)="N") THEN
830 IF (OPT$(4)=" " OR OPT$(4)="N") AND
840 (OPT$(8)=" " OR OPT$(8)="N") AND L
850 INK(RP,4)=0 THEN 830
860 REC=RP:GOSUB 1220:GOSUB 950:IF OPT$(
870 9)="Y" OR OPT$(9)="Y" THEN GOSUB 1
880 110
890 IF LINK(RP,4)>0 AND LEV<=VAL(OPT$(1
900 )) AND (OPT$(7)="Y" OR OPT$(7)="Y")
910 THEN TRP=LINK(RP,4):GOSUB 880:FRMT
920 =0
930 IF LINK(RP,1)>0 AND LEV<=VAL(OPT$(1
940 )) THEN RP=LINK(RP,1):LEV=LEV+1:GOTO
950 800
960 IF LINK(RP,3)>0 THEN RP=LINK(RP,3):
970 GOTO 800
980 IF LINK(RP,0)>0 THEN RP=LINK(RP,0):
990 LEV=LEV+1:GOTO 850
1000 IF LNC=1 AND PAGES(1)=" " THEN RETURN
1010 ELSE GOSUB 1160:FOR Z=1 TO 60:PA
1020 GES(Z)=" ":NEXT:LNC=1:RETURN
1030 FRMT=1
1040 REC=TRP:GOSUB 1220:GOSUB 950
1050 IF LINK(TRP,3)>0 THEN TRP=LINK(TRP,
1060 3):GOTO 890
1070 GOSUB 1110:RETURN
1080 BUILD PAGE OF OUTLINE AND TEXT
1090 GOSUB 1130:IF OPT$(2)="Y" OR OPT$(2
1100 )="Y" THEN TABSP=LEV*VAL(OPT$(3)) E
1110 LSE TABSP=0
1120 IF VAL(OPT$(10))-TABSP<20 THEN TABS
1130 P=VAL(OPT$(10))-20
1140 IF OPT$(9)="Y" OR OPT$(9)="Y" THEN
1150 1020
1160 PAGES(LNC)=HT$:GOSUB 1110:RETURN
1170 FORMAT TEXT FOR OUTPUT
1180 IF FRMT=1 THEN IF HT$="P" OR HT$="
1190 *P" THEN 1090 ELSE IF HT$="B" OR H
1200 T$="B" THEN 1100
1210 MXL=VAL(OPT$(10))-3:IF OPT$(2)="Y"
1220 OR OPT$(2)="Y" THEN MXL=MXL-TABSP
1230 BS=MXL-LEN(PAGES(LNC)):IF BS=0 THEN
1240 GOSUB 1110:GOTO 1040
1250 IF LEN(HT$)<BS THEN PAGES(LNC)=PAGE
1260 S(LNC)+HT$:RETURN
1270 FOR Z=BS TO 1 STEP -1:IF MIDS(HT$,Z
1280 ,1)=" " THEN P=Z-1:IF P=0 THEN 1110
1290 ELSE 1080
1300 NEXT:IF BS<MXL THEN GOSUB 1110:GOTO
1310 1040 ELSE PAGES(LNC)=LEFT$(HT$,MXL
1320 ):HT$=RIGHT$(HT$,LEN(HT$)-MXL):GOSU
1330 B 1110:GOTO 1040
1340 PAGES(LNC)=PAGES(LNC)+LEFT$(HT$,
1350 P):HT$=RIGHT$(HT$,LEN(HT$)-P-1):GO
1360 SUB 1110:GOTO 1040
1370 GOSUB 1110:GOSUB 1110:RETURN
1380 GOSUB 1110:GOSUB 1110:RETURN
1390 IF FRMT=0 THEN PAGES(LNC)=SPACES(TA
1400 BSP)+SP.ON$+PAGES(LNC) ELSE PAGES(L
1410 NC)=SPACES(TABSP)+BOLD.OFF$+PAGES(L
1420 NC)
1430 IF LNC=60 THEN GOSUB 1160:FOR Z=1 T
1440 O 60:PAGES(Z)=" ":NEXT:LNC=1:RETURN
1450 ELSE LNC=LNC+1:RETURN
1460 FOR Z=38 TO 1 STEP -1:IF MIDS(HT$,Z
1470 ,1)>CHR$(32) THEN 1150

```

```

1140 NEXT:HT$=" ":RETURN
1150 HT$=LEFT$(HT$,Z):RETURN
1160 FOR Z=1 TO 60:LPRIOT PAGES(Z):NEXT:
1170 LPRIOT CHR$(12):RETURN
1180 FILE ACCESS ROUTINES
1190
1200 GOSUB 1230:GET 2,1:NUMREC=CVI(NUMS)
1210 :NUSE=CVI(NUSES):FEMP=CVI(FERS):LEM
1220 P=CVI(LEERS):HFIR=CVI(HF$):HLST=CVI(
1230 HL$):THFIR=CVI(THF$):THLST=CVI(THL$)
1240 :FPFC=CVI(FPFC$):RETURN
1250 GET 1,REC:LINK(REC,0)=CVI(PBS):LINK
1260 (REC,1)=CVI(PFS):LINK(REC,2)=CVI(BB
1270 $):LINK(REC,3)=CVI(BFS):LINK(REC,4)
1280 =CVI(TS):RETURN
1290 GOSUB 1250:GET 2,REC:HT$=H$:RETURN
1300 FIELD 2,2 AS NUM$,2 AS NUSE$,2 AS F
1310 ERS$,2 AS LEERS$,2 AS HF$,2 AS HL$,2 A
1320 S THF$,2 AS THL$,2 AS FPFC$:RETURN
1330 FIELD 1,2 AS PBS$,2 AS PFS$,2 AS BBS$,
1340 2 AS BFS$,2 AS TS$:RETURN
1350 FIELD 2,38 AS H$:RETURN
1360 INITIALIZE VARIABLES
1370 ESC$=CHR$(27):BOLD.ON$=ESC$+CHR$(69
1380 )+ESC$+CHR$(71):BOLD.OFF$=ESC$+CHR$
1390 (70)+ESC$+CHR$(72):SP.ON$=" ":SP.OFF
1400 $=" ":
1410 DIM OPT$(10),OPTTEXT$(10),PAGES(60):
1420 RESTORE 1310:FOR Z=1 TO 10:READ OPT
1430 EXT$(Z),OPT$(Z):NEXT:RETURN
1440 DATA MAXIMUM REPORT GENERATION?,0,1
1450 NDENT GENERATIONS (Y/N)?,Y,INDEN
1460 T WIDTH (1 TO 5)?,3,PRINT OUTLINE
1470 (Y/N)?,Y,MARK HEADERS (Y/N)?,Y,
1480 BOLD HEADERS (Y/N)?,Y,PRINT TEX
1490 T (Y/N)?,Y,PRINT TEXT HEADERS (Y
1500 /N)?,Y,FORMAT TEXT (Y/N)?,Y
1510 DATA PRINTOUT WIDTH (20 TO 132)?,80
1520 AS=" ":WHILE AS=" ":AS=INKEY$:WEND:RE
1530 TURN
1540 RETURN TO MENU
1550
1560 CLS:LOCATE 12,1:PRINT "PLACE PROGRA
1570 M DISK IN DRIVE 'A':PRINT "PRESS"
1580 :CHR$(17):CHR$(217):"WHEN READY.":
1590 GOSUB 1330:CLS:LOCATE 12,1:PRINT "L
1600 OADING MAIN MENU...":RUN "A:ORGANIZ
1610 E"
1620 ERROR HANDLING ROUTINE
1630
1640 ERC=ERR:EL=ERL:CLS:RESTORE 1460:FOR
1650 Z=1 TO 20:READ E,M$:IF E=ERC THEN
1660 1430
1670 NEXT:LOCATE 12,1:PRINT "ERROR CONDI
1680 TION IN PROGRAM.":PRINT:PRINT "ER
1690 ROR CODE #";ERC;" IN LINE #";EL:GOT
1700 O 1440
1710 LOCATE 12,1:PRINT M$
1720 LOCATE 20,1:PRINT "PRESS ";CHR$(17)
1730 :CHR$(217):"TO RESTART PROGRAM":PR
1740 INT:PRINT "PRESS 'Esc' TO RETURN TO
1750 MAIN MENU"
1760 GOSUB 1330:IF AS=ESC$ THEN 1370 ELS
1770 E IF AS=CHR$(13) THEN RUN ELSE 1450
1780 DATA 24,DEVICE TIMEOUT,25,DEVICE FA
1790 ULT,27,OUT OF PAPER,52,BAD FILE NUM
1800 BER,53,FILE NOT FOUND,54,BAD FILE M
1810 ODE,55,FILE ALREADY OPEN,57,DEVICE
1820 I/O ERROR,58,FILE ALREADY EXISTS,61
1830 ,DISK IS FULL,62,FILE PROBLEM -- IN
1840 PUT PAST END,64,BAD FILE NAME
1850 DATA 67,TOO MANY FILES -- BAD NAME,
1860 68,DEVICE IS NOT AVAILABLE,69,COMMU
1870 NICATIONS BUFFER OVERFLOW,70,DISK I
1880 S WRITE PROTECTED,71,DISK IS NOT RE
1890 ADY,72,DISK MEDIA ERROR,75,PATH/FIL
1900 E ACCESS ERROR,76,PATH NOT FOUND

```

HCM

## THE ORGANIZER REPORTS

TI-99/4A

```

100 *****
110 * THE ORGANIZER *
120 * REPORTS *
130 *****
140 COPYRIGHT 1985
150 EMERALD VALLEY PUBLISHING CO.
160 BY WILLIAM K. BALTHROP
170 HOME COMPUTER MAGAZINE
180 VERSION 5.2.1
190 TI EXTENDED BASIC
200 WITH 32K MEMORY EXPANSION
210 AND DISK MEMORY SYSTEM
220 SAVE THIS PROGRAM WITH
230 THE FILE NAME "REPORTS"
240 REPLACING "DUMMY REPORTS"
250 FROM VOL. 5, NO. 1
260 "THE ORGANIZER" PROGRAM
270 ON ERROR 1200
280 CALL INIT : CALL LOAD(-31878,0)::
290 CALL CLEAR
300 ON ERROR 1210
310 DIM L$(501),PAGES(60),OS(12,2)
320 BONS=CHR$(27)&CHR$(69)&CHR$(27)&CHR
330 $(71)::BOFFS=CHR$(27)&CHR$(70)&CHR
340 $(27)&CHR$(72)::BK$=CHR$(15)
350 RESTORE 1150::FOR Z=0 TO 12::RE
360 AD OS(Z,1),OS(Z,2)::NEXT Z
370 DEF STN(Q)=ASC(SEGS(Z$,Q*2+1,1))*25
380 6+ASC(SEGS(Z$,Q*2+2,1))
390 DEF L(Q)=ASC(SEGS(L$(RP),Q*2+1,1))*
400 256+ASC(SEGS(L$(RP),Q*2+2,1))
410 CALL SCREEN(5)::FOR Z=1 TO 14::C
420 ALL COLOR(Z,16,5)::NEXT Z
430 LOAD FILE LINKS
440
450 DISPLAY AT(1,7)ERASE ALL:"THE ORGAN
460 IZER":TAB(10):"REPORTS"
470 DISPLAY AT(4,1):"ENTER FILE NAME:DS
480 K1.FILE::ACCEPT AT(4,20)SIZE(-10
490 ):FS::IF FS=" " THEN 1190
500 OPEN #1:"DSK"&FS&"_L",RELATIVE,INTE
510 RNAL,FIXED,11::OPEN #2:"DSK"&FS&"
520 _D",RELATIVE,FIXED,27

```

Continued



```

410 DISPLAY AT(8,1):"READING #":GOSUB
B 1110:FOR R=2 TO NR+1:GOSUB
1130:DISPLAY AT(8,11):R-1:NEX
T R
420 DISPLAY AT(16,1):"ENTER PRINTER DEV
ICE NAME":ACCEPT AT(18,1)SIZ
E(-28):PRNT$
430 I
440 I
450 I
460 I
470 DISPLAY AT(1,10)ERASE ALL:"REPORTS"
:TAB(128-LEN(FS))/2:FS
GOSUB 510:PR=23:PL=11:GOSUB
B 1080:IF AS="N" OR AS="n" THEN
480 GOSUB 760
490 CALL CLEAR:PR=12:PL=12:GOSUB
UB 1080:IF AS="N" OR AS="n" THEN
500 GOSUB 1100:GOTO 460
510 PR=4:PL=0:GOSUB 1080:IF AS
=BK$ THEN RETURN ELSE IF AS<>"I" THE
N IF OS(PL,2)=AS:GOTO 540
520 IF OS(PL,2)=AS:GOTO 540
530 PR=PR+1:PL=1:GOSUB 1090:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
540 PR=PR+1:PL=2:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
550 IF OS(PL,2)=AS:GOTO 540
560 PR=PR+1:PL=3:GOSUB 1090:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
570 IF VAL(OS(PL,2))<1 OR VAL(OS(PL,2))
>5 THEN PR=PR-1:GOTO 560
580 PR=PR+1:PL=4:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
590 IF OS(4,2)=AS:GOTO 540
600 GOSUB 610:IF AS=BK$ THEN RETURN
610 PR=PR+1:PL=5:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
620 IF OS(5,2)=AS:GOTO 540
630 PR=PR+1:PL=6:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
640 IF OS(6,2)=AS:GOTO 540
650 PR=PR+1:PL=7:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
660 IF OS(7,2)=AS:GOTO 540
670 IF OS(7,2)=AS:GOTO 540
680 PR=PR+1:PL=8:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
THEN OS(PL,2)=AS:GOTO 540
690 IF OS(PL,2)=AS:GOTO 540
700 GOSUB 610:IF AS=BK$ THEN RETURN
710 PR=PR+1:PL=9:GOSUB 1080:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
720 PR=PR+1:PL=10:GOSUB 1090:IF
AS=BK$ THEN RETURN ELSE IF AS<>"I"
730 RETURN
740 DISPLAY AT(20,1):"YOU MUST PRINT EI
THER THE OUTLINE OR THE TEXT":
CALL SOUND(100,110,0):FOR TD=1 TO
1000:NEXT TD
750 CALL HCHAR(4,1,32,640):RETURN
760 FRMT=0:LEV=0:LNC=1:RP=FPFC
770 IF OS(4,2)=AS:GOTO 800
780 IF OS(4,2)=AS:GOTO 800
790 IF OS(4,2)=AS:GOTO 800
800 IF OS(4,2)=AS:GOTO 800
810 IF L(1)>0 AND LEV<VAL(OS(1,2))AND(
OS(7,2)=AS:GOTO 800
820 IF L(3)>0 THEN RP=L(3):GOTO 770
830 IF L(6)>0 THEN RP=L(6):LEV=LEV-1
840 IF LNC=1 AND PAGES(1)= THEN RETUR

```

```

850 GOSUB 1060:LNC=1:RETURN
860 TRP=RP:RP=L(4):FRMT=1:GOTO 870
870 R=RP:GOSUB 1140:GOTO 870
880 IF L(3)>0 THEN RP=L(3):GOTO 870
890 GOSUB 1040:RP=TRP:RETURN
900 IF OS(2,2)=AS:GOTO 870
910 TABSP=LEV*VAL(OS(3,2))ELSE TABSP=0
920 IF VAL(OS(10,2))<20 THEN TABS
P=VAL(OS(10,2))-20
930 IF OS(9,2)=AS:GOTO 870
940 PAGES(LNC)=HS:GOSUB 1040:RETU
RN
950 IF FRMT=1 THEN IF HS="*P" OR HS="*B"
OR HS="*P"
960 BS=MXL-LEN(PAGES(LNC)):IF BS=0 TH
EN GOSUB 1040:GOTO 960
970 IF LEN(HS)<BS THEN PAGES(LNC)=PAGES
(LNC)&BS:GOTO 960
980 FOR Z=BS TO 1 STEP -1:IF SEG$(HS
,Z,1)= THEN P=Z-1:IF P=0 THEN
1000 NEXT Z:GOSUB 1040:GOTO 960
1010 PAGES(LNC)=PAGES(LNC)&BS:GOTO 960
1020 GOSUB 1040:GOTO 960
1030 IF HS= THEN RETURN ELSE IF ASC(HS
)=32 THEN HS=SEG$(HS,2,LEN(HS)-1):
GOTO 1030
1040 IF FRMT=0 THEN PAGES(LNC)=RPTS(" ",
TABSP)&SPONS&PAGES(LNC)ELSE PAGES(L
NC)=RPTS(" ",TABSP)&BOFFS&PAGES(LNC)
1050 IF LNC=60 THEN GOSUB 1060:LNC=1
:RETURN ELSE LNC=LNC+1:RETURN
1060 OPEN #3:PRINTS:FOR Z=1 TO 60:P
RINT #3:PAGES(Z):NEXT Z:PRINT
#3:CHR$(12):CLOSE #3
1070 FOR Z=1 TO 60:PAGES(Z)=:NEX
T Z:RETURN
1080 DISPLAY AT(PR,1):OS(PL,1):TAB(26):O
S(PL,2):ACCEPT AT(PR,26)SIZE(-1)
1090 VALIDATE("YyNn"):AS:RETURN
1100 DISPLAY AT(PR,1):OS(PL,1):TAB(26):O
S(PL,2):ACCEPT AT(PR,26)SIZE(-3)
1110 VALIDATE(DIGIT):AS:RETURN
1120 FOR Z=1 TO 60:PAGES(Z)=:NEX
T Z:RETURN
1130 INPUT #1,REC 0:ZS:NR=STN(0):NU
SE=STN(1):FEMP=STN(2):LEMP=STN(
3):HFIR=STN(4)
1140 INPUT #1,REC 1:ZS:HLST=STN(0):
THFIR=STN(1):THLST=STN(2):FPFC=
STN(3):RETURN
1150 INPUT #1,REC R:LS(R):RETURN
1160 DATA SELECT GENERATION(Y/N)?,N
DATA MAXIMUM GENERATION?,0,INDENT
GENERATIONS(Y/N)?,Y,INDENT WIDT
H(1-5)?,3,PRINT OUTLINE(Y/N)?,Y,
MARK HEADERS(Y/N)?,Y
1170 DATA TEXT(Y/N)?,Y,PRINT HEADERS(Y
/N)?,Y,FORMAT TEXT(Y/N)?,Y,PRINTO
UT WIDTH(20-132)?,80
1180 DATA IS THIS OK(Y/N)?,Y,PRINT ANOT
HER ONE(Y/N)?,N
1190 DISPLAY AT(21,1)ERASE ALL:"LOADING
MAIN MENU":RUN "DSK1.ORGANIZE
1200 DISPLAY AT(8,1)ERASE ALL:"THIS PROG
RAM REQUIRES THE 32K MEMORY EXPAN
SION":GOTO 1260
1210 CALL ERR(A,B,C,D):IF A<>130 AND A
<>109 THEN GOTO 1250
1220 DISPLAY AT(6,1)ERASE ALL:"THERE IS
A PROBLEM WITH THE DISK I/O SYSTEM.
HAVING TROUBLE WITH OPENING OR:
READING THE FILE."
1230 DISPLAY AT(11,1):"PLEASE CHECK YOUR
DISK DRIVE AND DISK:YOU MUST H
AVE A VALID ORGANIZER FILE TO USE
THIS PROGRAM."
1240 GOTO 1260
1250 DISPLAY AT(12,1)ERASE ALL:"ERROR DE
TECTED IN PROGRAM:"ERROR CODE="A:
ERROR TYPE="B:LINE #":D:GOTO
1260
1260 CALL SOUND(500,110,0)
1270 DISPLAY AT(22,1):"PRESS ENTER TO RE
START THIS PROGRAM:"PRESS 9 TO RET
URN TO MENU"
1280 CALL KEY(0,K,S):IF S=0 THEN 1280
1290 IF K<>13 THEN 1310
1300 FOR Z=1 TO 60:PAGES(Z)=:NEX
T Z:
1310 LNC=1:ON ERROR 1210:CL
OSE #1:CLOSE #2:RETURN 380
1310 IF K=57 OR K=15 THEN 1190 ELSE 1280

```





# 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™ at the same time the corrections appear in print. This is of special significance to Apple, IBM, and TI (Extended BASIC programs only) ON DISK™ 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. We are currently working on an easy method of "update merging" for 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 you are working with the right listing to ensure satisfactory results.

## MYSTERY WORDS

HCM Vol. 4, No. 2 / Last level of DeBUGS published—version .1

```

APPLE II Family
160 REM VERSION 4.2.2
183 IF PEEK(103)=1 AND PEEK(104)
185 POKE 103,1:POKE 104,64:POKE 16386,0
187 PRINT CHR$(4):RUN MYSTERY
341 VTAB=20:HTAB=1:PRINT TIME LIMIT
    TO ANSWER:PROBLEM:PRINT (3 TO 60
    ) SECONDS:GET A1$:IF A1$ < "0"
    OR A1$ > "9" THEN 341
342 PRINT A1$:
343 GET A2$:IF (A2$ < "0" OR A2$ > "9"
    ) AND A2$ = CHR$(13) THEN 343
344 IF A2$ = CHR$(13) THEN TLIM = VA
    L(TLIM):GOTO 346+A2$
345 IF TLIM = VAL(A1$) OR A1$+A2$
346 HOME:VTAB=21:PRINT "PRESS YOUR C
470 NTRONL KEY WHEN YOU THINK YOU KNO
    W THE ANSWER"
490 VTAB=23:HTAB=1:PRINT SPC(40):
    FOR ZZ=1 TO TLIM*17:ZA=PEEK
    (16384):POKE 16368,0:VTAB=2
    3:PRINT "TIME LEFT:":INT(TLIM
    ZZ/17):IF ZA=128=48 OR Z
    A=128=49 THEN 500
495 NEXT OF:TIME:CHR$(7):CHR$(7):"O
    UT NEXT:GOTO 400
500 F=1:A=Z:176:IF A=1 THEN B
530 VTAB=10:PRINT "OK,";P$(A):TR=0
575 IF TR=1 GOTO 585
580 PRINT:PRINT "THAT'S NOT RIGHT";P
    $(A):POKE 16368,0:FOR DE=1 TO
    2000:NEXT:POKE 16368,0:A=VTAB
    B:IF F=1 THEN F=0:HOME:VTAB
    10:PRINT "IT'S YOUR TURN TO TRY"
    :P$(A):TR=1:GOTO 540
585 PRINT:PRINT "THAT'S NOT RIGHT";P
    $(A):PRINT "THE CORRECT ANSWER IS
    :ANS
630 PRINT "THE SCORE IS:":PRINT P$(0)
    :S1:PRINT:
    GOTO 3900 REM VERSION 4.2.2
  
```

## TAX DEDUCTION FILER

HCM Vol. 4, No. 4 / Last level of DeBUGS published—version .2

```

TI-99/4A
150 REM VERSION 4.4.3
330 IF L/19<>INT(L/19) AND Z<>R THEN 440
    ELSE DISPLAY AT(20,1):"ENTER NUMBE
    R":OR PRESS ENTER FOR MORE
340 ACCEPT AT(22,1)VALIDATE(DIGIT)SIZE(
    3):ES::IF ES=" " AND Z=R THEN 200
    ELSE IF ES=" " THEN 500
345 E=VAL(ES):IF E>R OR E<1 THEN CALL
    SOUND(50,220,0):GOTO 340
490 IF L/19<>INT(L/19) AND Z<>R THEN 440
    ELSE DISPLAY AT(20,1):"PRESS ENTER
    FOR MORE":GOSUB 860
500 CALL CLEAR::L=1::NEXT Z::GOTO
    200
720 DISPLAY AT(9,1):"FILE NAME [":FNS:T
    AB(22):" ]":ACCEPT AT(9,12)SIZE(-
    10):FNS::ON ERROR 960
725 OPEN #1:DSK1:"&FNS,VARIABLE 60,INP
    UT::GOTO 740
730 DISPLAY AT(9,1):"ENTER DEVICE AND F
    ILE NAME:":DEVS::ACCEPT AT(10,1):
    DEVS::ON ERROR 960::OPEN #1:DEV
    $,VARIABLE 60,INPUT
740 INPUT #1:R::FOR Z=1 TO R::LINPU
    T #1:AS(Z):NEXT Z::CLOSE #1::
    GOTO 200
750 DISPLAY AT(1,7):M$(7):"1) CS1 - C
    ASSETTE":2) DSK1 - DISK DRIVE #1
    :3) OTHER DEVICE":4) ABORT F
    UNCTION"
760 GOSUB 860::IF K<49 OR K>52 THEN 7
    60 ELSE ON K-48 GOTO 770,780,790,20
    0
780 DISPLAY AT(9,1):"FILE NAME [":FNS:T
    AB(22):" ]":ACCEPT AT(9,12)SIZE(-
    10):FNS::ON ERROR 960::OPEN #1:
    "DSK1:"&FNS,VARIABLE 60,OUTPUT
785 GOTO 800
790 DISPLAY AT(9,1):"ENTER DEVICE AND F
    ILE NAME:":DEVS::ACCEPT AT(10,1):
    DEVS::ON ERROR 960::OPEN #1:DEV
    $,VARIABLE 60,OUTPUT
955 REM DISK ERROR ROUTINE
960 PRINT "FILE CAN'T BE OPENED":CAL
    L SOUND(200,110,0):FOR DE=1 TO 50
    0:NEXT DE::CALL CLEAR::RETUR
    N 200
  
```

## FILE MANAGER

HCM Vol. 5, No. 1 / Last level of DeBUGS published—version .1

```

APPLE II Family
180 REM VERSION 5.1.2
1025 GOSUB 2120
1530 REM ***DELETED LINE
  
```

## BOOLEAN BRAIN

HCM Vol. 4, No. 4 / Last level of DeBUGS published—version .1

```

COMMODORE 64
150 REM VERSION 4.4.2
1170 GETA$:IFA$<"0" OR A$>"9" THEN GOTO 1170
  
```



# SEA OF STATES

HCM Vol. 4, No. 2 / Last level of DeBugs published—version .2

This DeBug is for disk systems only.  
For cassette DeBug refer to *DeBugs On Display*,  
HCM Volume 4, Number 4, page 130.

		TI-99/4A	
100	** Deleted line...	510	AS="A TALKING OCTOPUS AMBUSHES YOU
120	** Deleted line...	520	IF SB=6 THEN SH=116: GOSUB 660
150	REM VERSION 4.2.4		RE SCARED BY THE OCTOPUS: YOU LOSE
170	CALL CLEAR: PRINT "IF YOU'RE USIN		: GOSUB 1070: C=-1: GOTO 48
	CALL DISK DRIVE (W/O MEM. EXPANSION)	530	DISPLAY AT (15,1): "HE SAYS: "YOU MU
172	YOU NEED TO TYPE: "SEA OF STATES"		ST ANSWER THIS QUESTION: "X=
	PRINT "CALL FILES(1), AND 'NEW'.	570	S(5): Y=S(6): V=6: GOSUB 1075
175	THEN RELOAD THE PROGRAM.		DISPLAY AT (15,1): "SORRY THAT'S WRO
180	FOR TD=1 TO 2000: NEXT TD: DIM SS(8		NG, &ST\$(ST,1)&: IT FIGHTS: "YOU AN
190	CALL CLEAR: RANDOMIZE: CALL SCREEN(5)		D YOU LOSE: "GOTO 480
	: ST\$(5,2): S(7): CALL SCREEN(5)	610	CALL MOTION(#1,0,-9): DISPLAY AT (1
220	DEF SB=ASC(SEG\$(S(5)),S(6),1)-4		S,1): "CORRECT, THE OCTOPUS: GOTO
	: S(5): S(6): S(7): S(8): S(9): S(10):	620	SES YOU AND GIVES YOU: GOTO 480
230	CALL SOUND(100,-6,0): RESTORE 1080		AS="SHARKS ATTACK YOU: YOU LOSE
	: CALL MOTION(#1,0,-1,2,0,-1):		C=2: SH=120: GOSUB 660: C=-1
250	DISPLAY AT (10,8): READ ST\$(A,1), ST\$(		: GOTO 480
	A,2): NEXT A: DISPLAY AT (12,1):	630	AS="YOU HAVE FOUND A WRECK AND: "
250	YOU ARE IN SEARCH OF GOLD!		C=7: SH=124: GOSUB 660: X=S(
	DISPLAY AT (16,1): "YOU WILL ALSO FIN		5): Y=S(6): V=5: GOSUB 1075
254	RT OCTOPI: "SHARKS" WRECKS: AND SMA		GOTO 480
260	GOSUB 260: GOTO 270	640	AS="YOU ARE AT A DIVING BELL, A SAF
	FOR A=1 TO 8: SS(A)=": NEXT A		E WAY UP: C=7: SH=128: GOSU
	: FOR A=1 TO 8: FOR B=1 TO 8: S	650	B 660: GOTO 330
	SS(A)=SS(A)&STR\$(INT(TAN(RND*1.37)))		AS="YOU HAVE FOUND A WRECK. IT'S AL
270	: NEXT B: NEXT A: RETURN		READY BEEN SALVAGED: C=7: SH=1
	CALL DELSPRITE(#3): X=INT(RND*8)+1	700	24: GOSUB 660: GOTO 330
	: Y=INT(RND*8)+1: V=4: GOSUB	780	** Deleted line...
320	1075 DELSPRITE(#1,#3,#4): CALL HCH		DISPLAY AT (15,1): "THE SHARKS FIND Y
	AR(15,1,32,256): ON INT(SB+1) GOTO	820	OSUB 1070: GOSUB 1070: GOTO 960
360	440,510,620,630,640,650,660,670,680		FOR A=5 TO 15 STEP 5: DISPLAY AT (
	CALL KEY(0,A,B): IF A>87 OR A<69 T		17,A): "UP": GOSUB 1070: NEXT A
365	HEN 360 ELSE IF S(5)>7 AND A=78 OR	930	: GOSUB 260: DISPLAY AT (20,1): "Y
	S(5)<2 AND A=83 OR S(6)>7 AND A=69		OU'RE AT LEVEL 2: GOSUB 1070
	OR S(6)<2 AND A=87 THEN 1010		DISPLAY AT (10,9): "CONGRATULATIONS!
370	ON A=68 GOTO 420,360,360,360,360,360		: "YOU WENT AND RETURNED FROM: "
400	10,360,710,380,430,360,360,360,4	940	: "THE SEA OF STATES": GOTO 960
410	** Deleted line...	980	** Deleted line...
420	S(5)=S(5)+1: GOTO 320		PRINT "YOU GOT "&STR\$(S(3)/S(4)+100
430	S(5)=S(5)-1: GOTO 320	1040	)&"% RIGHT.
440	S(6)=S(6)+1: GOTO 320		AS="YOU'RE AT THE BOTTOMLESS PIT":
450	S(6)=S(6)-1: GOTO 320		CALL CHAR(132,RPT\$(RPT\$(0,24)&R
	IF SB=0 THEN ON INT(RND*3+1) GOTO 45		PT\$(0,8),2): C=5: SH=132: G
	: 460,470 ELSE 510	1075	OTO 1060
	AS="YOU FIND ONLY SAND & SHELLS.":		SS(X)=SEG\$(S(X),1,Y-1)&STR\$(V)&SEG
	C=16: SH=112: GOSUB 660: GO	1120	\$ (S(X),Y+1,LEN(S(X))-Y): RETURN
	TO 330,470 AS="THE CORAL IS ATTRACTI		DATA NEW HAMPSHIRE,CONCORD,NEW JERS
	VE BUT WORTHLESS.": C=10: SH=10		EY,TRENTON,NEW MEXICO,SANTA FE,NEW
	S: GOSUB 660: GOTO 330		YORK,ALBANY,NORTH CAROLINA,RALEIGH,
			NORTH DAKOTA,BISMARCK,OHIO

## OUTLINE EDITOR

HCM Vol. 5, No. 1 / Last level of DeBugs published—version .1

		TI-99/4A	
180	I VERSION 5.1.2	150	REM VERSION 4.4.3
1610	DISPLAY AT (5,1): "SELECT ONE: "DI	260	REM ***DELETED LINE
	PLAY AT (7,2): "1) CURRENT PARENT":	3560	PRINT "THIS GAME IS A SIMULATION OF
	: "2) ONE LEVEL": "3) ENTIRE OUTL		STOCK MARKET TRANSACTIONS":
1850	INE"	3570	PRINT "OVER A SET PERIOD OF TIME. IN
	DISPLAY AT (17,1): "ENTER LEVEL TO SO		THE SIMULATION, YOU COMPETE":
	RT: "ACCEPT AT (17,21) SIZE(3) VALI	3580	PRINT "WITH OTHER PLAYERS OR ALO
1860	DATE(DIGIT): SRL		NE IF YOU WISH. "
	S=0: DISPLAY AT (20,1): "LEVEL: "S	3600	PRINT "MONEY BEFORE THE ENDOF PLAY.
	: SP=FPFC: IF SRL=0 THEN SP=0: "		IN ORDER: "
	GOSUB 1680: RETURN ELSE S=S+1: "	3610	PRINT "TO DO THIS, HERE ARE SOME GA
	GOSUB 1980		ME-PLAYING CLUES: "
1930	SP=0: S=0: DISPLAY AT (20,1): "LE	3620	PRINT "CRSRDOWN: BUY LOW, SELL AND/O
	VEL: "S: GOSUB 1680: SP=FPFC: "		RTRADE FOR PROFIT. "
	GOSUB 1680: CALL HCHAR(18,1,32,3	3630	PRINT "USE LOANS TO FATTEN YOUR PORT
3130	2)		FOLIO, BUT BEWARE OF: "
	MR,CC=SCR(1): FOR Z=1 TO 12: CAL	3640	PRINT "THE 'BEAR' MARKET!
	L GL(LS(MR),2,T1): IF T1=0 THEN SF		GOOD LUCK!!
	C=MR: GOSUB 4200: RETURN: ELS	3650	REM ***DELETED LINE
	E MR=T1	3660	REM ***DELETED LINE
3140	NEXT Z: SFC=MR: GOSUB 4200: R	3670	REM ***DELETED LINE
	ETURN	3680	REM ***DELETED LINE
3160	SFC=SCR(12): CC=SCR(21): GOSUB 42	3880	PRINT "BE PRINTED IF YOU TRY TO DO
	00: RETURN	3890	SO. "
			REM ***DELETED LINE

## MARKET MADNESS

HCM Vol. 4, No. 4 / Last level of DeBugs published—version .2

		COMMODORE 64	
150	REM VERSION 4.4.3	150	REM VERSION 4.4.3
260	REM ***DELETED LINE	260	REM ***DELETED LINE
3560	PRINT "THIS GAME IS A SIMULATION OF	3560	PRINT "THIS GAME IS A SIMULATION OF
	STOCK MARKET TRANSACTIONS":		STOCK MARKET TRANSACTIONS":
3570	PRINT "OVER A SET PERIOD OF TIME. IN	3570	PRINT "OVER A SET PERIOD OF TIME. IN
	THE SIMULATION, YOU COMPETE":		THE SIMULATION, YOU COMPETE":
3580	PRINT "WITH OTHER PLAYERS OR ALO	3580	PRINT "WITH OTHER PLAYERS OR ALO
	NE IF YOU WISH. "		NE IF YOU WISH. "
3600	PRINT "MONEY BEFORE THE ENDOF PLAY.	3600	PRINT "MONEY BEFORE THE ENDOF PLAY.
	IN ORDER: "		IN ORDER: "
3610	PRINT "TO DO THIS, HERE ARE SOME GA	3610	PRINT "TO DO THIS, HERE ARE SOME GA
	ME-PLAYING CLUES: "		ME-PLAYING CLUES: "
3620	PRINT "CRSRDOWN: BUY LOW, SELL AND/O	3620	PRINT "CRSRDOWN: BUY LOW, SELL AND/O
	RTRADE FOR PROFIT. "		RTRADE FOR PROFIT. "
3630	PRINT "USE LOANS TO FATTEN YOUR PORT	3630	PRINT "USE LOANS TO FATTEN YOUR PORT
	FOLIO, BUT BEWARE OF: "		FOLIO, BUT BEWARE OF: "
3640	PRINT "THE 'BEAR' MARKET!	3640	PRINT "THE 'BEAR' MARKET!
	GOOD LUCK!!		GOOD LUCK!!
3650	REM ***DELETED LINE	3650	REM ***DELETED LINE
3660	REM ***DELETED LINE	3660	REM ***DELETED LINE
3670	REM ***DELETED LINE	3670	REM ***DELETED LINE
3680	REM ***DELETED LINE	3680	REM ***DELETED LINE
3880	PRINT "BE PRINTED IF YOU TRY TO DO	3880	PRINT "BE PRINTED IF YOU TRY TO DO
	SO. "		SO. "
3890	REM ***DELETED LINE	3890	REM ***DELETED LINE



## BOOLEAN BRAIN

HCM Vol. 4, No. 3 / Updated to include the PC—not a DeBug

IBM PC & IBM PCjr

```

160 REM IBM PCjr WITH CARTRIDGE BASIC
170 REM IBM PC FROM DOS 2.1 OR WITH COLOR
180 REM COLOR GRAPHICS ADAPTER AND
190 CLS:SCREEN 1:DIM A(2,10),RM(10,4):R
    ANDOMIZE TIMER:KEY OFF
660 LINE(95,55)-(223,95),0,BF:FOR Z=1 T
    O 200:X=INT(RND*125)+95:Y=INT(RND*3
    7)+55:LINE(X,Y)-(X+3,Y+3),INT(RND*
    2)+1,BF:NEXT
670 FOR Z=1 TO 200:SOUND RND*10000+110,
    1:FOR TD=1 TO 50:NEXT:SCORE=IN
    T(ND/SC/ND*10000)
680 LINE(0,155)-(319,199),0,BF:LOCATE
    20,1:PRINT CONGRATULATIONS—YOU
    HAVE FOUND THE CENTRAL PROCESSING
    UNIT.:IF SCORE<600 THEN PRINT "TH
    E COMPUTER IS DAMAGED. YOU LOSE
    BECAUSE OF DATA LOSS." ELSE PRIN
    T "YOU REPAIR THE COMPUTER AND ESCA
    PE."
690 FOR TD=1 TO 3000:NEXT:PRINT "YOUR S
    CORE="SCORE:PRINT:PRINT "PLAY AGA
    IN (Y/N)?"::GOTO 720
700 FOR Z=1 TO 15:FOR Y=1 TO 15:SOUND Y
    *100+1000,2:NEXT:NEXT
840 ND=ND+1:FOR Z=1 TO 10:GT(Z)=0:NEXT:
    CLS:DRAW "S8":COL=1:FOR B=1 TO 9:T(
    B)=INT(RND*2)+1:Y=A(1,B):Z=A(2,B):O
    N T(B) GOSUB 970,980:NEXT:DRAW "S4"
850 FOR Z=16 TO 176 STEP 40:LINE(10,Z)
    NEXT
    (24,Z),1:LINE(10,Z+8)-(24,Z+8),1:
    NEXT
860 COL=1:DRAW "S4":GOSUB 730:GOSUB 740
    :GOSUB 750:GOSUB 760:GOSUB 770:GOSU
    B 780:GOSUB 790:GOSUB 800:GOSUB 810
    :GOSUB 820:COL=0
870 REM
890 GOTO 880
900 COLOR 0,0:CLS:DRAW "C3BM0,23M79,47M
    239,47M319,23BM319,143M239,119NU72M
    79,119NU72M0,143BM47,87P1,3BM159,87
    P2,3BM287,87P1,3BM159,23P3,3BM159,1
    60P3,3":RETURN
910 DRAW "C3BM143,119U56R32D56BH5P=C:,3
    BM183,79C3R16D16L16U16BF2D5R12U5L12
    BD9D3R12U3L12BFP2,3BU6P3,3BD3P2,3
920 DRAW "C3BM7,140M7,51M47,59M47,129BH
    3P=C:,3C3BM71,79L16D16R16U16BG2D5L1
    2U5R12BD9D3L12U3R12BGP2,3BU2P3,3BU4
    P2,3
930 DRAW "C3BM271,129M271,59M311,51M311
    ,140BM287,119P=C:,3":RETURN
940 PAINT(159,65),2,3:RETURN
950 PAINT(8,52),2,3:RETURN
960 PAINT(310,52),2,3:RETURN
1010 DATA KEYBOARD,INTERFACE,5,4,8,7,INP
    UT,PORT,4,3,6,9,VIDEO PROCESSING,2,
    5,7,4,SOUND CONTROL ROOM,1,2,3,5,RA
    M ROOM,3,1,4,8,DISK CONTROLLER ROOM
    ,7,8,9,2,DISK DRIVE,7,6,1,3,ROM ROO
    M,6,9,5,1,PORT CONTROL,8,10,2,6,CEN
    TRAL PROCESSING CONTROL,9,10,10,10
  
```

## MISSILE MATH

HCM Vol. 4, No. 3 / Last level of DeBugs published—version .1

IBM PC & IBM PCjr

```

160 REM VERSION 4.3.4
170 REM IBM PCjr WITH CARTRIDGE BASIC FROM
    DOS 2.1 OR
180 REM IBM PC BASICA WITH COLOR
    GRAPHICS ADAPTER AND
190 REM COLOR MONITOR
480 LOCATE 23,1:PRINT "***** GOOD WOR
    K—THAT'S RIGHT *****"
    "YOUR ANSWER WAS";CHR$(ANS+64);
    "*****"
590 CLS:LOCATE 3,10:PRINT "MULTIPLICATI
    ON GAME":PRINT:PRINT "EACH MISSILE
    CONTAINS AN ANSWER.:PRINT "YOU MUS
    T CHOOSE THE ANSWER WHICH FITS TH
    E PROBLEM SHOWN BELOW.:PRINT:PRINT
    "PRESS EITHER A B OR C FOR THE ANS
    WER.":RETURN
  
```

## MARKET MADNESS

HCM Vol. 4, No. 4 / Last level of DeBugs published—version .2

IBM PC & IBM PCjr

```

160 'VERSION 4.4.3
170 'IBM PCjr WITH CARTRIDGE BASIC
    FROM DOS 2.1
620 LOCATE 21,20:PRINT STRINGS$(40,32)::
    LOCATE 22,1:PRINT "HOW MANY SHARES
    OF";S$(SS)::INPUT A$:GOSUB 1690:IF
    STAT=1 THEN RETURN ELSE NOS=INT(VA
    L(A$)):IF NOS>P(PN,SS) THEN LOCATE
    23,1:PRINT "YOU ONLY HAVE";P(PN,SS)
    ;"SHARES!":GOTO 620
630 LOCATE 23,1:PRINT SPC(39)::LOCATE 2
    3,1:PRINT "CURRENT VALUE="S$(SS):
    INPUT "SELLING PRICE";A$:GOSUB 169
    0:IF STAT=1 THEN RETURN ELSE SP=INT
    (VAL(A$)):IF SP<S$(SS)/2 OR SP>S$(SS)
    *2 THEN 630
  
```

## WILD KINGDOM

HCM Vol. 4, No. 3 / Last level of DeBugs published—version .1

APPLE II Family

```

160 REM VERSION 4.3.2
180 IF PEEK(104)=64 AND PEEK(103)
    THEN 190
182 POKE 104,64:POKE 103,1:POKE 16384
    0
185 PRINT CHR$(4);"RUN KINGDOM"
440 IF K=ASC("A") OR K=11 THEN 52
    0
470 IF K=ASC("Z") OR K=10 THEN 58
    0
  
```

## SNAP-CALC

HCM Vol. 4, No. 3 / Last level of DeBugs published—version .4

APPLE II Family

```

170 REM VERSION 4.3.6
5290 AS=CHR$(PEEK(-16384))-128)
5295 POKE -16368,0
  
```



# SENSATIONAL SOFTWARE GIVEAWAY

## 1 YOUR CHOICE OF THIS ISSUE'S PROGRAMS AT A REMARKABLE PRICE—



**ONLY \$4.95!**



Price will change to \$6.95 upon becoming a "Back Issue." (See date on order form below)

**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.**

## 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 \$4.95 (\$6.95 in Canada & Mexico, \$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.

## \*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 \$9.90 Premium Value) of ON TAPE™ or ON DISK™. Subscribe for 2 years and receive **4 Issues** (a \$19.80 Premium Value). And with a 3-year subscription we'll give you **6 full Issues** (a \$29.70 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!

AND ENJOY THE

## 3 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!**

## 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

**Please allow 6-8 weeks for your first issue—Magazine & Media shipped separately.**

Satisfaction Guaranteed—or the unfulfilled 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.

SGV3520385

### 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. 2

**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 (\$6.95 in Canada & Mexico, \$9.95 Foreign Airmail).

**\$4.95 PRICE GOOD THROUGH May 15, 1985**

**After this date, order back issue @ \$6.95 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**

ON TAPE or ON DISK

**PLUS 4 FREE**

ON TAPE or ON DISK

**PLUS 6 FREE**

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.

**Canada add \$15 for software subscription.**

**Software subscription not available in other countries at this time.**



# HOME COMPUTER<sup>TM</sup> 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+ /Ile ☐Commodore 64  
☐VIC-20 ☐IBM PC ☐PCjr ☐Other \_\_\_\_\_

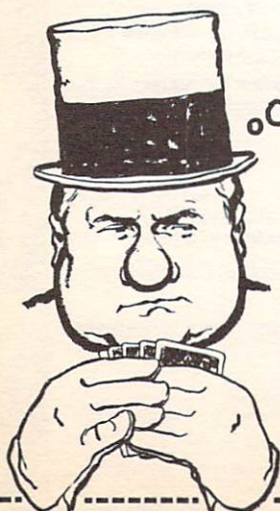
### FOR READERS WHO PLAN TO BUY A HOME COMPUTER

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+ /Ile ☐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<sup>TM</sup> 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 &amp; 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

ITEMS	PRICE
Home Computer Magazine Back Issues (Circle Issues Desired)	\$3.95 each — U.S. \$4.50 each — Canada \$5.50 each — Foreign Surface \$7.50 each — Foreign Air
ON DISK & ON TAPE Back Issues (Circle Issues Desired)	\$6.95 each — U.S. \$8.95 each — Canada \$10.95 each — Foreign Air
SAVE EVEN MORE—Order Combined Sets (Circle Magazine & Media Sets Desired)	\$8.90 each set — U.S. \$11.90 each set — Canada \$11.90 each set — Foreign Surface

Indicate your choice of media: (Check one)

ON TAPE<sup>TM</sup>: ☐ C-64 ☐ TI-99/4AON DISK<sup>TM</sup>: ☐ 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 &amp; Prices Subject To Change Without Notice.

## The Best Of 99'er

### —Book & Tape Set—

See Page 8

## Special Close-Out Offer

See Page 8

Please Print

Name \_\_\_\_\_

Address \_\_\_\_\_

City \_\_\_\_\_ State \_\_\_\_\_ Zip \_\_\_\_\_

YES! Please send me *THE BEST OF 99'er ON TAPE*  
along with myFREE copy of the book *The Best Of 99'er*.

and the SPECIAL BONUS (while supplies last)

of Simon's Saucer<sup>TM</sup> and the 99'er Programmer's Guide.

Enclosed is \$35. (Shipping and Handling FREE!)

**For more information see page 8.**

Offer &amp; 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 &amp; 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



## Save \$\$\$ On BACK ISSUES of



### SPECIAL CLOSE-OUT PRICES FOR MAGAZINES, DISKS & TAPES NOW IN EFFECT FOR TI-99/4A USERS!

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 &amp; 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

99'er Home Computer Magazine, Disk, & Tape Back Issues  
Exclusively For The TI-99/4A Home Computer

ISSUE	Magazine Only	Mag. & Media (Disk or Tape)	ISSUE	Magazine Only	Mag. & Media (Disk or Tape)	ISSUE	Magazine Only	Mag. & Media (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.

QTY	MAGAZINE PRICES			MAG/MEDIA SET PRICES			MEDIA PRICES		
	U.S.	Canada	Foreign	U.S.	Canada	Foreign	U.S.	Canada	Foreign
1	\$3.95	\$4.50	\$5.50	\$7.50	\$9.50	\$10.50	\$3.95	\$5.95	\$6.95
3	5.95	7.50	8.95	14.95	17.95	18.95	10.95	13.95	14.95
6	10.95	12.50	13.95	28.50	33.50	34.50	20.95	26.95	27.95
12	21.90	24.00	25.95	49.95	55.95	56.95	39.95	44.95	45.95

Defective media gladly exchanged.

NO REFUNDS on media.

SUB TOTAL \$ \_\_\_\_\_

Offer &amp; Prices Subject To Change Without Notice.

**For more information see inside back cover.**

BCV3520385



# GUIDE TO **HOME COMPUTER**<sup>TM</sup> magazine READER SERVICES

See Rear Bind-In Card

Subscriptions

See Inside Front Cover

Back Issues



See Rear Bind-In Card

ON TAPE<sup>TM</sup>



ON DISK<sup>TM</sup>



This Issue's Software

See Rear Bind-In Card

Program Subscriptions

See Inside Front Cover

Back Issues



Back Issues



See Inside Back Cover

The Best Of 99'er



See LISTINGS Contents page



Blank-Media Service



**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.



- 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.



**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

**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 Add • Color Mapping • Dynamic Manipulation of Screen Character Graphics • The Beginner's Guide to Cassette Operation With the Home Computer • The School Book Letters and Darts • Star Space Game • 3-D Animation on the Home Computer • Programming: A Who is LOGO? • 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 Crystal • and much, much more.

**NOVEMBER 1982 (Partial Contents)**  
• Chatting with Your Micro: Languages for the Home Computer • The Micro Jaws Arcade Game • A Wheel Printer • The Micro Jaws Arcade Game • A Knight's Tour in TI BASIC • LOGO Has Style • ASPIR: A Language for Children • A 2-System Beginner's Tutorial • An Interview with a System Pioneer • A Mini-Memory Screen Dump to the Home Computer Printer • Up Scoop—An Exciting Undersea Combat Game • Strategy for Munch Man • A Brief Introduction to the 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)**  
• Text Scriber: A Text Editor for the Home Computer • Christmas Computer Carol • Managing a Mailing List the Futura Way • Parsec: The Arcade Game • Plotting with the Home Computer • Fly By Five • Preventing the Situation—On Not Memory Full • A Colorful Tour of TI-Fest: The Home Computer Show • Santa's Workshop: The Making of a Home Computer • The New Arcade World • 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 • System Basics • Debugging in LOGO • The Dow & Goale Flight Simulator • A Home Computer Master Musical Game Review • 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 • Interview With an Arcade Game Designer • TI Invaders • Programming With Pascal • Cyber Cop • News and Happenings in the Home Computer World • Arcade Game Reviews • The TI Adventure Game • Programming Tips • and much, much more.

**FEBRUARY 1983 (Partial Contents)**  
• Texas Instruments at the Writer's Consumer Electronics Show • Home Computer Printers on Review • How to Create Math Datasets in LOGO • Vectors in LOGO • ASPIR: A Language for Teachers • The Joy of Advertising—Part 2 • Interview with the Voice of Parsec • Why You Need a Printer for Your Home Computer • Lifetime to Trap Space Game • Night Blockade Battleship Game • Tower of Hanoi Pocket Program • Computer Gaming Software Review • News of Late Developments in the World of Home Computers • and much, much more.

**MARCH 1983 (Partial Contents)**  
• An Introduction to the TI-99/4A Computer • The Hex Bus and the AIA Connection • Making Your Own Hex and Sool Game • Disabled Children's Games and Grow • Super Catalogue—A Review of a Disk Library Utility Program • TI's New CG-40 Compact Computer • Robots and Their Social Impact • Twenty Questions With Roddy Redford: The Gravity of LOGO • Jorjick: An Overview of Remote Computers • Parsec: Strategy • Converting Extended BASIC to Assembly Language • Main • Muncher • Mini • Memory Disassembler Utility • Pulling the Shade on Sprites • Letters on LOGO • Tiny Tutorials • Games, reviews, and much, much more.

**APRIL 1983 (Partial Contents)**  
• Computer Assisted Planning to Build Your Nest Egg • Text 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 Chess Puzzle • Super Landings • A Programming Lesson in Mini Memory • Colorful World—Reading Readiness for the subchapters • Games Review: A Maze-Run Boy Alley Game • Giant and Dwarf's Enchantment 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 VHS • The Multiplan Medium Balances Your Checkbook and Budget • Activity Accountant Helps School Secretaries with Extracurricular Activities • Maximizing Your Mini Memory • A Review of RAM • Exploring Enhanced BASIC on the Compact Computer • The LOGO Tortoise Debates the BASIC Hare • A Pocket Program to Organize Data • Linked Lists • Mentality Handicapped Learners Team Up with the TI-99/4A • The Wonders of Diskette Storage • Business • 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'er Connection • Tune Your Guitar with Our TI Tuning Fork • Talk to Your Computer—Voice Technology is Here • Gameware Buffet's Lat or Be Eaten: A 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 CG-40 • Go on a LOGO Vacation • Letters on LOGO • A Review of Upper Room Software's Programs for Special Language • Construct an RC2200system 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 Graphic Groupers • Five Data Organizers in Never Out of Sorts • TI-99'er at the Consumer Electronics Show • Wardrobe: The Music and the Book • Editing with Multiplan • The LOGO Logician Presents To Model is to Learn • LOGO Mosaic Designs Fit the Green • 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-Roo • A Book Review of Learn BASIC for CG-40/400 • 3-D Animation with the TI-99/4A Video Chip • Games Reviews • Group Grapevine • and much, much more.

**AUGUST 1983 (Partial Contents)**  
• The Home Computer Goes To Work • Bit One: Part Two at the Fashion Factory • Better Business Bar Graphs in Graphic Groupers • Accessaries and Peripherals • Peripherals: Vision 99 Hardware Reviews • Pocket Surprise: Part Two in Extended BASIC • A Review of Mini Memory • A Review of TI-99/4A • TI-99/4A • LOGO's Functions, Sets and Turtles • The CG-40 and 8044: Take the Data, and RUN • A 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 No Vill and Crime and Punishment • Extended BASIC • 99'er Hall of Fame • 99'er Digest Update on New Products, and much, much more!

**OCTOBER 1983 (Partial Contents)**  
• Education with your Home Computer • Five Creative Learning Activities for Children • Let's Build America • Have No Fear: Assembly Language VHS! • Part 2 • Squeezing the Most Out of TI BASIC • TI-99/4A Tutorial • LOGO Lesson • Interview with Dale Dobson • TI-99/4A: At Home in the Office • PLATO's Progress • Les Imore and Debug • The Multiplan Medium • Computer Arrested Instruction—99'er Interviews the Kids • Gameware Buffet: Taco Man and Robot Chase • Game Reviews of Jail Break and Arithmetica • Hall of Fame • 99'er Digest • and much, much more.

THESE 2 MAGAZINES OUT OF PRINT  
—TAPES AND DISKS ARE STILL AVAILABLE—

**Hurry—Supplies Are Limited. Order Yours Today**

Offer & Prices Subject To Change Without Notice.



# ALL PROGRAMS IN THIS MAGAZINE



## ONLY \$4.95\* DELIVERED RIGHT TO YOUR DOOR!

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\* postpaid (barely covering the cost of a blank floppy disk or cassette tape), you receive all the programs for your particular brand of computer —Truly A "Software Giveaway!"

To Order, Use The Bind-In Card Inside Rear Cover.

\* Current Issue Price Only — See Center Bind-In Card For Back Issue Prices. Offer & Prices Subject To Change Without Notice.