

PROOF OF
PURCHASE

Software Giveaway with Purchase of this Magazine

See back cover

HOME COMPUTERTM magazine

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

Vol. 4 No. 4

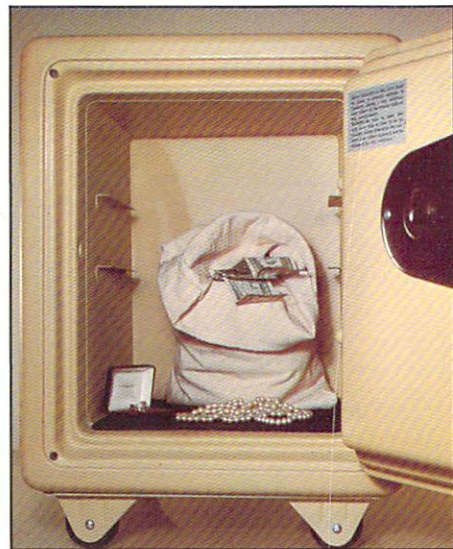
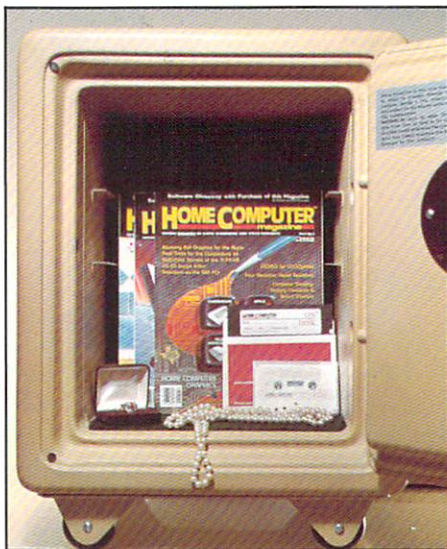
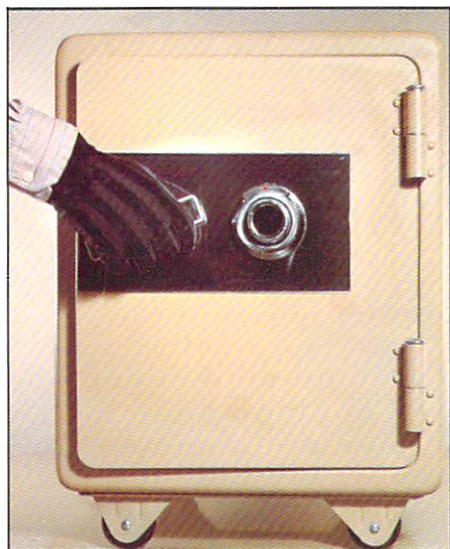
\$3.50 in USA
\$4.50 in Canada



Computer Sports

- The Real Truth About the Apple IIc
 - How To Add a 2nd PCjr Drive
 - Souping Up Your TI-99/4A with Speed BASIC & Double-Drive
 - Basic Sessions with Simon on the C-64
 - Dozens of Ready-to-Run Programs—Filing, Investing, & Sports
- Plus Reviews Galore of the Season's Hottest Sports Games**

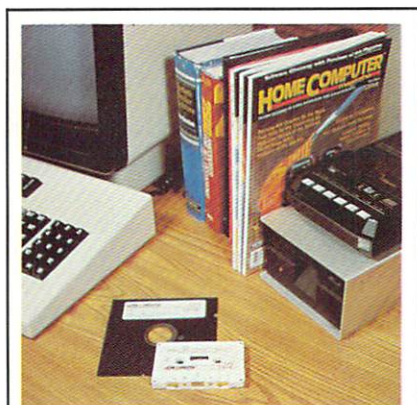




MISSING ANY VALUABLES?

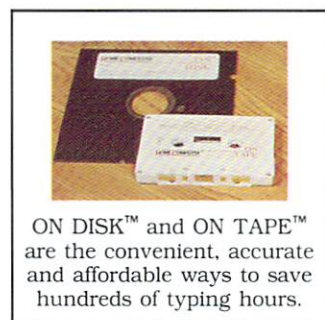
If you're missing any back issues of **HOME COMPUTER**[™] magazine—
you're missing more than you'll ever know . . .

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



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

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



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

Collect all the programs from each magazine issue on a ready-to-RUN quality floppy disk or cassette tape available in

separate versions for Apple, Commodore, IBM, and Texas Instruments home computers.

**“Safeguard” Your Home Computer Knowledge—
Order Valuable Back Issues Today!**

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

"The 3 Most Common Faults Of Computer Magazines are . . .

1 Diluted Content

For most computer magazines, editorial content plays second fiddle to advertising. In many of today's "successful" publications, strong tutorial and programming material is sparse—lightly sprinkled in between page after page of splashy ads—almost as an afterthought . . . Some of the "most successful" magazines try to compensate for this weak content ratio by printing and binding an issue that is three to four times the amount of paper a reader can comfortably handle and digest. And in the "less successful" publications, the void from unsold ad space is packed with "fluff"—items like re-hashed press releases and photos, filler articles in search of a story, "big name" opinion columns, and old "news" items that contribute virtually nothing to a reader's computer knowledge and enjoyment.



2 Clumsy Design

Unfortunately, layout and design also play second fiddle to advertising space needs. Intervening advertisements break up the "flow" of textual and visual material, making comprehension *more difficult*. Furthermore, all the effort spent in producing clever illustrations, crisp photos, creative typography, and harmonious color usage is often for naught—thanks to visual clashes with adjacent advertising "art."

3 Slanted Focus

Some computer magazines have been known to "do anything to get an ad"—such as publishing "canned" stories touting advertisers, taming down reviews, ignoring competing products from non-advertisers, etc. However, most computer magazines today live by a strong code of ethics. But ethics *isn't* the entire problem. For although a magazine professes to have an editorial content "untarnished" by its advertising content, few (if any) go out of their way to avoid a more subtle bias—editorial calendars dictated by ad-sales staff needs, and the expedient practice of dispensing only *good* reviews because of ". . . too many new products to waste limited magazine space reviewing bad ones."

. . . And Here's What We Just Did About Them."

We on the staff of *Home Computer Magazine* have a standard of "success" that is different from all the rest. When a magazine's success depends upon how well it serves the *reader*, rather than how well it sells the *advertiser*, an amazing thing happens—excellence is inevitable.

All we ask is that you examine our magazine closely. Notice the care we take in balancing: (1) the amount of coverage per computer brand; (2) the article and program mix (of entertainment, productivity, education, and utilities); and (3) the comprehension level for a diverse group of readers. Notice, too, the full measure of high-quality software programs included in each issue. And you won't find anyone else who presents their magazine's software listings in as clear and consistent a format—or who offers *all* the issue's programs on floppy disk or cassette tape for only \$3.95 delivered! As for strong tutorial and "how-to" material, balanced reviews, and elegance of magazine design—you be the judge. We think you'll discover, now more than ever, what we've been saying all along: *Once you compare—there's no comparison.*™

FOR IMMEDIATE RELEASE

COMPUTER MAGAZINE MAKES UNPRECEDENTED MOVE

Emerald Valley Publishing Co. announced today that beginning with its September 1984 issue, *Home Computer Magazine* will no longer carry outside advertising.

According to publisher Gary M. Kaplan, "We want *Home Computer Magazine* to stand out and be recognized as the best publication in its field. By removing the advertising content from the magazine, we have the editorial and artistic freedom to produce a truly unique publication that will set the standard for editorial quality, integrity, and readability for the entire industry."

The new magazine format will allow each article to be presented in its entirety without being interrupted by distracting advertising material. It will also prevent articles from being broken by intervening editorial material resulting from a less-than-flexible layout required to accommodate the needs of advertisers.

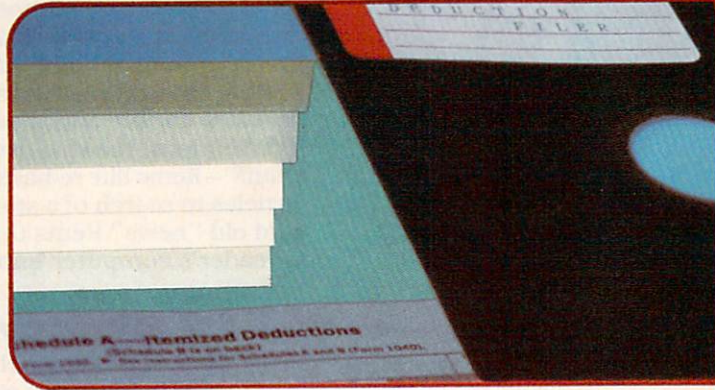
"We have thoroughly analyzed the financial considerations of this unprecedented move," Kaplan continued. "Our profitability projection has yielded very favorable results, and undoubtedly reflects the current magazine's uncommon strengths: its extremely high sell-through percentage on newsstands; its large, inexpensively acquired subscriber base; and its companion *ON DISK Revue* (tm), a spin-off software line recently introduced at the Summer Consumer Electronics Show and slated for retail distribution this fall."

Subscribers to *Home Computer Magazine* will also be kept abreast of additional product availability through a separately mailed, 32-page publication called *Home Computer Digest* (tm). This supplementary publication will be mailed approximately nine times per year and will contain mail-order advertising plus limited editorial material geared to readers who purchase products by mail.

























###

Gary M. Kaplan
Publisher
















HOME COMPUTERTM magazine







FEATURES

- 11 //c: The Core of a New Machine** 
Yet more fruit from the Apple tree. *by Peter Baum and the HCM Staff*
- 20 On the Home Court: Computer Sports Simulation**
Exploring the ties between computers and athletics. *by Wayne Koberstein*
- 35 Razzle Dazzle** 
Quick graphics magic for the 99/4A. *by W.K. Balthrop*
- 45 Simon Sez** 
Plug in 114 new BASIC commands to the C-64. *by W.K. Balthrop*
- 48 Tax Deduction Filer**     
10-40 good buddy! *by Roger Wood*
- 51 Kaleido Computer**     
A new medium for an old toy. *by Melody Covington and the HCM Staff*
- 69 Multiplan Medium (part VIII)**     
Applying "net present value" plus a review of Multiplan for the C-64. *by Patricia Swift*
- 73 Have No Fear: Assembly Language Won't Byte (part IV)** 
Assembly language on the 99/4A made easy—really! *by Peter Lottrup and the HCM Staff*
- 76 The RS-232 Interface: Your Link to the Periphery**     
The "standard" explained. *by Patricia Swift*
- 82 One for the Money, Two for the Slow—Adding a Second Drive to the PCjr** 
And they said it couldn't be done... *by David G. Brader*

GAMEWARE BUFFETTM

- 56 Boolean Brain**  
Wander down logic paths inside your computer. *by W. K. Balthrop*
- 58 Stadium Jumping**     
This game doesn't include leaping over sports facilities. *by Kent & Kathy Gemmel and the HCM Staff*
- 60 Market Madness**     
A vigorous exercise in investing. *by Brian Lee and the HCM Staff*
- 68 Elementary Addition and Subtraction**   
An educational program for the preschool crowd. *by Mark Dewese and the HCM Staff*

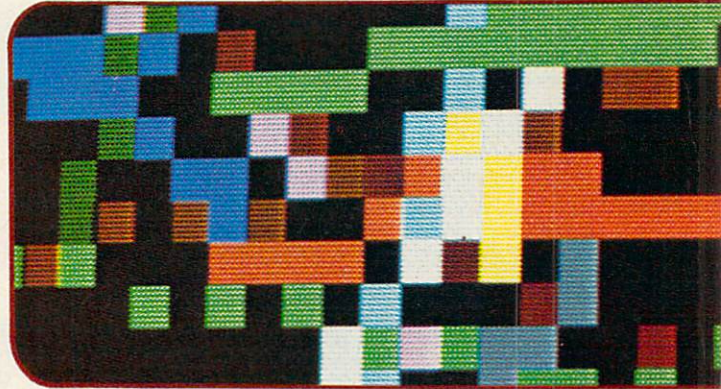
PRODUCT REVIEWS















- 23 International Soccer** 
Is it a game, or is it real? *A Review*
- 24 One-on-One**  
Sweat it out with the pros. *A Review*
- 25 Star League Baseball** 
One, two, three strikes you're out! *A Review*






CONTENTS











SEPTEMBER, 1984 VOLUME 4 NUMBER 4



- | | | |
|----|--|--|
| 26 | HCM Video Olympiad
Find out who wins the gold. |  
<i>A Review</i> |
| 28 | Country Club
Tee off in your living room. | 
<i>A Review</i> |
| 29 | Pole Position
Road racing action. |     
<i>A Review</i> |
| 30 | Bermuda Race
Yachting on the open sea. |  
<i>A Review</i> |
| 31 | Buck 'n' Kirk
Compare 2 starstruck games: Buck Rogers and Star Trek. | 
<i>A Review</i> |
| 32 | Jr. Addition:
A Review of the
Tecmar JrCaptain Peripheral
Juicing up Junior. | 
<i>A Review</i> |
| 36 | SST BASIC Compiler System
Give your TI-99/4A supersonic speed. | 
<i>A Review</i> |
| 40 | Two for TI:
A Review of the
CompuAdd Dual Disk-Drive Package
Double your storage, double your fun. | 
<i>A Review</i> |

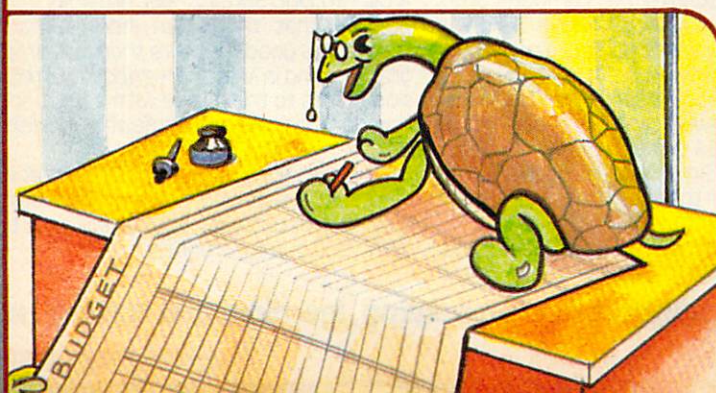
- | | | |
|----|--|--|
| 42 | Mousing Around on the Apple II
Mac-magic for the IIe. | 
<i>A Review</i> |
| 44 | A Shortcut to 99/4A Printing:
A Review of the Axiom GP-100 TI II
Printing without an expansion system. | 
<i>A Review</i> |
| 70 | Multiplan for the C-64
How this package compares to other machine versions | 
<i>A Review</i> |

LOGO TIMES™

- | | | |
|----|--|---|
| 87 | LOGO Spreadsheet
If you thought LOGO was just for graphics, read on. |     
<i>by Rich Haller and the HCM Staff</i> |
| 90 | Missionary Impossible
Don't rock the boat with these cannibals. |     
<i>by Roger Kirchner and the HCM Staff</i> |

DEPARTMENTS

- | | | | |
|----|------------------------------|-----------------------------------|---------------------------------|
| 3 | On Screen | 93 | Program Listing Contents |
| 6 | Inside/Outside HCM | 130 | DeBugs on Display* |
| 8 | Letters to the Editor | Home Computer Tech Notes: | |
| 19 | HCM Review Criteria | 78 | Apple |
| 38 | Group Grapevine | 79 | TI |
| 46 | Industry Watch | 80 | IBM |
| 47 | Any Questions? | 81 | Commodore |
| 63 | HCM Product News | | |
| 92 | Program Typing Guide | *See also "Letters to the Editor" | |



HOME COMPUTER magazine

Home Computer Magazine (ISSN 0747-055X) is published monthly by Emerald Valley Publishing Co., P.O. Box 5537, Eugene, OR 97405. 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 \$7 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 \$3.00, and \$3.75 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 5537, Eugene, OR 97405. 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 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 September 1984 issue and the issue as a collective work is Copyright © 1984 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.

Publisher/Editor-in-Chief Gary M. Kaplan
Executive Editor David G. Brader
Managing Editor Walter Hego
Associate Editor Wayne Koberstein
Sr. Technical Editors William K. Balthrop
Roger Wood

Technical Editors
D. Donaldson, Tom Green, G.R. Michaels,
Steven P. Nelson, Patricia Swift

User Group Editor Judy Campbell
Assistant Editor Dana M. Campbell

Program Translators
Hendrik Broekhoff, Stephen A. Cordon,
Ann Dahm, Richard Haller, Scott Kindt,
Randy Thompson, Rebecca Van Dalsem

Contributing Editors
Michael Brownsworth, William M. Goodman,
Henry Gorman, Jr., Richard Haller, S. T. Holl,
Roger Kirchner

Asst. to the Publisher Rhea J. Grundy
Production Manager Norman Winney, Jr.

Creative Director Gei-Lei Gom
Typesetting Curtis Byrd

Photography
Nelson Stevens, K.D. Wainsworth

Production Assistant Rachel Knight
Customer Relations Tel. (503) 341-1029
Sharon Hinshaw

Dealer Sales & Distribution Tel. (503) 341-1032
Wendell Anderson, Michael Flagg,
Paula Holland

Main Switchboard Tel. (503) 485-8796

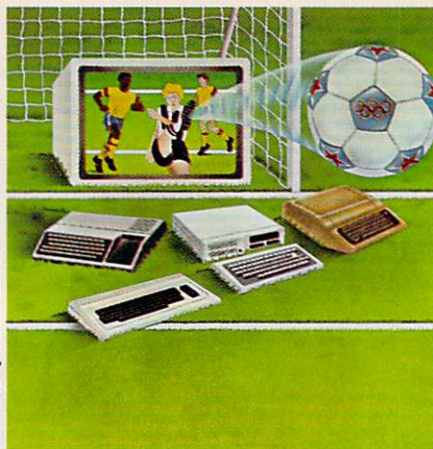


Illustration by Sharon Harker

Outside HCM

Kick-off! That soccer ball is bouncing right into your living room with a new generation of sports simulation games. Today, these games are sporting increased sophistication, inspired by the XXIII Olympiad and the proliferation of computer-graphics in athletic training. Now, with the help of Home Computer Magazine, you can view and participate in this wide world of computer sports—complete with the thrill of a victory RUN, but without the agony of "de feet."

Inside HCM

Wait a minute, ole sport! Summer fun doesn't end when the days get shorter—and neither does the fun of summer sports. You may have to go indoors when the ground gets soggy or the wind blows cool, but—you can still enjoy the thrill of swinging a bat, running the 440, or kicking a goal if you're fortunate enough to own a home computer.

This month, Home Computer Magazine features a special review of the new sports simulation games. We kick off this section with *On the Home Court*, a blimp's eye view of this exciting field of gaming software, including an interview with athlete/programmer, Eric Hammond. We hope all the fans—and the budding software artists—out there will find this article and the accompanying reviews stimulating and even inspiring.

But first, take a slice at some fresh produce as we peer into an Apple IIc: *The Core of a New Machine*. Like our report on the IBM PCjr published three issues ago, this article—with extensive documentation and photos—gets right to the heart of the compatibility issue.

Now and then, it helps to have a little advice—in the form of our in-depth tutorials—to guide you along the road to computer glory. This month, HCM has several articles to instruct and encourage the user-in-training. For example, learn about net present value in Part VIII of *Multiplan Medium*, which exercises another aspect of a very useful electronic worksheet. (*Multiplan* was recently issued for the C-64, so Commodore owners will notice a special review of this program accompanying the tutorial.)

BASIC 99/4A programmers who are still intimidated by talking to their computer on its own level are advised to *Have No Fear: Assembly Language Won't Byte (Part IV)*. And for both beginners and more advanced 99/4A users, we pause for a little *Razzle Dazzle*, a short and sweet graphics treat. A similar treat awaits C-64 owners with *Simon Sez*, a mini-tutorial in Simon's BASIC. Both of these articles premier this month as new, regular features in HCM.

For IBM PCjr owners, we have a real granddaddy of a "how-to" article—*One For the*

Money... Two For the Slow: Adding a Second Disk Drive to PCjr. If you're tired of switching disks in and out of Junior, you may well be attracted by this relatively inexpensive alternative. And any computer user can discover the principles of interfacing—or how to hook all those computer gizmos together—with *The RS-232 Interface: Your Link to the Periphery*.

You can also learn a lot from our incisive product reviews, as we examine: *Tecmar Jr. Captain*, a memory-expander—and more—for the PCjr; the *SST Compiler*, taking off from BASIC to the speed of assembly language; dual drives for the TI-99/4A, with our own photo-guide to installation; *AppleMouse II (with MousePaint)* for the Apple IIe; and a TI printer by Axiom that doesn't need an expansion box.

With our HCM key-in programs, you can step up to home plate—your computer—and score a RUN every time. From its center-field position in this issue, our BASIC software section covers all the bases. Lead off on tax-time now with *Tax Deduction Filer*, and learn the art of creating colorful graphics patterns with *Kaleido Computer*, a combined program/tutorial. C-64 and TI-99/4A owners can try an adventure in computer education inside the *Boolean Brain*, while Apple and IBM users—who played *Boolean Brain* last month—receive our practical learning aid, *Elementary Addition and Subtraction*. And everyone—on all machines we cover—can practice *Stadium Jumping* (with horses, not over stadiums), or experience a taste of *Market Madness* (stocks, not groceries).

Two LOGO programs complete our line-up: A useful *LOGO Spreadsheet* provides proof positive that this often-underestimated language is good for more than just simple graphics; and in *Missionary Impossible*, recursion comes to the rescue with a LOGO solution to a logical—and somewhat ticklish—puzzle.

So don't worry about those shortening summer days, ole sport. When the mits and gloves are all put away, you can still go to bat with your home computer.

Until next month, have fun reading, learning, and RUNNING HCM



*The Perfect Gift For
Any Occasion!*

MAKE SOMEONE HAPPY!

GIVE A GIFT SUBSCRIPTION TO

HOME COMPUTER[™] magazine

As a *bonus* gift we'll send **YOU**
or the **GIFT RECIPIENT**

2 issues of **ON DISK[™]** or **ON TAPE[™]**

ABSOLUTELY FREE!

Comments From Our Happy Readers

"You broke new ground when you introduced your unique typesetting style for your program listings; well, you have done it again by separating all of the listings in the magazine into one section in the center of the magazine. I call that genuine brilliance!
Warren Agee, Livonia, MI

"Well it happened again. Your magazine arrived in the mail, and I'm completely delighted with it."
Chris L. Chaffin, Omaha, NE

"I have subscribed to your magazine since its inception. I must say it has been most informative and has provided me with answers to many of my questions. Your feel for what the public wants is uncanny!"
Larry A. Hamel, Millington, TN

"I just received your August issue. I ordered a 3 year subscription exactly 1 year ago, and I have seen it grow in size and quality. This latest issue, with the separate section of program listings, reaffirms my wise subscription investment."
Mike Oliver, Clarendon Hills, IL

"When I saw the new version of your magazine I was elated! Naturally I subscribed.
Doug Barker, Exeter, CA

"I was a former subscriber to the 99'er Home Computer Magazine and I thought it was great. Then when I got the first issue of the Home Computer Magazine, I was twice as happy. It was a lot of information and great articles. Keep up the good work!"
Jenny Bures, Thousand Oaks, CA

"You have done a superb job of reaching other types of Home Computer enthusiasts and expanding your clientele while not depriving us 99'ers or leaving us by the wayside. The quality of the magazine is unsurpassed by any other, and I have looked at several different magazines! Hats off to you folks for your originality and continued endeavor to reach perfection."
John R. Stewart, Tucson, AZ

"...I am extremely pleased that a magazine such as Home Computer Magazine is around. I find the magazine extremely well written and of invaluable aid. . .keep up the good work with the magazine."
James L. Grigsby, Richmond, KY

Thanks to your thoughtfulness. . .your friends, family, and associates can enjoy a gift that keeps on giving all year through! They'll enjoy

12 BIG issues of HCM

delivered right to their door each and every month. . .

And you or the gift recipient will receive

2 months of
our magazine
program service

ON TAPE[™] or ON DISK[™]
ABSOLUTELY FREE!!

To Order A Gift-Subscription, Use Bind-In Card in Center of Magazine.



Dozens of top quality key-in-and-RUN programs for Apple, Commodore, IBM, and Texas Instruments home computers appear in each issue.



FREE software—ON TAPE[™] or ON DISK[™]—the same high-quality programs published monthly in the magazine. This cassette tape or floppy disk program service—normally a \$3.95 per month extra cost—is the convenient, accurate and affordable way to save hundreds of typing hours.

Letters to the Editor

TI/Okidata Link

Dear Sir:

I experienced problems in connecting an Okidata printer to the TI-99/4A computer just as did Mr. Wolly Barabash.

After some trial and error, I found that the printer would work with only a minor change. The connections that I used are:

TEXAS INSTRUMENTS		OKIDATA	
Term.	Description	Term.	Description
1	HANDSHAKE OUT	1	DATA STROBE
2-9	DATA	2-9	DATA
10	HANDSHAKE IN	11	BUSY
11	LOGIC GROUND	10-30	DATA RETURN

These are the standard connections that you would expect for connecting a Centronics printer to the TI computer. But to make it work, I connected a 270k Ohm resistor in series with the HANDSHAKE IN - BUSY wire. I installed the resistor inside the Centronics connector at the Okidata end of the cable.

I asked both TI and Okidata if this connection was acceptable, but got no response to that particular question. The printer has been working fine for over a year now.

Thomas Nisius
Westlake, OH 44145

For those users that are do-it-yourselfers, you may wish to give this a try. For those of you who would rather play it safe, read the following two letters.

Dear Sir:

I am writing to congratulate you on your superb magazine and its great coverage of the great TI-99/4A.

This concerns a letter from Wolly Barabash that says he has a problem finding an interface for his Okidata printer. I do not have an Okidata printer, but I have seen the needed cable. He can order it from: Tenex, P. O. Box 6578, South Bend, IN 46660. The part number is 10036, parallel cable-Okidata.

Again, thanks for a great magazine.

Brian Neidig
Taylor, TX 76574

Dear Sir:

In your August 1984 issue a reader expressed his dismay to hooking up his Okidata to a TI. There is a company called Innovative Electronics and Computing, 4150 Fox Street, A-5, Denver, CO 80216. I am presently using an OKI92 with their cable with no problems at all. The cost is about \$25, the stock number is (CBL-1146). I hope this information will help those who are interested in hooking up an OKI to a TI.

Joe Rodomista
Selden, NY 11784

Apple 3d Iie Question

Dear Sir:

I'm frustrated! Being new at computing yet fairly intelligent and resourceful, I was intrigued by the "Apple Graphics in Three Dimensions" of Michael Brownsworth's two-part article. Not trusting myself to type in the whole program, I sent for the two disks that you offered. But, try as I might, I have not been able to load a single image onto my screen.

I have a 64K Apple Iie with two disk drives and a monochrome monitor. Here are the problems I

have encountered with your set of programs:

1) Every time I choose one of the three display objects (cube, pyramid, house) from the menu, the screen just goes blank and nothing else will work.

2) When I typed RUN EDITOR 3-D, I got an error message "undefined statement error in 3620." Obviously I fail to understand something very basic and crucial about this program. Is every necessary file provided on the disk that comes with Vol. 4, No. 2 or do I have to go back to the disk for No. 1 and copy something?

The other programs on the disks run okay.

Barbara Matthies
Ames, Iowa 50010

Everything you need is on the disk for Vol. 4, No. 2—it's in the form of a turn-key menu-driven system as described on page 40 in Vol. 4, No. 2 of HCM. The disk does not, however, include the same HELLO program described in the article—if it did, the disk would always start running the Applesoft 3-D system upon booting up. To use any of the 3-D programs, Barbara, all you have to do after booting the disk is type: EXEC LOMEM.EXEC. Then you select what you want to do from the menu that appears and the appropriate program is RUN for you. There is a detailed discussion in the Apple Tech Note on page 99 of the same issue covering some of the aspects of this technique.

So, You Want Less Advertising

Dear Sir:

How about more articles for the PC and Commodore 64 and less advertising?

Compared to the program listings in Compute's Gazette, your listings are not easy to read. The listings are just too small to look at without getting eyestrain. On the positive side, your articles are very well-written and understandable.

One last thought. How about publishing some programs that use Simon's BASIC?

Ira Rubin
New York, NY 10023

Well, I hope we have anticipated your request for less advertising in this issue. As you can see we have eliminated all outside advertising from the magazine. We have anticipated another one of your wishes. Take a look in the Table of Contents and find the special feature entitled, "Simon Sez" to start learning about using Simon's BASIC. As for the matter of listing size, readers tell us that we more than compensate for it with our typeset clarity and quantity. We do also offer readers with poorer eyesight (as well as those who don't have the time or desire to type) the option of very inexpensive prerecorded cassette tapes or diskettes (see back cover).

Easy Script Price Was Wrong

Dear Sir:

I would like to comment on two of the product reviews in the last issue of Home Computer Magazine. The first is the review of Easy Script for the Commodore 64. This was a very good review except for the price that was listed. You gave the price as \$99.95. I believe this is the wrong price. I have been using this program and only paid \$39 for it. I know this is not list price, but I am sure that the price you gave was too high. This has to be the best word processor that you can buy for the price. As stated in the review, the manual is really the only fault with the program. I would like to see listed

in your reviews the language in which the programs are written (machine language for Easy Script).

The second review is for the Home Accountant. I have been using this program on the Commodore 64 since the first of the year. I find several things wrong with this review. I cannot say anything except for the version for the C-64. I have had considerable trouble getting a version that works as it should. After much letter-writing I received another disk and now have a disk where everything works. This is indeed a powerful program as stated in the review, but it is very slow as it is written in BASIC. I believe this should have been in the article. The graph that is shown in the article is nothing like the ones in the C-64 version. The colors used in the C-64 version are very dull and drab. With the colors available, better colors should have been used. I don't know what version you tested, but it was not for the Commodore 64. I will replace this program with a faster one when I can find one.

Now, I would like to comment on the magazine itself. I have always liked the format of the magazine. I find the program listings the easiest of any magazine to key-in.

Jim Gibson
St. Joseph, MO 64504

Easy Script on the Commodore 64, according to the manufacturer, has a suggested retail price of \$54.95. So, Jim, you are right.

It sounds like you had a great deal of bad luck getting a decent copy of the Home Accountant from Continental Software. The C-64 version that we received worked without any of the problems that you describe, other than the speed of the program, which we did mention in the last part of the article. Being written in BASIC, as you state, does make the program slow. In addition, the 1541 Commodore disk adds its own factor of slowness to the operation. The photograph in the article was taken from the Apple IIe version, and not the Commodore 64.

TI Spelling Checker

Dear Sir:

In the August 1984 issue of HCM, William Koseluk asked if anyone knew of a spelling-checker program for TI-Writer. Tom Kirk has developed a program to test the spelling of words; it operates out of the TI-Writer utility option. It comes on two disks and includes a 20,000-word dictionary. In addition, users may add their own dictionaries to the system. However, the program has not been released pending the completion of the instruction manual. Interested users may write for more information to: Tom Kirk, 2606 Ponderosa Drive, Omaha, NE 68123.

Loring Rose
Pantego, NC 27860

Loring, your letter arrived the same day that a package arrived from Mr. Kirk. He has sent us a copy for review of his 99/4 Auto Spell-Check program (which is available from his company Dragonslayer American Software Company, reachable after 5 p.m. Nebraska time at (402) 291-8323). Look for our review of this package in the near future.

More Reusable Peripherals for TI

Dear Sir:

By now I am certain you have received many letters stating the very same thing, but even at the risk

of not being original, let me say it again. What a pleasant surprise to see your magazine on my neighborhood grocer's magazine shelf. It was truly like meeting a friend one thought to have passed away. No offense, but I had visions of your work, if not you personally, being in the literary "Valhalla!" Welcome to the living.

In a more recent issue, John Paulson's letter in Vol. 4, No. 2 regarding peripherals that can be used by both the TI and future state-of-the-art equipment: Taxan's RGB 210 can also handle composite color output as well as RGB. I am currently using such a monitor which sells for under \$250.

Harry Plettner
Schaumburg, IL

Thanks for the tip on the Taxan RGB 210 monitor, Harry. We have also found that the new Sears Performance television is ideal for home computer use. This Sears unit retails for \$349 and it includes both RGB and composite inputs, plus it is a color television to boot. We have been using one in the Editorial Department for several months now and are very pleased with it.

LAST MINUTE DeBUG

In the August Issue of *Home Computer Magazine*, after type-in verification, our paste-up crew inadvertently sliced part of line 780 from the IBM PC and PCjr version of *Spider Graphics*. Line 780 should read:

```
780 IF B$ > "9" THEN COL=ASC(B$)-55 ELSE  
COL=ASC(B$)-48
```

Frozen TI Computer in Argentina

Dear Sir:

I have recently taken out a subscription to your magazine and must praise you for its content—first class! I like the ready-to-run programs as I enjoy following the thought and sometimes changing the steps.

Could you please help me on a technical point on my TI-99/4A? Twice now, when making up a program, I have pressed "Enter" at the end of a line and nothing has happened. The screen freezes, the flashing dot disappears, and nothing can be done to move from that position, no matter what key is pressed. The only way out appears to be to switch off, thereby losing all the program typed in, and starting again. On one occasion, I had typed in one of your programs and this happened on the last line! Why does this happen and is there anything that can be done to save the situation without switching off? A friend of mine tells me he has had the same experience.

I shall be very grateful if you would give me some advice on this situation by either writing to me by air mail to the above address or by publishing this letter with your answer in your magazine (which I receive by air mail).

M.K. Atkinson
Buenos Aires, Argentina

Several things can cause a home computer to "freeze" as you described. First is a static electricity discharge from the operator shuffling around in the chair on a carpet. Second, is poor AC power. Third, and least likely, is the actual malfunction of the computer itself. Two inexpensive precautions that you can take immediately are (1) make sure the computer is in a static-free environment and (2) SAVE whatever you're working on every few lines so that if the computer does hang-up you will not lose all

of your work. This is a good habit to get into and we recommend it for anyone using any computer.

Apple DOS or DOS Not

Dear Sir:

I was a subscriber of 99'er Magazine and found it extremely helpful in my education on the TI-99/4A. However, when TI discontinued the machine, I felt the need to move to a computer which was more firmly implanted in the market. So, I moved to an Apple IIe as it seems many of your other readers did.

In the past months I have learned much about Apple and I am a bit disturbed by some of your program listings for the Apple II series. I feel that one of the main purposes of copying a program out of a magazine is to learn the programming techniques of other programmers. Because of this, the listed programs should illustrate the suggested form of the manufacturer (unless it is demonstrating some unique function or capability of the machine). As an example, I cite your recent spreadsheet program (which is quite good), Snap-Calc (HCM August 1984). Under the Apple II series listing, the printer

we recommend something very similar to what you suggest for the program to run properly when loaded under ProDOS.

Conversation About JoyTalk

Dear Sir:

Can you persuade your technical section to come up with lots more projects like "Joytalk" which was very successful? If one was to follow the ads, it would take \$2000 to kit up for the equivalent of Joytalk in Ireland, with PEB, 32K memory, and RS232 before the alphabet could appear on a printer. No hardware or software in Ireland.

Andy MacMahon
County Cork, Ireland

Andy, we've been looking for projects to put in the magazine that are similar to Joytalk. We would welcome any assistance from readers who have built neat little "black boxes" to hook up to their home computers.

TI/C-64 Statistics Software Sought

Dear Sir:

I was initially a subscriber to your magazine when it was devoted entirely to the 99/4 series of computers and was quite satisfied with it. Following the departure of Texas Instruments from the home computer market, I purchased a Commodore 64 with disk drive and monitor in order to ensure that I would be able to take advantage of newer software which might not be produced for the discontinued 99/4A. Your decision to expand the magazine to cover four computers, including the Commodore 64, was greatly appreciated by this subscriber.

I request your help in locating commercial statistics software that I can use on either of these machines for data reduction obtained from a research project to be conducted in my office at this VA. The software which is sold by the manufacturers is not sufficient for my needs as I require programs which would do multiple linear regression, analysis of variance as well as the more basic statistical functions. Any referrals to possible software vendors with these sorts of programs would be appreciated.

David R. Moody
Salem, VA 24153

Another challenge. Do any of you folks out there know of software that will help David out? If so, please drop us a line.

No DMA on PCjr

Dear Sir:

I am directing this letter to Gary Kaplan and/or William K. Balthrop, authors of an excellent article about the PCjr in your Volume 4, No. 1 issue, entitled "A Detailed Look Inside the Peanut's Shell." I am interested in knowing if DMA (Direct Memory Access) is available from any third-party vendors you may know of. I am planning to order Tecmar's Jr. Captain and they do not include it. I am writing IBM additionally as I understand an upgrade version of the PCjr is due. It may or may not have DMA.

Colin Smith
Lewiston, NY 14092

We believe at this time, the only way to get around the DMA problem would be some form of hardware and software modification inside the PCjr itself.

Continued

Letters

to the Editor

The new version of PCjr that was just announced still does not include DMA (see the IBM new product announcement elsewhere in this issue). We are extremely interested in finding a method to circumvent this DMA problem, as I'm sure you are. If other readers have already met this challenge of the jr., please let us know.

TI Bulletin Boards Abound

Dear Sir:

In response to two of your past letters regarding telecomputing, I submit a list of TIBB(tm) bulletin board systems that are currently operating in the USA and Canada. This program, part X BASIC and part machine language, is for sale by Mr. Ralph Fowler of Kennesaw, Georgia and is also marketed by CR Distributing, who advertised in your August issue, page 153. Mr. Fowler can also be contacted via his own BBS at (404) 425-5254.

Additionally, Mr. John Clulow is presently producing a Bulletin Board System program, which he apparently will distribute to users groups when completed.

An excellent source of BBS listings is available on the ONLINE BBS at (913) 649-1207. Also, the POST section of THE SOURCE frequently advertises new boards in operation. I assume COMPUSERVE's post section would contain listings as well.

T.L. Atkinson
Dartmouth, Nova Scotia

Thanks for the information, Terry. Unfortunately, we don't have space to print the entire list of bulletin boards here. We can include yours however, for our readers. For folks in and around Nova Scotia, the number is (902) 434-3121 to reach Terry's TIBBS.

Dear Sir:

It is very hard for me to believe a magazine of your caliber would just now be hearing about 99/4A Bulletin Board Systems when for quite a few years, Ralph Fowler has been marketing his TIBBS system, and also the CALTEX-99 systems have been on-line for two years.

There are six CALTEX systems and approximately 40+ TIBBS systems on-line nationwide, TIBBS being the system I am running at (415) 355-3092.

Our color TI-Lines TIBBS runs 24 hours and provides continued support for the TI-99/4A and also runs on a 99/4A. The system contains a listing of all other TI boards nationwide and other items of interest.

Mark S. Wong
San Mateo, CA 94404

In the big, wide, wonderful world of home computing there are probably several things which we have overlooked—and we appreciate it when anyone draws them to our attention. Thank you very much, Mark.

Zork on IIc

Dear Sir:

I am thinking of buying the new Apple IIc and I have a few questions about it.

In your magazine you have programs for the Apple II, II+, and IIc. Will these programs work on the IIc? Will you cover the IIc? Also, in Volume 4, No. 2 you reviewed Zork by Infocom. Zork can

be run on the Apple IIe, but can it be run on the IIc? Could you please review the Apple IIc?

Like so many others, I would like to thank you very much for such a fine magazine.

Danny Newton
Phoenix, AZ

Danny, as you can see from our previous issue and this one, we definitely are covering the Apple IIc and the Apple programs that we put in the magazine are tested on the IIc. If there are any differences at all in the operation, they will be so noted in the article. Regarding Infocom's Zork I which we reviewed for the Apple IIe, we have tested it on the Apple IIc and it appears to work on this machine as well.

Eliminate Accidental QUITTING

Dear Sir:

I wrote to tell you of the great job you are doing with your magazine. I enjoy typing in your programs.

I also wrote to inform TI-99/4A users who are bothered when they accidentally push the quit command (FCTN +) when they meant to push the plus sign. They can disable the quit command if they have memory expansion and TI Extended BASIC. They can remove it by typing in:

100 CALL INIT
110 CALL LOAD(-31806,16)

Then press RUN and your quit troubles are over.

Jeff Markey
Fort Dodge, IA 50501

Thanks, Jeff, for that tip. You may have just saved the sanity of thousands of Extended BASIC programmers.

Using TI Printer With C-64

Dear Sir:

Since you cover both the TI-99/4A and the Commodore 64 computers, may I appeal to you for advice that no computer store has been able to give?

I have a Commodore 64 computer and a TI-99/4A printer. How can I connect the printer to the computer? As far as I know, the printer does not have a Centronics parallel capability.

It would be nice to be able to use the user's port on the C-64 for a modem and at the same time have the printer hooked up. Is this possible?

Your advice would certainly be welcome.

Edmond Reynolds
Somerset, CA 95684

The Texas Instruments 99/4 Impact Printer has both a serial input and a Centronics parallel input. To use the parallel input with your Commodore 64 (or VIC-20), will require the purchase of a special interface adaptor such as the Cardco Centronics parallel adaptor. This adaptor sells for \$99.95 and is available from Cardco, Inc. 300 S. Topeka, Wichita, KS 67202. Although it is difficult to get it set up initially, the results are worth the effort. We recommend this unit.

Count-Sil vs Snap Calc

Dear Sir:

This letter regards the review of our product Count-Sil in your August issue of Home Computer Magazine.

We are very disappointed not only in the review but the fact it was placed in an issue which contained a six-page write-up of Snap Calc, a free spreadsheet from Home Computer Magazine. The write-up on Snap Calc professed all its great features, its ease of use, etc. . . . The review on Count-Sil did not point out the same features plus the many additional benefits we offer. In fact, as a reader of this issue, I would not buy my product when I can either type in Snap Calc or order a copy from Home Computer.

Since we knew a review was going to be in this issue we put a two-color half-page ad in the issue. When I saw the new issue, I was shocked that I would be in competition with Home Computer Magazine and saw that I had just invested in an ad that (based on the issue) I will be lucky to get back 1/10 of my investment.

I wish your magazine had kept us better informed and had recognized the effect of putting the Snap Calc article in with our review.

Sandy Foote, President
Systems Interface, Ltd.
Nepean, Ontario, Canada K2G 3J3

The August issue's theme was productivity. We consider Snap Calc to be productivity software, so it was featured in that particular issue. Likewise, we consider Count-Sil to be productivity software, so it was reviewed in that particular issue. The coverage of Snap Calc was extensive because the entire product is in the magazine. Snap Calc is fully exposed to our readers, and they are free to evaluate the software without risking any money. Our readers can only evaluate Count-Sil by purchasing the product or trusting our review. We feel the Count-Sil review was a fair assessment of your product.

Our primary business interest has always been to secure loyal readers for Home Computer Magazine. Over the years, we've discovered that standard magazine business practices—including the catering to the needs (and sometimes, whims) of advertisers such as yourself—can actually hinder editorial efforts to serve the best interests of a magazine's readers.

Therefore, starting with this issue you're now reading, we have eliminated this potential conflict of interest by not accepting any outside advertising (see editorial by publisher on page 3). It probably won't make you feel any better, Sandy, to hear all this, but we on the staff of this magazine feel very proud to be pioneering this new editorial freedom and reader-service orientation. HCM

Special Announcement:

Home Computer Magazine is looking for "One Liner's". If you have written a 1-line program in any language that is available on the computers we cover, send it in addressed to Letters to the Editor. It may win a prize and be printed here!

IIC:

The Core of a New Machine

by
Peter Baum and the HCM Staff

Peter Baum is an Apple II hardware engineer who makes his home in Apple's Cupertino facility. (We understand that he also maintains an apartment for occasional use.) Having worked with the Apple II family since "ancient days," Peter is considered one of the best sources of Apple knowledge.



The Apple IIC: The Portable Apple IIe

With the IIC, Apple has introduced the world to the first lightweight, portable computer with a large, ready-made software base. Most "portable" computers sold today are 30-pound boxes with a handle, or lap-computers with very little software available. The Apple IIC, priced at \$1295, is designed to be software compatible with a majority of the estimated 10,000 programs available for the Apple IIe. This 7.5 pound computer fits into most briefcases due to its small size (11.5 x 12 x 2 1/4 inches). Because of its size and compatibility, many people think of the Apple IIC as the Apple IIe's little brother. But as you read on, you may find this label to be misleading.

The IIC is just as powerful as the IIe, and includes many convenient features that are not built-into the IIe—such as two serial ports, a mouse/game port, a built-in disk drive, and 80-column capability. These features can be added to the IIe, but require purchase of up to five peripheral plug-in cards. For the typical consumer, especially the novice computer user, Apple has packed the IIC with all the most needed features.

To stay consistent with the IIC theme of easy use, Apple has placed icons or pictures above the connectors on the back panel. Peripheral makers can manufacture cables with matching icons, enabling quick and easy cable connection for the user. In this regard, the IIC represents another example of progress toward user-friendliness by the computer industry.

Everything Included

While talking to first-time computer buyers, Apple discovered that the initial expe-

rience at home was crucial to keeping the buyer satisfied. Therefore, Apple has attempted to package the IIC in such a way that the user's first session is a positive experience and not frustrating. The package includes everything needed to have the buyer operating the computer within the first hour. To achieve this, all the cables, connectors, and an RF modulator (to hook the computer up to a television) are now packaged with this new breed of Apple.

Included with the IIC are six tutorial disks which introduce the buyer to the computer. These disks contain introductions to the keyboard, applications (such as word-processors and spreadsheets), and programming languages (such as BASIC and Logo). These "interactive tutorials" use the computer along with the simplified manual to show how things work, instead of just a large boring manual like those included with some computer systems. [The only thing noticeably "missing" from the IIC package is a manual for using Applesoft BASIC. We just cannot accept the rationale for Apple unbundling this "basic" and charging extra for it.—Ed.]



The Apple IIC shown with the new Apple Scribe Printer, soft carrying case, IIC monitor, AppleMouse II, IIC external disk drive, modem, and joystick.

Apple IIe to Apple IIc Software Compatibility Sampler

Home Education

Facemaker (Spinnaker)	Compatible
MasterType (Scarborough)	New Version for IIc
Music Construction Set (Electronic Arts)	Update costs \$7.50
Spellcopter (DesignWare)	Compatible
The Most Amazing Thing (Spinnaker)	Compatible

Home Management/Productivity

Bank Street Writer (Broderbund)	Update costs \$20
Home Accountant (Continental)	Compatible
HomeWord (Sierra On-Line)	Compatible
PFS:File, PFS:Graph, PFS:Report, PFS:Write (Software Publishing)	Update costs \$35/module

Home Entertainment

Lode Runner (Broderbund)	Update costs \$10
One-on-One (Electronic Arts)	Compatible with 1 joystick
Pinball Construction Set (Electronic Arts)	Update costs \$7.50
Zaxxon (Datasoft)	Compatible
Zork I (Infocom)	Compatible

Portability

Right now the IIc is more of a transportable computer than a true portable. Apple has announced that a flat panel display for it will be available in the fall. This liquid crystal display, along with a battery pack from a third party vendor, will un-tether the computer and allow it to roam anywhere. [One company has announced a rechargeable battery power system in a backpack, which also includes room for the IIc and flat panel. The CARI from the DiscWasher Company of Columbia, MO. has a suggested retail price of \$249—Ed.]

The IIc was designed with a two-part power supply. The external power pack that comes with the computer converts the 110 volts from the AC line into 12 volts DC for the computer's use. Inside the IIc is the second part of the power supply unit which converts the single DC input voltage to 3 voltages, +5,—12 and +12 DC volts, all of which are regulated. This internal power supply portion has been designed to accept an input voltage between +7 and +24 DC volts. This means that the computer will also run using anything from a car battery to a motor-home's generator for a power source. (We are sure some enterprising company will soon be selling a power cord for the IIc which plugs into a car's cigarette lighter.)

Software Compatibility

Apple has estimated that 90% of the approximately 10,000 programs designed for the Apple IIe will run on the IIc without modification. [This figure of 10,000 includes a very large number of programs with questionable value. We have attempted to select a few of the more popular programs that run on the IIe for inclusion in the Software Compatibility Chart shown here.—Ed.] Apple established the 90% figure by coordinating testing of over 700 different programs on the IIc. This final result was published as the IIc Compatibility List and has been sent to Apple dealers.

Apple IIc Specifications

Size:	Apple IIc is 12-1/4" deep, 11-3/8" wide, and 2-1/2" high. The DC power supply module is 5-1/4" deep, 2-7/8" wide, and 2-1/2" high.
Weight:	The Apple IIc weighs 7.6 pounds and its power supply module weighs 3.5 pounds.
Power Source:	The machine comes with a 15VDC power supply module that sets on the table. This "transformer-like" unit plugs into 110 volt AC wall current and feeds the DC power to the IIc power input jack. Any other DC power pack that can meet the input requirements may be used (car battery, portable rechargeable battery-pack).
Sound:	As with the rest of the Apple II family, sound is generated by user-supplied software that "plucks" the built-in speaker. Two differences with the IIc—there is a volume control and an earphone jack.
Keyboard:	Standard keyboard layout with full-travel keys. Includes a switch for changing the keyboard configuration from "QWERTY" to Dvorak.
Video:	Supports all Apple II family display modes. (See Video Display Modes chart.) Has a switch for changing from 40 column to 80 column display.
Memory:	128K bytes of RAM and 16K bytes of ROM containing AppleSoft BASIC and I/O support software.
Processor:	65C02 8-bit CMOS Microprocessor. (Low-power version 6502.)
Disk Drive:	Single drive built into case. Accessed from right side.
Operating System:	Comes with Apple ProDOS. Will work with DOS 3.3.
Expansion Ports:	All expansion is through external attachments.
Serial Ports:	Two RS232-compatible ports are available.
Disk Drive Port:	External floppy disk drives and other peripherals may be added via this port.
Mouse/Game Port:	A single joystick or the AppleMouse II can be connected to this port.
Video Ports:	A video output port to support an RF modulator for TV hookup (Included with computer), an RGB adapter, or a flat panel 80x24 LCD display. This port can also be used with a light pen. An RCA jack output port is also included for NTSC composite video monitors.

Stickers which designate a program as *IIC* compatible have been made available to software developers, so that they can indicate on their packages which programs will work on the *IIC*.

Key(board) Features

The Apple *IIC* keyboard layout is virtually identical to the *IIE*'s. It is a standard, full-size keyboard with 62 keys that can generate all 128 ASCII character codes (which include upper-and lower-case characters). The "home-row" keys D and K have small bumps to aid touch-typists. The keys themselves have been redesigned so that they are flatter, and they offer a tactile and auditory response different from the ones used on the *IIE* because of a special absorbent pad located under the keys themselves.

The [RESET] key, above the keyboard, is located on the far right on the *IIE* and on the far left on the *IIC*. Next to the [RESET] key are two new keys. One is for switching video output between 40-and 80-column text displays. This 40/80 switch was added to the computer so that a user could switch a program to 40 columns when using a low-resolution monitor such as a television, and back to 80-columns when using a high-resolution monitor.

The other switch is for changing keyboard layouts. This switch was originally designed to switch between languages in foreign countries. On international models of the *IIC*, it changes both the keyboard layout and the character set between the common standard for that part of the world and the U.S. standard. This is done because a majority of the software for the Apple is written in English. In the U.S. version of the *IIC*, the

switch is used for changing the layout from the standard QWERTY type (Sholes) to the Dvorak layout. The Dvorak layout is generally considered faster and easier to type on because the most commonly used keys are placed on the home row. (The original QWERTY layout was designed to slow down typists on the old-fashioned typewriters, so that the mechanical keys wouldn't get tangled!)



The top of the *IIC*. Note the reset, 40/80 (column), and keyboard switches on the top left, and the disk use and power indicators on the top right. When the keyboard switch is activated, the Dvorak layout (shown below) is selected.

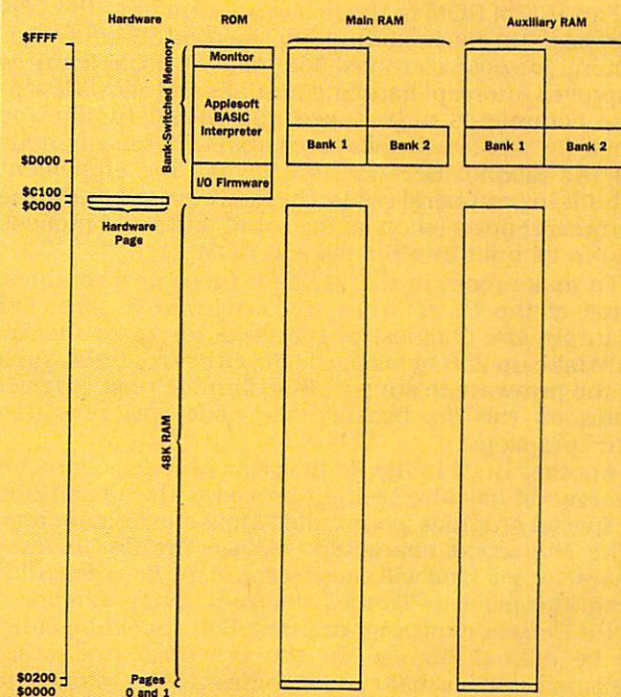


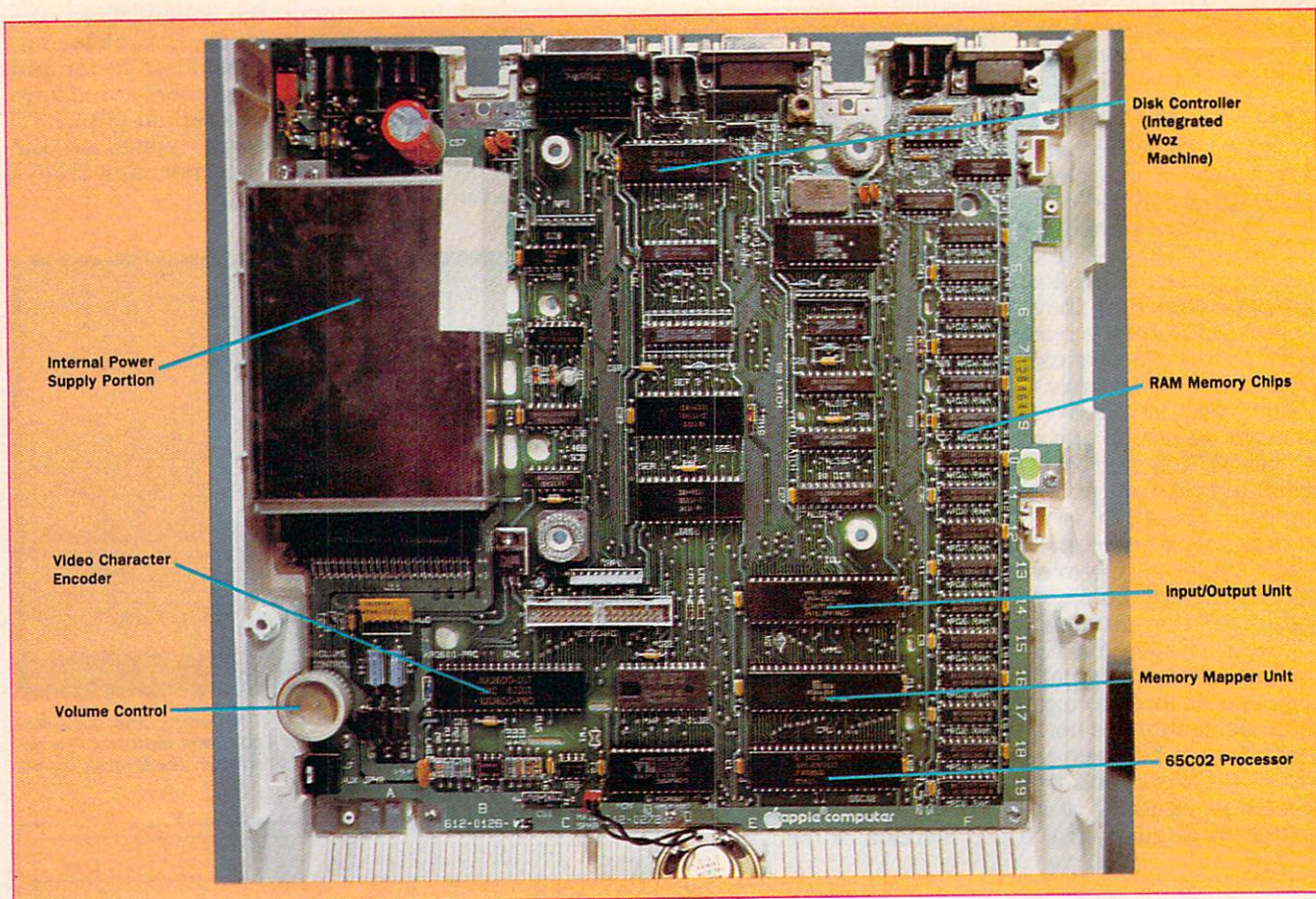
Apple *IIC* Video Display Modes

Display Mode	Screen Page	Mode Supported by Firmware <i>IIC</i>
40 Column Text	1	Yes
40 Column Text	2	No
80 Column Text	1	Yes
80 Column Text	2	No
LoRes (Low Res. Graphics)	1	Yes (<i>AppleSoft</i>)
LoRes	2	No
LoRes mixed, 40 Col. Text	1	Yes (<i>AppleSoft</i>)
LoRes mixed, 40 Col. Text	2	No
LoRes mixed, 80 Col. Text	1	Yes (<i>AppleSoft</i>)
LoRes mixed, 80 Col. Text	2	No
Double LoRes	1	Yes (<i>AppleSoft</i>)
Double LoRes	2	No
Double LoRes mixed, 80 Col. Text	1	No
Double LoRes mixed, 80 Col. Text	2	No
HiRes (High Res. Graphics)	1	Yes (<i>AppleSoft</i>)
HiRes	2	Yes (<i>AppleSoft</i>)
HiRes mixed, 40 Col. Text	2	Yes (<i>AppleSoft</i>)
HiRes mixed, 40 Col. Text	2	No
HiRes mixed, 80 Col. Text	1	Yes (<i>AppleSoft</i>)
HiRes mixed, 80 Col. Text	2	No
Double HiRes	1	No
Double HiRes	2	No
Double HiRes mixed, 80 Col. Text	1	No
Double HiRes mixed, 80 Col. Text	2	No

The Apple *IIC* has the ability to display all of the available modes which are found on a 128K Apple *IIE*. LoRes graphics are defined as 40h x 48v with 16 colors available; HiRes graphics as 280h x 192v with 6 colors; and double HiRes graphics as 560h x 192v with 16 colors.

Apple *IIC* Memory Map





IIc Memory Organization

The memory configuration of the Apple IIc is the same as the memory in the IIe. Both contain 16K of ROM for built-in firmware. The IIc, however, contains twice the amount of RAM as a standardly equipped IIe. The IIc comes with 128K of RAM configured to look like an Apple IIe with the Extended 80-column card. Any program that has been designed to use 128K on an Apple IIe will also take advantage of the full 128K on the Apple IIc.

The 16K of ROM in the IIc is very similar to the 16K of ROM in the IIe, including Applesoft BASIC and other general-purpose routines. The new IIc ROM features improved interrupt handling routines and allows lower-case commands to be typed into BASIC (unlike the Apple IIe, which will always respond with the infamous SYNTAX ERROR). Because the IIc has the equivalent of built-in peripheral cards, the firmware to handle the peripheral ports (such as the serial ports and mouse), had to be built into the system ROM.

To make room in the ROM for these new routines, some of the IIe routines were reduced in size. For example, the diagnostics that take up 1K in the IIe, only take up 256 bytes in the IIc. Other routines, such as the firmware to support 80-columns, require fewer bytes on the IIc, because the code was rewritten and compacted.

Another ROM in the IIc that was used for character generation has also been improved with the addition of special graphics icons called Mousetext characters. [The Mousetext characters replace the IIe "inverse character set" and will cause some of the IIe-compatible programs (such as *Think Tank* from Living Videotext, or the PFS series of programs from Software Publishing) to be difficult to use on the IIc. This, and other incompatibilities, has caused some software vendors to produce two versions of their programs.—Ed.] These graphics symbols are used by programs which feature

the mouse to present a Macintosh-like user interface. Just like the Apple IIe, the IIc can also display the 96 printable ASCII characters, including upper and lower case characters. These characters can also be displayed in inverse, while another mode allows some of them to flash.

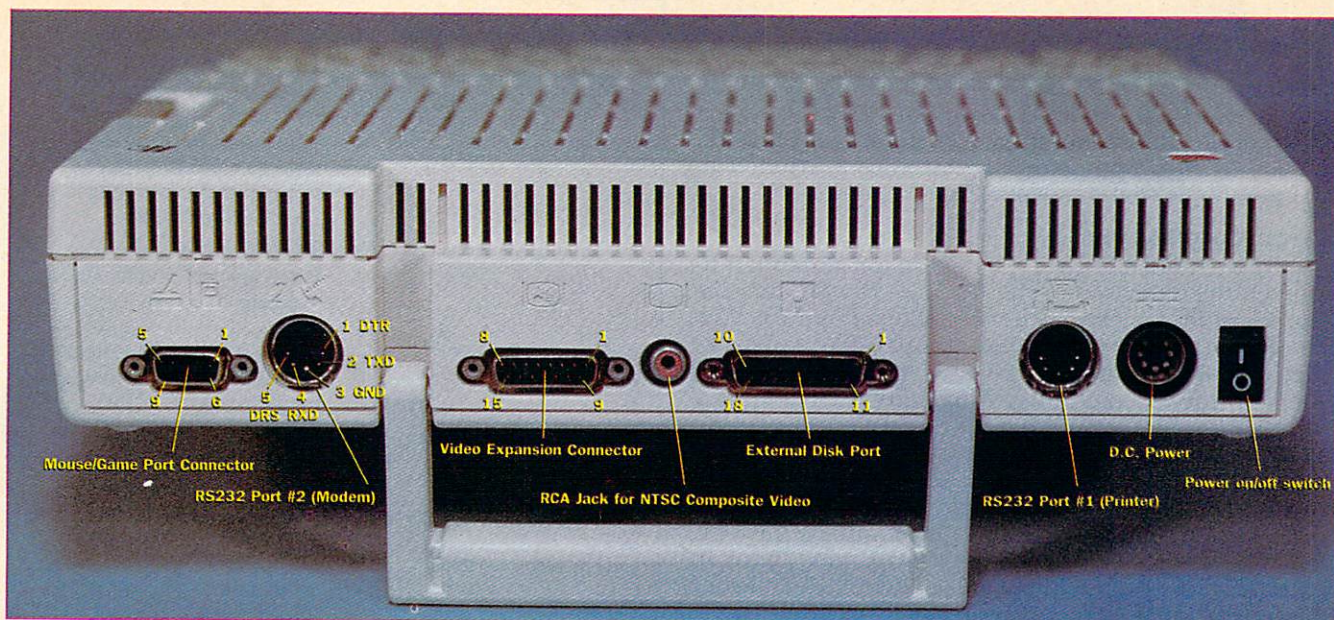
Odds & Ends

The IIc doesn't have a cassette port because of the built-in disk drive. Almost all Apple software is distributed on floppy disks and there has been little demand for the cassette. The firmware to support the cassette has also been eliminated, while the associated softswitch locations are unused in the IIc.

The IIc also doesn't have the full game I/O capability found in the IIe—its game port was built to support only one joystick, instead of two. [The parts of games that require support of two joysticks, such as the two-



The left front side of the IIc showing the volume control and earphone jack. When earphones are plugged in, the internal speaker is disabled.



player option of *One-on-One* reviewed elsewhere in this issue, will not function on the *IIc*. Before you spend the money on a new game for your Apple *IIc*, make sure it only needs a maximum of one joystick.—Ed.] The *IIc* does not incorporate annunciators, which are output signals from the *IIe* game connector. The annunciators were especially handy to the hobbyist for connecting simple peripheral devices to the computer. These omissions were made to allow the *IIc* game connector to be used with the new Apple mouse.

The *IIc* contains the same sound capabilities of the *IIe*, utilizing a built-in speaker. But it also has a few new features, such as a volume control knob and an earphone jack on the side of the computer. When a pair of earphones is plugged into this jack, the built-in speaker is disabled. Finally, Apple included a sound signal on the connector used for the modulator so that a speaker built into a television set may be used.

Ports for Expansion

The *IIc* contains 5 expansion ports, which consist of two serial ports, a video expansion port, an external disk port, and the mouse/game input port. These connectors, with their associated icons, are shown in the photo of the *IIc*'s rear panel.

Serial Ports 1 2

The two serial ports are configured to act as if Apple's Super Serial cards were plugged into slots 1 and 2 of the Apple *IIe*. Generally, port 1 is used to connect a printer while port 2 is used with a modem

for communication. They may also be used for connecting many other peripherals to the *IIc*. These ports will accept most RS232-interfaced peripherals.

The serial ports on the Apple *IIc* use DIN-5 connectors and are RS232-compatible. [Because most peripherals use a "DB-25" type connector,

special cables or adapters must be used with the *IIc*.—Ed.] They can support 15 different baud rates, ranging from 50 baud to 19.2K baud. The serial ports are driven by a special integrated circuit called a 6551 Asynchronous Communications Interface Adapter. This same chip is used by the Apple *IIe* Super Serial Card.

Adding A Printer 1

Port 1 was designed to connect to a serial printer like the Apple Imagewriter or the recently introduced Scribe printer. Once the correctly configured cable is in place, all that is necessary is matching the RS232 parameters. The Scribe printer, which was announced when Apple introduced the *IIc*, retails for \$299, and employs thermal-transfer technology to print on any paper (including letterhead, computer paper, or even transparencies). In high-quality print mode it is capable of printing 50 characters per second (cps); in draft mode it can print at 80 cps. The Scribe can print both graphics and text, and can also print in color if an optional color ribbon is used (though the present cost of the color ribbons makes color printing expensive).

The thermal-transfer technique used by the Scribe is different from the thermal technology used by the old Apple Silentype or the recently introduced IBM Compact printer—both of which have lower resolution and require special paper. The Scribe printer is much quieter than the typical impact printer, which perhaps makes it more suitable for the home or a crowded office environment.

One of the features we especially like about the Scribe is that its cable connectors are on the side of the printer, not the back. This keeps the paper from catching the cables.

Adding a Modem 2

Onboard firmware will support simple terminal communications when either a 300- or 1200-baud modem (such as the Transmodem 1200 or the Hayes Smartmodem) is connected to the second serial port. For users who want more sophisticated capabilities than those offered by the firmware, there are several communications programs on the market. [The rough text of this article was transferred between California and Oregon using Apple's *Access II* program on one end, and the *Person-to-Person* program (from TRUTEC Software of Berkeley, CA.) on the other end.—Ed.]

Other devices such as clocks, music and speech synthesizers, and multiple-function peripherals have



The internal disk drive is always conveniently at hand.

been designed for this port. For example, the Versabox from Prometheus Products (Fremont, CA.) includes a serial/parallel port adapter, printer buffer, and a clock, and the Cricket from Street Electronics (Carpinteria, CA.) combines text-to-speech, sound, and a clock.

Video Expansion

Two of the back-panel connectors are intended to connect video devices to the IIc. Like the IIe, there is a standard RCA jack to connect a video monitor. This jack is used to deliver an NTSC-compatible composite-video signal that creates a display on either a color or monochrome video monitor. The other connector is a 15-pin D-type (DB-15) which is used for connecting sophisticated video display devices to the IIc.

The port includes two signals for connecting an RF modulator (supplied with the computer): a sound signal and the same composite-video signal found on the RCA jack. The sound signal enables the RF modulator to generate sound on the internal speaker of a television. This video expansion connector can be used to directly connect the Apple flat-panel display.

As mentioned earlier, the Apple flat-panel display is a small, lightweight, full-screen, 80-character by 24-line display. It weighs approximately 2.5 pounds and is less than 1.5 inches thick. It does not plug into the wall or require external batteries, but instead gets its power from the system. The flat panel display utilizes a liquid crystal display (LCD) technology which displays 560 horizontal by 192 vertical dots. This technology is similar to that used in many calculators and watches.

Other pins on the port can be used to produce signals compatible with an RGB monitor, and still others can be used for light pen inputs. Koala Technologies Corp. of Santa Clara, California has announced that the Gibson Light Pen will work through this port. A suitably adapted RGB monitor connected to this port will allow a user to display both 80-column text and color graphics on the same monitor. [Apple has introduced the new AppleColor Monitor 100, an RGB monitor, at a suggested retail price of \$599. A low-cost RGB color adapter for the IIc will be introduced by Apple later this year.—Ed.]

Be careful before buying an RGB monitor. Some companies may design their IIc RGB adaptors so that they work only with an Apple-compatible RGB monitor and not with an IBM RGB monitor. The IBM monitors will not display the Apple colors correctly.

The video port contains a power pin with +12 volts. This pin was included so that some peripheral devices wouldn't need a separate power supply. The power from this pin is limited to about 3.5 watts (300 ma. maximum can be drawn from this pin).



Pinout for Video Expansion Connector

Pin	Name	IIe *	Description
1	TXT	N	Text mode signal to video timing chip (TMG). Set to inverse of GR except in Double HiRes mode.
2	14M	Y	14 Mhz. clock
3	SYNC	Y	Video vertical and horizontal sync signal.
4	SEGB	Y	In text mode indicates second low order vertical counter, while in graphics mode indicates LORES.
5	Sound	N	One-volt sound signal from audio hybrid circuit.
6	LDPS	Y	Strobe to video parallel to serial shift-register used to load video data bus into shift register.
7	WNDW	Y	Video non-blank window, includes both HBL and VBL.
8	+12VDC	N	Regulated +12 Volts, drives 350 ma.
9	PRAS	Y	Multiplexed RAM row address strobe.
10	GR	Y	Graphics mode enable signal.
11	SEROUT	Y	Digital video serial output from 74LS166.
12	NTSC	N	Composite NTSC video signal.
13	GND	Y	Ground
14	VID7	Y	High bit of video data bus.
15	CREF	Y	3.58 Mhz. video color reference (same as 3.58M).

*The "IIe" column indicates which signals are available from the Apple IIe video connector with a Y (yes) or N (no).

External Disk Port

The external disk port was designed to allow easy connection of a second floppy disk drive to the IIc. This external disk is accessed by the software as slot 6, drive 2.

The signals for the external disk port come from a special chip, the IWM (Integrated Woz Machine, so named for Steven Wozniak, designer and co-founder of Apple), which replaces the disk controller card of the Apple IIe. The disk port connector is a DB-19 type.

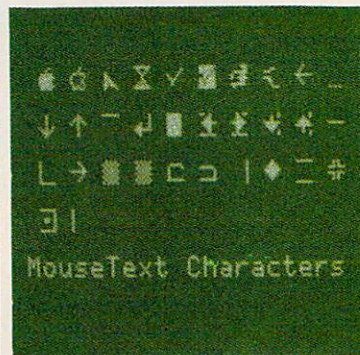
The new IIc external disk *cannot* be connected to an Apple IIe disk controller card with a simple ribbon cable



Apple's New RGB color monitor was shown at the Summer Consumer Electronics Show.



The new Apple flat panel LCD display also made an appearance at the C.E.S. in Chicago.



The "Mouse Text" characters as they appear on the IIc monochrome monitor.



Disk Drive Port Connector Pinouts

Pinout Apple //c rear panel DB-19 connector	Signal	Pinout Apple //e controller card (20 pin header)
1	GND	1
2	GND	3
3	GND	5
4	GND	7
5	-12V	9
6	+5V	11
7	+12V	13
8	+12V	15
9	EXTINT	
10	WRPROT	20
11	PHASE0	2
12	PHASE1	4
13	PHASE2	6
14	PHASE3	8
15	WRREQ	10
16	nc	
17	ENBL2	14
18	RDDATA	16
19	WRDATA	18
	nc	17 (+12V)
	nc	19 (+12V)
	nc	12 (+5V)

This chart shows the //c port connections in the left column and the corresponding pin definitions for the //e in the right column.

(20-pin insulation displacement header to DB-19) because pins 9 and 16 of the DB-19 should not be connected, and pin 19 should connect to pin 20 of the card.

It is possible to modify the cable of a //e disk drive so that it will work on the //c, but you do so at your own risk. It is very easy to make a mistake, seriously damaging your disk drive or computer. If you attempt the conversion, make sure that the DB-19 pin 10 is connected to pin 20 of the disk, and DB-19 pin 9 is not connected (disk pin 19 should also be left unconnected). [Note: Neither the author of this article, nor the publisher of this magazine assume any liability for damages that may arise from this conversion.—Ed.]

The external disk port will typically be used to add a second floppy disk drive, but other companies have shown that this port can be used for many other devices. [The speed and performance of devices interfaced in this manner, however, remains to be seen.—Ed.] Some companies have already announced plans to use this port for hard disk drives and CP/M boxes (these peripherals will contain a Z-80 microprocessor and allow the user to run the CP/M operating system and applications). In fact, at the //c introduction, one company, Quark Engineering Inc., demonstrated a 10-megabyte hard disk attached to this port.

The Mouse/Game Port

The last expansion port is the mouse/game port. This port uses a DB-9 connector similar to the DB-9 on the back panel of the //e. The Apple //e joystick and paddle will work in this port, however, the //c connector does not allow a second joystick or a third pushbutton to be connected, as does the //e game port. Instead, the port contains other signals which support the Apple mouse. Either the //c mouse or the Macintosh/Lisa mouse can be used. It may be possible to design an adapter which allows a second joystick to be connected, but it would not be software compatible with the inputs for the second joystick on the //e. This

means that most of the popular games wouldn't work with this second joystick. Other companies intend to use this port to connect other input devices such as bit pads or graphics tablets, though the Apple graphics tablet won't work in this port. The KoalaPad for the //e will work without any modification on the //c, while ChalkBoard's PowerPad requires a special cable adapter to work on the //c.

The mouse/game port consists of only input signals. Some of the input pins serve a dual role and are shared by the mouse and the joystick.

Addressing Mouse/Game Port Input Signals

Address	Function
\$C061	Switch 0 and OPEN-APPLE key
\$C062	Switch 1 and SOLID-APPLE key
\$C063	Mouse Button
\$C064	Paddle 0
\$C065	Paddle 1
\$C070	Triggers Paddle Timer

All addresses in the chart are in hexadecimal. When you read a byte from one of these locations, only the high-order bit, bit 7, contains valid information. The memory locations for the third pushbutton and paddles 2 and 3 are reserved on the //c for future use. Like the Apple //e, the [OPEN APPLE] and [SOLID APPLE] keys can be used instead of the paddle or joystick pushbuttons.

//c Programming Considerations

The new firmware ROM was extensively rewritten for the //c. Apple started by modifying the firmware used in the //e ROM, removing some bugs in the process, and then adding new routines to support the serial ports, built-in disk drive, and mouse port. Most of these new routines reside in the \$C100 to \$CFFF area, which is reserved in the //e for ROM that is located on the peripheral cards.

The //c, unlike the //e, fully supports interrupts. Within a computer, an interrupt signal is used to indicate that some device needs attention. The firmware makes extensive use of this new feature, allowing the mouse, serial ports, and keyboard to be run in interrupt mode. For example, when the keyboard interrupt mode is used, the processor can continue a time-consuming task without missing a keystroke. The //e is another story: During any long operation within this older Apple (and all of its predecessors), the processor has to periodically check to see if a key has been pressed or it will miss detecting it.

The //c firmware supports use of the mouse, paddle, or joystick. If the mouse is connected, the paddle-read routine in the firmware will take input from the mouse



Mouse/Game Port Connector

PIN #	Mouse Use	Game Controller Use
1	MOUSE ID if grounded Indicates mouse is plugged in.	PADDLE BUTTON 1 when at +5VDC
2	+5VDC	+5VDC
3	GROUND	GROUND
4	X DIRECTION	N/A
5	X MOVE INTERRUPT	PDLO
6	N/A	N/A
7	MOUSE BUTTON active when grounded	PADDLE BUTTON 0 active when +5VDC
8	Y DIRECTION	PDL1
9	Y MOVE INTERRUPT	N/A

Apple IIc Instruction Mnemonics

New 65C02 Instructions Mnemonics

HEX	MNEMONIC	DESCRIPTION
80	BRA	Branch Relative Always [relative]
3A	DEA	DEcrement Accumulator [accumulator]
1A	INA	INcrement Accumulator [accumulator]
DA	PHX	Push X on stack [implied]
5A	PHY	Push Y on stack [implied]
FA	PLX	Pull X from stack [implied]
7A	PLY	Pull Y from stack [implied]
9C	STZ	STore Zero [absolute]
9E	STZ	STore Zero [ABS, X]
64	STZ	STore Zero [zero page]
74	STZ	STore Zero [ZPG, X]
1C	TRB	Test and Rest memory Bits with accumulator [absolute]
14	TRB	Test and Rest memory Bits with accumulator [zero page]
0C	TSB	Test and Set memory Bits with accumulator [absolute]
04	TSB	Test and Set memory Bits with accumulator [zero page]

Additional 65C02 Instruction Addressing Modes

HEX	MNEMONIC	DESCRIPTION
72	ADC	Add memory to accumulator with Carry [(ZPG)]
32	AND	"AND" memory with accumulator [(ZPG)]
3C	BIT	Test memory BITS with accumulator [ABS, X]
34	BIT	Test memory BITS with accumulator [ZPG, X]
D2	CMP	CoMPare memory and accumulator [(ZPG)]
52	EOR	"Exclusive OR" memory with accumulator [(ZPG)]
7C	JMP	JuMP (new addressing mode) [ABS(IND, X)]
B2	LDA	LoaD Accumulator with memory [(ZPG)]
12	ORA	"OR" memory with Accumulator [(ZPG)]
F2	SBC	SuBtract memory from accumulator with Carry [(ZPG)]
92	STA	STore Accumulator in memory [(ZPG)]

instead of the paddles or joystick. Because the mouse is plugged into the same port as the paddles, the user is saved from having to switch between pointing devices. Alas, this will not work for many existing games because they use their own paddle-read routine instead of the built-in firmware.

Here is a simple Applesoft program (under ProDos) that prints on the screen the current X and Y positions, for the mouse instead of the paddles. Lines 10 and 20 initialize the mouse and tell the firmware to use the mouse instead of the paddles for input:

```

5  DS=CHR$(4)
10 PRINT DS;"PR#4"
20 PRINT CHR$(1) :REM initializes mouse
30 PRINT DS;"PR#0"
40 PRINT" PDL 0 =";PDL(0);"____";
50 HTAB 20
60 PRINT" PDL 1 =";PDL(1);"____";
70 HTAB 1
80 GOTO 40

```

[It should be noted that the ProDos utilities diskette (that comes currently with the IIc) has bugs. When we tried to convert an Apple DOS 3.3 formatted program disk to an Apple ProDos formatted program disk using the utility provided, we met with failure. Hopefully, Apple will correct this in the near future. Watch an upcoming issue of HCM for an article detailing programming of the IIc, including more on this problem.—Ed.]

IIe or IIc—That Is The Question

Apple continues to sell the IIe, with sales expected to generally fall into two areas—schools and businesses. Schools will continue to buy the IIe because of its lower entry cost and the availability from Apple of *SchoolBus*, a classroom network. This network allows a teacher to monitor student activities.

Scientific and industrial users will continue to buy the IIe because of the availability of the *Profile* Hard Disk and peripheral cards for data acquisition and process control. But the IIc will serve the needs of the typical consumer very effectively. It can run most of the popular software available, such as *Appleworks*, and can also be expanded (with a modem) for use with computer information services such as Compuserve and The Source. Along with ease of set-up, the machine's portability is seen as its major advantage over the IIe—allowing the business person to more conveniently move it between home and office.

The home user will also find the IIc's portability very useful; it won't tie the computer to a single room. For example, in the afternoon it can be used by the kids running educational programs on the color television, and later in the evening mom and dad can move it into the study for tax planning or letter writing. The study could contain both a printer and a monitor (for 80-columns), peripherals that are typically needed for these activities. The icon-labelled, quick-connect ports on the back panel of the IIc, along with its portability and huge software base, make this scenario conveniently possible.

In summary, for those of you who are looking for a computer to support high-performance applications such as a full-blown accounting system or a large relational data base, I doubt the IIc is the computer for you. But if you're looking for a powerful portable personal computer for home and office—especially one with a large existing software base—I suggest you make haste down to your nearest dealer or user's group "IIc" the newest Apple.

HCM

HCM Review Criteria

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

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

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

HCM Review



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

* Performance—

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

* Engrossment—

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

OR

* Ease of Use—

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

OR

* Ease of Set-up—

How well the product design facilitates easy installation.

* Documentation—

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

Products may also be evaluated in the following areas:

* Flexibility—

Can the product be adapted to the specific needs of the users?

* Cost/Benefit—

Is the product worth the user's investment in time and money?

* Necessity—

Is the product a solution for which a problem already exists?

* Originality—

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

* Longevity—

The "Boredom Factor." Does the program sustain interest?

* Rewards—

Are the audio-visual rewards motivating and appropriate?

* Concept Presentation—

Are the concepts presented clearly, logically, and in depth?

* Special Effects—

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

Attention Software Authors & Peripheral Inventors:

* WANT TO BE DISCOVERED? *

Home Computer Magazine Wants To Give You A Chance!

We are looking for home computer products that have not received the attention they deserve. Each month, we will be singling out one such package for special review. If you have a unique commercial product of exceptional quality—but your advertising and promotion budget has

not allowed you to capture major media attention—we want to see it. We will consider reviewing any product that meets our high standards.

We are an Equal Opportunity Reviewer!

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

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

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

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

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

*Compu-prestidigitation

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



ON THE HOME COURT

Computer Sports Simulations

Sweat socks lying in the corner, dirty shorts and jersey adding their special aroma to the darkened room—these are familiar props to the “workout” generation. But what’s this eerie glow? Ahhh, it’s a video screen, with little figures running about, throwing, catching, making a steal . . . Not real athletes here, but cartoon-like players magically responding to the commands of the human who sits on the edge of a chair—sweating, flushed, and totally involved.

by Wayne Koberstein
HCM Staff

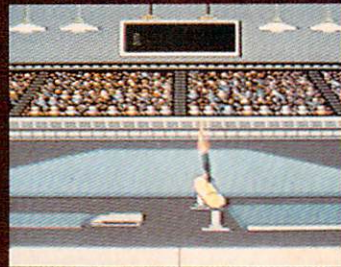
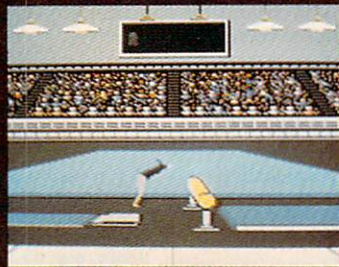
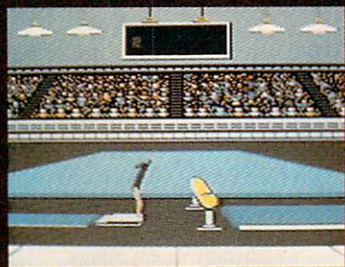
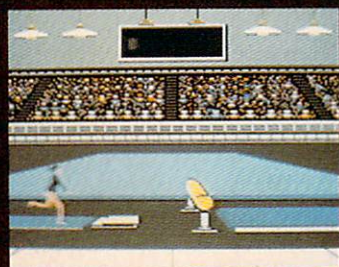
Since the invention of television, many sports fans have spent a lot of time watching rather than playing. You might think computer sports are another lazy, vicarious way to play. Sweatless and safe, replacing gloves and bats with joysticks and fire buttons, sports simulation games may indeed lead to greater indolence. But is this really true? Ask 20-year-old Eric Hammond, who—with the help of two star athletes, Dr. J and Larry Bird—created *One-on-One*. Eric has been programming since the age of 16. Basketball, however, may be his real love. Talking to this young man will dispel any myths you might have about the stereotypical “nerd hacker.”

Computers and sports have intermingled since this new technology first entered the public arena. Initially, the machines simply kept records and figured

averages. Eventually, they became even better at it than everyone's Uncle Henry—that veritable walking library of sports trivia. Nowadays, as shown dramatically in commercials for the 1984 Olympics, computers actually teach people how to run, jump, and throw better. Trainers at all levels of virtually every sport are using software designed to regulate daily routines, and are comparing an athlete's actual performance to an ideal performance generated by a computer.

From Pong . . .

As computers have entered the sports world, sports have entered the home court in a new form of video magic. When *Pong* served up the first volley of video games in 1972, our fundamental relationship to television radically changed. Suddenly, we could be more than just passive observers—we *could make something happen on the screen!* Of course, what we





could make happen was rather crude, and the device we used then was limited to its one simple function, but the basic appeal of video games was the same then as it is now: We can take control of the action and create a unique performance each time we play.

From *Pong* to *One-on-One*, sports games have often led the video arcade field. Both individual and team sports are natural subjects for this medium, with rules long "play-tested" and a wealth of tradition and built-in excitement. Sports simulation is a logical step forward in the evolution of both sports and computers, and its benefits are not confined to one field or the other.

We have seen a continuous chain of improvements in gaming technology since video games helped introduce the concept of the home computer. Early sports simulation games took the rules of a given sport and boiled them down to a basic framework. Stick figures or even little dots represented players; the final scheme often resembled the crude sketches coaches draw to depict plays. Early games were also like chess, in which position is more important than motion. Today's games combine real-time action, life-like animation, and quick feedback between screen and joystick to create a sense of tactile control—of actually performing in the particular sport. Visual excitement and humor, surprises, and sometimes elaborate strategies conjure up the illusion of being there—*inside the event*.

...To One on One

More memory, better graphics, and faster processors in the home models have brought many games (as well as prodigal sons and daughters) from the arcades into our living rooms. That is one side of the change. On the other side are steady advances in the programmers' art. In today's market, a successful video game must perform at a high tech-

nological level, and it must do so in a way that is both original and clever. Just as Olympic athletes defy the limits of their own bodies, software artists sometimes transcend the context of the machine—scoring victories through the sheer strength of their intelligent ideas. Nothing becomes boring so quickly as a game whose fancy new graphics mask a tired old algorithm.

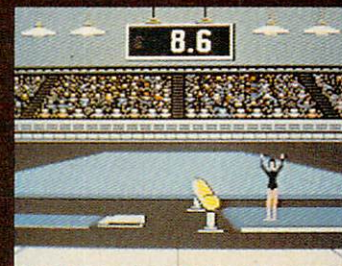
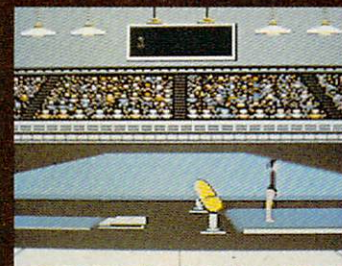
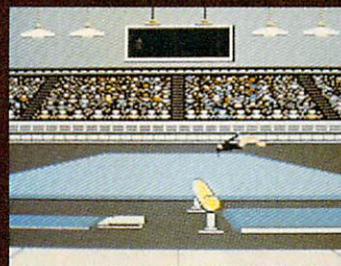
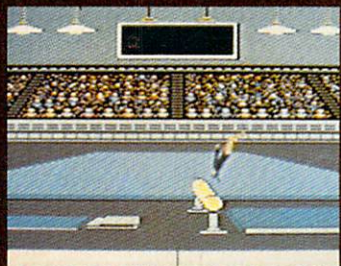
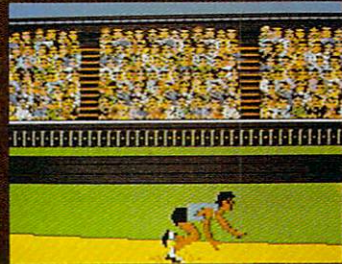
Sports simulation games may have also more relevance to "real" sports than most of us now realize. As trainers develop more techniques for the computer as a training instrument, and as game programmers borrow from these techniques and apply them to simulations, these games will have increasing value to the serious athlete. Every new medium has helped athletes better understand their sport by providing a new perspective. Merely watching great players in action—which through film and TV, more people have a chance to do—can greatly improve a learning player's game. But imagine the possibilities

"When *Pong* served up the first volley of video games in 1972, our fundamental relationship to television radically changed."

in being able to "play" against moves borrowed from the best players in any given sport. Kinetic training of this kind in the home—imparting good timing, reflexes, and a sense of proper motion—

could inspire a whole new generation of athletes.

Although programmers have made good use of their improved palette, they have not fully exploited its potential. Many home video games have reached the technical limits of speed and graphics of the Commodore or Apple II family computers. But other, new machines such as the Macintosh or PCjr have capabilities beyond those that programmers usually work with. For example, the Commodore and Apple II models have almost identical processors: the 6510 on the C64, the 6502 on the Apple IIe, and the 65C02 on the IIc. PCjr uses the 8088, a 16-bit chip that runs about 4-1/2 times faster than 8-bit processors on the other machines. PCjr also sports much higher resolution graphics and a beautiful selection of colors. But programmers have been slow to write for the IBM model—perhaps because they are simply





more accustomed to the old chips, or because IBM itself has not encouraged this kind of development. [Also from this standpoint, it's a shame that Texas Instruments withdrew from the home computer consumer marketplace prior to introducing its model 99/8—a powerful 16-bit machine with a new high-speed microprocessor (TMS9995) and abundant RAM. It just might have been the "ultimate" home machine for sports simulations.—Ed.]

And Beyond . . .

In the near future, we will probably continue to see improvements in sports games programs. Expect to see more involvement of experienced athletes in software design, more clever strategies, more natural on-screen motion, better graphics, sound, speed, and performance. Even with the existing technology of the "older" machines, there is plenty of room for refinement. In the far future of sports simulation, one thing seems likely: In our lifetimes, today's games will seem more primitive than *Pong* now does. Home computers will grow in memory and processing speed, and we will find new ways to interact with them. What else?

We will probably not have to wait another 6000 years to see the kind of moving hologram depicted in the short video game sequence of *Search for Spock*. By then, the joystick will be an ancient artifact, and the hologram replaced by something even scriptwriters have difficulty imagining.

. . . Your opponent appears in the playing area. He is a figure out of the distant past: Julius Irving, one of the greatest players in the ancient game of basketball. He seems so real! With a smile, he reaches out to shake your hand as you step onto the court. His hand is warm. This is perfect! His every move has been carefully reconstructed from the central archive files. They have missed nothing. And now it's your turn to go one-on-one against the best. Sensing each move you make, the computer responds with a countermove by the great Dr. J. Soon you are sweating, flushed, and totally involved . . .

HCM

An Interview with Software Artist, Eric Hammond

Eric Hammond has been programming computer games since 1980. His other professional credits include *Marauder*, *Battle Cruiser*, and *Maze Craze Construction Set*. A music major at Principia, a private school in Illinois, Eric plans to attend UCLA or UCSD in the future to become a member of the college basketball team.

HCM: First of all, how did you get the inspiration for *One-on-One*?

ERIC: Electronic Arts called me up about a year-and-a-half ago. They wanted to do a football game—but I play a lot of basketball, and love it, so I suggested we do a basketball game instead. It started when I thought about a one-on-one situation. I wrote down a script of all the features and how to implement the realism in the game. We discussed it for a long time before the start of the project and then went through a solid stage of design.

HCM: How did you work once you started programming?

ERIC: I started working on just the player mechanics. I spent about three weeks working an animation editor, where you can put the head, arms, legs and torso together to make a complete figure. Then I went through to the very end touching up little dots here and there and seeing that the animation was as smooth as possible.

HCM: So you had the game just about designed before Dr. J and Larry Bird even got involved?

ERIC: That's pretty true. They added things like the fatigue line and the spin. How they spin around was added about two months before the game was finished.

HCM: Did you use any film studies or photographs to capture the look and movement of these two players?

ERIC: Julius came out in July of 1983 to a clinic at the local YMCA, and I have about 500 stills of him shooting around, dunking, and so on. I used those as ideas for where the arms should be when he dunks, and how he has his body set in the air. That helped a lot. Also, I have the tape of the 1982 All-Star game with Bird and J in it. I single-stepped with the Beta machine to figure out how these guys move.

HCM: So, Dr. J actually had more direct input than Larry Bird?

ERIC: Yeah, we talked to him twice—and Bird once. We went back in August that same year to Springfield, Mass. They were both back for the Spaulding Endorsers meeting, and we caught them both at the same time, so we took them aside one day and talked about *One-on-One* and showed it to them. We sat Bird down in a motor home with the game, and he had some ideas about how to incorporate fatigue. That's where the idea came from.

HCM: You can tell that the touch they contributed to the game makes it a lot more realistic.

ERIC: Yes. And the biggest thing is that it inspired me to do a really good game. I thought, if they're on it, I want it to

be the best possible.

HCM: Do you think you've started a trend towards more athletes becoming involved in sports simulation games?

ERIC: That would be great, because their involvement just makes the programmer work that much harder, and it was kind of a sweet thing. I'm a very avid basketball player and also a programmer, so I'm sure that helped quite a bit—especially in designing the computer player, because I thought about what would I do, or what would Julius do, you know—how would he go around thinking and go in for the basket, and where would he jump—everything you can think of.

HCM: What machine did you originally design this game on?

ERIC: It was Apple based. And then I just put it over to Commodore and Atari.

HCM: When you went to the Commodore, did you incorporate sprites at all, or did you just stay in bit map?

ERIC: I stayed strictly in bit map. It worked out really well. When I worked with Commodore, I threw out everything as far as operating systems go. I used most of the RAM up there and left it almost like a bare Apple. Commodore was the biggest conversion, because you need a pretty bizarre routine to do all the masking and photograph background graphics.

HCM: Do you usually work with Apple first and then translate to the Commodore?

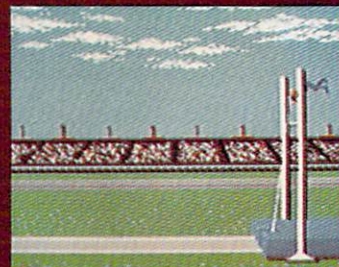
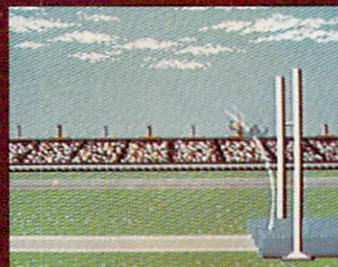
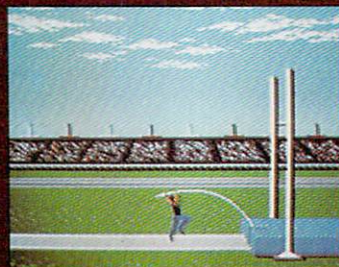
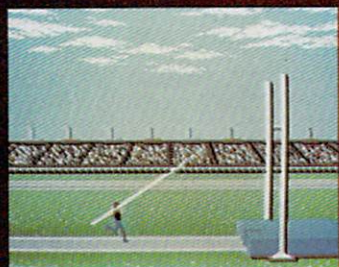
ERIC: If I come on a problem, I might go to the Commodore and see if I can use sprites to help it. But usually, if you do it on the Apple, the other machines fall in line.

HCM: It's like the Apple is the lowest common denominator, is that it?

ERIC: That's exactly right. You can do a lot on the Apple if you just use some tricks here and there.

HCM: How about the future? What are your plans, and what do you see happening to sports games in general?

ERIC: I think you're going to see some really incredible games come out because of the 68000 processor now on the Macintosh. It's pretty exciting, because with that kind of power you can use some really nice intelligence—graphics that haven't been done yet and couldn't be done on a 6502 machine. I'm doing the Mac conversion of *One-on-One* this summer, and I'll be able to put some more things in, like double and triple animation—so it'll be super smooth. In fact, it might turn into a three-on-three.





Name: International Soccer
 Program Type: Soccer simulation game
 Machine: Commodore 64
 Distributor: Commodore Business Machines, Inc.
 1200 Wilson Drive
 West Chester, PA 19380
 Price: \$24.95 cartridge
 System Requirements: Joystick
 Performance: Poor Fair Good Excellent
 Engrossment: _____
 Documentation: _____

shirt color. For example, the grey uniform turns black, or the yellow turns light green. Once the action starts, you need to be on your toes. When you are in possession of the ball, the fire button causes your player to kick or "head" the ball—whichever is appropriate—and also activates the goalie whenever the ball is close to your goal. This "double-duty" for the fire-button leads to a distinct problem when you attempt to have a defender clear the ball away from your own goal. Instead of your defender kicking the ball, the goalie may leap when you push the button, and if your opponent has control of the ball, he can then score easily.

The Rundown On Patterns

According to the documentation, the rest of the players on a team "run patterns in their appropriate zones." This brings up one of the weakest aspects of the game. Note that the documentation does not say, "run appropriate patterns in their zones." Often the other players run away from the ball, or just stand still as the man with the ball runs by them. This is quite frustrating.

In addition, it is up to you to figure out which player is from which zone. When you are controlling a player, you can take him anywhere on the field, but if you think your other players on the screen will try to get in position for a pass, you may be in for a rude surprise. A midfielder not under joystick control won't race to assist your man, who may be charging to make a goal. All too often the midfielder will turn and run back toward midfield instead. If, however, you are controlling the midfielder, you can make him take the ball right in to try to score. It's too bad the game's programmer made all the players identical, because if the players were somehow identified by position, it would help you plan your attack strategy. The documentation never does explain what it means by "appropriate zones," so you only discover its implications if you already know something about soccer, or after you've played this game for some time.

The game is not totally bug-free. Occasionally part or all of one of the players momentarily disappears from the screen. The player still exists and is active, but for a second you can't see him—a disconcerting characteristic.

But even with the drawbacks mentioned above, *International Soccer* is exciting and very entertaining—even, you may find, pleasantly distracting. And because of the number of difficulty levels it offers, it is enjoyable the day you buy it, and promises to stay challenging.

HCM

Sure the party was loud and confusing, but when I asked Charlie who won the soccer game he was watching, to my surprise he said he did. Then he pulled out a cartridge to show me, and the players on the screen froze in place. So this is *International Soccer*!

At a casual glance, this game does seem real enough to deceive—although a closer look reveals it to be a computer simulation. Good hand/eye coordination and a working knowledge of the real game is all you need to sit down and enjoy this excellent video version.

Regulation soccer is played with eleven players to a team, but the Commodore facsimile has only seven per team. This, however, doesn't limit the excitement level. In fact, because the joystick can't control several players at a time, six players and a goalie is plenty for you to handle. You can play the game against either the computer (at any of 9 levels of difficulty), or a human opponent. Level 1 is, according to the instructions, "eminently beatable, an adequate opponent for a young child." I found this statement to be pretty accurate, as I was able to win the first time I played at this level—thus qualifying me as a young child... By contrast, I was never able to beat the computer at level 9, although I was able to compete at level 7 after a good deal of practice.

International Soccer's graphics are colorful, with remarkably good animation—making the game fun to watch. Apparently, the program uses bit-map mode, with horizontal "smooth-scrolling" of the screen for maximum fluidity of motion. The players' movements, however, are a bit choppy—but considering the amount of action on the screen and the speed of the C-64's processor, we could hardly expect the motion to be any smoother. One might complain about the limited viewing area—the screen "pans" up and down the field like a TV camera—but, because the field is so well marked, no one would get lost.

On Field Fashion

Color is a big factor in the game's appeal. Not only is the entire stadium as colorful as a bright spring day, but

INTERNATIONAL SOCCER

A review by Roger Wood
 HCM Staff

"Good hand/eye coordination and a working knowledge of the real game is all you need to enjoy this excellent video version."

choosing from various team colors lends even more variety to game play. One problem though: If one team has red jerseys, and the other team has orange, you may have trouble distinguishing between the two on screen. The game remains consistent with real soccer by selecting colors for the goalies' uniforms different from either of the teams' colors.

International Soccer consists of two halves, each lasting 200 "time units" (seconds)—and there are no time-outs or pauses. Before the ball is put in play, the players all run out from the side of the field to take their positions. At the end of the half, both teams file off the field and then return, which adds a 15-second delay at the beginning of the game and another 30-second delay at the half. It all seems rather unnecessary. One could argue that this little ritual adds a nice touch of realism (and at first it is kind of cute), but I soon found it boring. If the player could control these delays, the game would be much more flexible and enjoyable. Instead, once you start playing, the game time is totally controlled by the computer. You can re-start a game from the beginning by pressing the [RESTORE] key. I discovered this option quite by accident—it isn't mentioned in the documentation.

The two opposing players closest to the ball are under direct joystick control, and are identified by a change in

one on one

A review
by **Steve Nelson**
HCM Staff

Frankly, when I heard that the new basketball stimulation game, *One-on-One*, was challenging, fun, and even had decent graphics, I was ready to be disappointed. Over the years I have seen a lot of sports simulation games designed for home computers, and most of them had crude graphics and were sadly lacking in realism. But when the package finally arrived in our offices—even before playing the game, I knew something about it was going to be different. And when I played it, it knocked my sweatsocks off. What a great game!

Levels of Play

One-on-One comes with four different levels of play, each one requiring more skill and practice than the last. The first level, Parks and Recreation, is fairly easy to master. The game is slow-paced, and the referees don't call it very close. Here you can practice your best moves and shots before advancing to the next level, Varsity. On this level, your opponent plays tougher, stealing the ball, slam dunking it, and generally embarrassing you off the court. Persistence and honing your basketball skills will eventually enable you to keep up on this level.

The next step up is College Ball, and here you begin to understand what the word "fast" means. The Doctor and Larry Bird tear up the halfcourt with steals, turn-around jumpers, slam dunks, and perimeter shooting that is not to be believed. The referees call the game much closer now, and if you can win consistently at this level, you deserve a Michelob Light after the game.

The final level is Pro, and if you plan on competing here, you had better hang on to your gymshorts, because the computer is looking for a blow-out.

Once you decide on which level of play you want to compete, you have to choose whether you want to play against Larry Bird or the Doctor. There is also a two-player option so you can compete against a friend. This is a great feature, as playing the computer on the higher levels is quite difficult.

One-on-One is a very sophisticated game—the players' movements are fast, very controlled and quite realistic.

HCM Review



Name: One-on-One
Program type: Basketball simulation game
Machine: Commodore 64,
Apple IIe, IIc
Distributor: Electronic Arts
2755 Campus Dr.
San Mateo, CA. 94403
Price: \$40 diskette

System Requirements: Joysticks, disk drive

Poor Fair Good Excellent

Performance:
Engrossment:
Documentation:



*Hey—all you armchair athletes:
Care to find out how tough it is to play against
Larry Bird or Dr. J.?*

The things you can do with the ball via the joystick are absolutely incredible; turn-around jump shots, slam dunks, reverse layups, even 3-pointers just roll off your player's fingertips. You can get position on your opponent, cut to the inside or the outside, block shots, and even steal the ball.

Other realistic game elements included are the 24-second shot clock, hot streaks, instant replays, player fatigue, turnovers due to traveling or fouls, and the ability to shatter the backboard with a particularly powerful slam dunk (a janitor comes out and sweeps up the mess so you can continue playing).

Take It From The Pros

Built into the program are the two pro athletes' individual playing characteristics. Dr. J. is a step quicker and flashier than Larry Bird, but The Bird is a little bigger and stronger—giving him the edge when rebounding. He also uses his greater strength to his advantage when playing on the inside, even though he is much better at hitting the hoop from the outside. On the other hand, Dr. J. relies on his faster speed, which allows him to cut to the inside or go around the defense.

Many of the outstanding features that were incorporated into *One-on-One* were suggested by these two basketball superstars. For instance, Larry Bird insisted that the screen figures actually become tired during the game, and Dr. J. suggested adding the ability to block a shot from behind when a player goes up for, what he calls, a lazy layup. Their input is obviously one of the reasons *One-on-One* works so well.

Eric Hammond, the game's creator, has done an excellent job of transferring the excitement, challenge, and even the sounds of basketball into your home computer. *One-on-One* is by far one of the best sports simulation games I have ever played.

One-on-One plays almost identically on the Apple IIe and the C-64. Unfortunately,

the Apple IIc can accommodate only one joystick, so you must use the keyboard for defense whenever you are in the two-player mode.

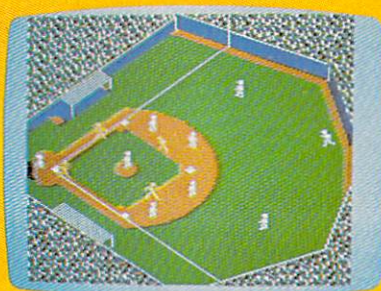
The game comes with excellent documentation, including quotes from the Doctor and Larry Bird as well as pictures and helpful hints on playing offense and defense. Part of the initial appeal of the game is the way it is all creatively packaged, designed to look like a record album.

Although *One-on-One* is a great game, in terms of pure basketball, it's still not perfect. For instance, the 24-second shot clock varies with the level of difficulty, speeding up as you play the more competitive levels. Also, you can reset the clock by just shooting the ball—you do not have to hit the rim, as official NBA rules require. You must press the fire button to shoot, steal, or block a shot and in fast action play, the computer can confuse the three. This confusion can cause turnovers on the college or pro levels if, for example, you haven't cleared the ball before the computer thinks you've taken a shot. (On the two lower levels, you get to keep the ball, but you must inbound it.)

The computer also does not change its strategy in different game situations—such as playing for the last shot, or stalling when ahead—but it is so good on the higher levels, it seems to win most of the time without any strategy.

These are all minor complaints and don't detract from the game's overall enjoyment. Given the limitations of processing speed and memory, adding features to correct these departures from real basketball would unduly complicate this video game and impede its already excellent flow. If you enjoy a good game of basketball, or if you're just looking for a challenging computer game, *One-on-One* will meet all of your expectations and then some. Now excuse me while I go thrash Dr. J. one more time.

HCM



Name: Star League Baseball
 Program Type: Game
 Machine: Commodore 64
 Distributor: Gamestar, Inc.
 1302 State St.
 Santa Barbara, CA 93101
 Price: \$29.95 disk
 System Requirements: Joystick, disk drive
 Poor Fair Good Excellent
 Performance: _____
 Engrossment: _____
 Documentation: _____

**"All that's missing
 is the wind,
 and a few beer
 and hot dog vendors."**

Make no mistake about it: From the playing of the national anthem, to the seventh inning stretch, to the tie-breaking extra innings, this is a true-to-life baseball game, and you had better be ready to play when your team steps onto the diamond. Gamestar's *Star League Baseball* ranks with major league games, so if you are competing against the computer, beware—it makes no mistakes.

A good measure of sports simulation programs is the extent to which they force their users to think and react as if they were physically playing the sport. This game measures up quite well in this respect. What's amazing is that—although on screen you see graphics little better than stick figures chasing a tiny ball—the computer game plays so much like the real thing.

After spending some time on the Batting Practice option, send your player up to the plate in the Game mode. As the strikes and balls whiz by, your fingers may twitch on the joystick, itching to swing. To prevent striking out, you must constantly keep your eye on the ball, scrutinizing every pitch to the exclusion of outside distractions. Once on base, you can wait for your chance to steal, lead off a bit, or perhaps become trapped in a game of hotbox.

When in the field, you must judge the height and distance of fly balls (by watching their shadows) to make the catch. Then you must quickly pick the correct base at which to make the play, aim, and throw. You must also decide what kind of pitches the pitcher will throw. You have a choice of almost every legal pitch. When the opposition has a chance to score, the "organ" starts up the call to "CHARGE!" putting even more pressure on you to not walk the batter or allow a hit. It's all up to you.

Play Ball!

Star League Baseball can accommodate two players with joysticks, or you can play alone against the computer. Begin by selecting your starting team—the LINERS hit for an average, the SLUGGERS hit for the fences. Try to pick the team that takes advantage of your opponent's fielding weaknesses. The same goes for pitchers—

Star League Baseball

A review by Dana M. Campbell
 HCM Staff

you have a choice of three: "Heat" Muldoon, the wild one; "Curves" Cassidy, the controlled one; and the relief pitcher, "Knuckles" Flanagan. Each throws eight different pitches, selected with joystick directions and released with the fire button. The pitches are labeled in the manual. It would be nice to be able to bring in a relief pitcher before the seventh inning, but you cannot. If Heat wears out in the fourth inning from throwing too many fastballs, you're stuck with him.

There are no definitive easy or hard levels of play to choose from. By adjusting the pitching and hitting combinations for each team, you have many possible game strategies, and whether they are easy or hard depends on each user's various strengths and weaknesses.

Fielding, bunting, hitting, throwing, and running are accomplished via various combinations of pressing the joystick button and moving the stick. After a few games, the moves required are quickly memorized; however, it is the timing of those moves that is crucial, and difficult to master.

Three factors make *Star League Baseball* more frustrating than it need be. First, the players run too slowly when retrieving the ball, and the ball travels much too slowly when thrown from the outfield. It rarely, if ever, makes it in fast enough to prevent a run. Making an out-at-base from an infield hit is a little bit easier, but not much.

Second, as the manual states, the player closest to the ball will move to make the play, but, if he misses it, he must chase it. Control does not switch between players to provide a backup when an infielder errors. Thus, if you send the closest player back to stay near his base, the ball just lies there.

Third, the base runners move forward or backward as you move the joystick right or left. It would be easier

to move the joystick in the four compass directions corresponding to the way the diamond faces you on the screen. Conversely, when throwing, the ball travels according to the way the diamond is set up on the screen, not according to the direction the player is facing. This method of moving and throwing is confusing and unnatural, and I lost quite a few runs because of it.

The Breaks Of The Game

You do get a few breaks, however. The computer team doesn't lead off or steal, and its runners only advance one base on a hit, no matter how long it takes you to retrieve the ball. As an experiment, I purposely dallied near the stands for a while before throwing in the ball, and nobody moved a muscle (or a pixel) once on base.

During batting practice, "Heat" throws a variety of pitches for you to swing at. He throws very few balls. The computer infielders retrieve your hits, tag up, and throw the ball in. The pace here is quick, and your concentration must be keen.

The "Official Souvenir Program" provides all the information you'll need when you leave the dugout, from loading the diskette to a scouting report on the pitchers. Simple diagrams detail the joystick maneuvers required for certain moves, and "Tips for Stars" includes some valuable hints that apply whether you are playing a real game or this video version.

With the exception of the three drawbacks mentioned above, *Star League Baseball* is consistently realistic, both in its performance and the overall atmosphere it provides. The game's sound imitations of the crack of a bat, the pop of a ball into a glove, and a cheering crowd all make you feel like you are spending an afternoon in Candlestick Park. All that's missing is the wind, and a few beer and hot dog vendors.

HCM

HCM Video Olympiad '84

Olympics Simulation

by Steve Nelson
HCM Staff

During this Olympic year, a bonanza of sports simulation games has flooded the market. Of these, some of the most exciting and realistic are depictions of the Games themselves.



Name: Summer Games
Program type: Olympics simulation
Machines: C-64,
Apple II Family (soon)
Distributor: EPYX
1043 Kiel Ct.
Sunnyvale, CA 94086
Price: \$35/39 diskette
System Requirements: joysticks, disk drive

Poor Fair Good Excellent

Performance:
Engrossment:
Documentation:

Name: HESGames
Program type: Olympics simulation
Machine: C-64,
Apple II Family (September)
Distributor: HesWare
150 North Hill
Brisbane, CA 94005
Price: \$34.95 diskette
System Requirements: joysticks, disk drive

Poor Fair Good Excellent

Performance:
Engrossment:
Documentation:

Name: Decathlon
Program type: Olympics simulation
Machine: Apple II Family
Distributor: Microsoft
10700 Northrup Way
Bellevue, WA 98704
Price: \$29.95
System Requirements: joysticks, disk drive

Poor Fair Good Excellent

Performance:
Engrossment:
Documentation:

For review purposes in our own HCM Video Olympiad, we grouped three of these outstanding simulation programs into "teams" represented by three different flags: Microsoft (*Decathlon*), HesWare (*HESGames*), and EPYX (*Summer Games*). We then evaluated each team and gave each an overall score based on their performance in terms of graphics, animation, and realistic play. Then, we awarded the medals: bronze, silver, and gold. Are you ready for the instant replays and final results? Let the games begin!

Decathlon

Leading off on the quest for gold is *Decathlon* by Microsoft. Developed for the Apple II, it depicts the sport long regarded as the premier event in the Summer Games. *Decathlon* is really ten games in one: the 100-meter dash, long jump, shot put, high jump, 400-meter dash, 110-meter hurdles, discus throw, pole vault, javelin throw, and 1500-meter run.

As the athlete competes in each event, he scores points; the highest total score for all ten events determines who gets the gold medal. If you are playing alone, your goal is to try and beat Bruce Jenner's record of 8,618 points. If you are competing against other players, the highest total score wins the gold.

Of all the *Decathlon* events, shot-putting is the most difficult to master,

but for technical rather than "athletic" reasons. It took me several tries with various joysticks before I found one that would work properly. (Many joysticks will not set to a true "0" center, as this event requires.)

To compete in the other events, you must use the keyboard—and it takes some practice on it before you can successfully finish the game. In fact, some of the instructions are quite complex and require more of a workout for your fingers than any insight into the sports themselves.

Unfortunately, the graphics of *Decathlon* are not really in the same league as the other two games reviewed here. Instead of creating all elements of an event graphically, including crowd and background scenery, Microsoft's *Decathlon* does not go into detail. Instead, its scenario is very simple: a plain figure on a black background, or a track with a pointed dot on it (representing the runners).

Score:	
Graphics	7.0
Animation	7.5
Realism	7.0

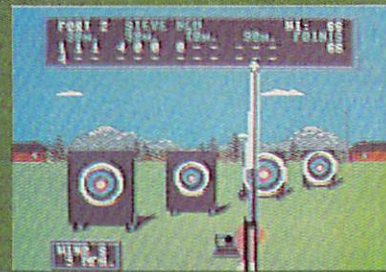
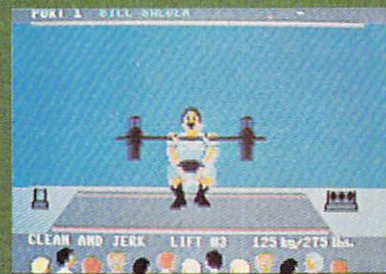
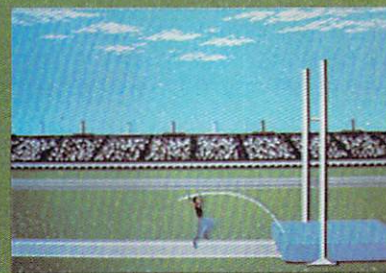
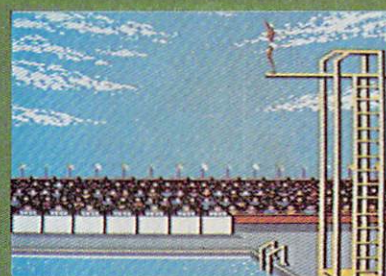
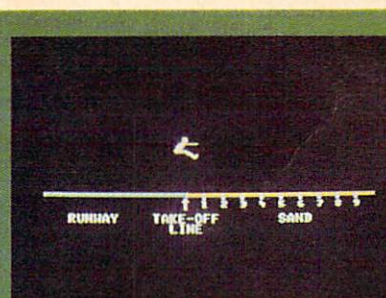
HESGames

HesWare's entry for the HCM Video Olympiad is *HESGames*. This game includes six events: the 100-meter dash, 110-meter hurdles, long jump, archery, diving, and weight lifting (which is actually two events in one—the clean and jerk, and the two-hand snatch). All of these events have excellent graphics which portray the events, athletes, crowds, and background scenery in such intricate detail that I felt as if I was watching a well-made animated feature. You control the athletes' movements with the joystick, and their response is very good—they run, jump, shoot, dive, and lift with a smoothness and control that is hard to beat. Whether you are playing against other competitors, or by yourself, *HESGames* is challenging, fun, and visually quite a treat.

You can compete in just one event, or go for the gold in all eight, one after another—similar to *Decathlon*. Also, you can at any time look up the world record in each event, and use it as a goal to strive for while competing.

Like *HESGames*, this program takes full advantage of the Commodore's video chip. Even the background is very detailed and colorfully realistic. The actual competitors move with a very fluid motion—you have total control with the joystick, and the response of the athletes is phenomenal. But, unfortunately, it doesn't allow you to save performances on disk.

Each event comes complete with excellent documentation, giving step-by-step instructions on how to make the athletes perform at their gold-medal best. Some of the events are quite complicated, however, and



1. Decathlon/Long Jump
2. Decathlon/Pole Vault
3. Summer Games/Diving
4. Summer Games/Pole Vault
5. HESGames/Weight Lifting
6. HESGames/Archery

"You control the athletes' movements with the joystick, and their response is very good—they run, jump, shoot, dive, and lift with a smoothness and control that is hard to beat."

During play, you can save a particularly good performance on disk, and the game even offers you an instant replay option. You can also check the Olympic record or world record for the event you are competing in and try to beat it.

I found very little that bothered me about this game, except for some trouble I had entering information into the main menu screen—perhaps because there are so many options to respond to. This screen asks you to enter so much information—right down to the color of your socks—that it's easy to tire of the whole process. One other objection concerned not being allowed to practice an event before you compete. You can observe an event (except archery) in the demo mode, but you cannot participate. The other competing "teams" allow you unlimited practice time—as should *HESGames*.

almost require holding the joystick in one hand and the instructions in the other until you get used to playing.

Score:	
Graphics	9.5
Animation	9.0
Realism	9.0

Awarding the Medals

The final results are all in, the points have been tabulated, and the medals are about to be awarded.

Overall Team Score:

Gold	
Summer Games (total points 27.5)	

Silver	
HESGames (total points 27.0)	

Bronze	
Decathlon (total points 21.5)	

Score:	
Graphics	9.0
Animation	9.5
Realism	8.5

Summer Games

The final competitor, *Summer Games* by EPYX, comes closest to the real thing. From the beautiful opening ceremonies, to the final competition, *Summer Games* provides superb graphics, excellent control of the athletes, and is, in general, a fine example for all sports simulation games.

One to eight players can compete in eight events including the pole vault, diving, 400-meter relay, 100-meter dash, gymnastics, freestyle relay, 100-meter freestyle, and skeet shooting.

Summer Games, because of its realism, garnered the gold medal. *HESGames*, by not allowing you to practice events before actual competition, has to settle for the silver. The bronze goes to *Decathlon*, which tried hard, but just couldn't quite keep up. *Decathlon* is simply outclassed, but in the interest of fairness, we should mention that it is a few years old—and it was programmed on the graphically-difficult Apple. This comparison is a good indication of how far programmers have advanced in terms of animation and graphics—at least on the Commodore.

The Apple II Family versions of *Summer Games* and *HESGames* were not available in time for review. (See review box.)

HCM



Name: Country Club
 Program type: Golf Simulation
 Machine: TI-99/4A
 Distributor: User-Happy Simulations
 133 North Prairie
 Whitewater, WI 53190
 Price: \$19.95 disk or cassette
 System Requirements:
 Extended BASIC, Memory Expansion
 Poor Fair Good Excellent
 Performance:
 Engrossment:
 Documentation:

Country Club

A review
 by Tom Green
 HCM Staff

When the adjustable club hooked into the golf market, golf purists were aghast; a serious golfer would never consider using such a contraption. Nevertheless, the adjustable club gained popularity.

But a game that simulates golf? To a seasoned player, this must be the last affront—or is it really an affront at all?

User-Happy Simulations thinks not. They offer a golf game for golfers and video-game buffs alike. *Country Club* delivers hours of realistic and challenging golf play, conveniently bringing the fairway into your living space. Now, teeing off is as easy as booting up.

Course Guide

Country Club is noticeably different from other simulation gameware. No pre-game jingle. No arcade tune to accompany tee-off. The lack of music, however, is not meant to defy arcade protocol—there is a practical reason why this was done. The program uses memory space to achieve realism in the kind of variables associated with the game of golf, not for catchy tunes. Let's leave the caddy shack behind now and scout the fairway for hole #1.

You begin by choosing from three options: Advanced, Beginner or Editor. There are two different 18-hole courses: Option 1 will allow play on the more difficult course, and Option 2 is for play on the easier course. Option 3 is a subroutine that allows "hacker" manipulation of fairway graphics.

Handicap scoring adjusts for play levels: Pro, Good, Fair and Clod. By choosing the Clod level, for example, a player introduces more error into the control and power of strokes, but the game becomes more challenging. You can deduct your total Clod level handicap at the end of a game and

possibly underscore someone at the Pro level.

"At the Clod level, you can deduct your total handicap at the end of the game and possibly underscore someone playing at the Pro level."

The club being used, the force of the shot, the terrain where the shot takes place, and the wind all affect the accuracy of a shot. Players have 15 clubs from which to choose. The wedge and short irons offer the best control, and naturally as a club gets "longer," control decreases. Thus, the club with the least control is the driver, because it is the longest club.

The force of a shot is set on a scale from one to ten, with one being the weakest stroke. However, only 9 and 10 seem to have any effect on control, simulating the unpredictable results of an overpowered shot.

Terrain also affects control, as in real golf: On the putting green and tee, excellent control is possible. Shots from the fairway usually have good control. But when shooting from the rough, control is fair to poor, and from a sand trap, the outcome is unpredictable. The wind is present only in the advanced game, affecting direction and distance. Shots must be adjusted to allow for the bearing and strength of the wind. "Yardage Cards" are supplied with the package to approximate the distance of each club stroke for a certain power and terrain orientation.

Ideally, the direction bearings in a game like this would have 360 settings, to simulate the degrees of a circle. *Country Club* lets you determine direction on scale from 0 to 60 (think of a stopwatch and its hand as a pointer in one of 60 directions). The precision of this scale is more than adequate for

increasing the accuracy of a shot. Another accuracy aid is a transparent card (supplied with the package) with direction arrows indicating every fifth position which, when held in front of the screen, serves as a visual guide in choosing a direction for your shot.

Country Club displays fairways as aerial views. Detailed enough to include trees, the fairways are designed to introduce as much variation in terrain and hazards as a real course.

Fore!

The order that players tee off and take their subsequent shots follows traditional golf etiquette. When the last player reaches the putting green, the fairway screen is replaced with an aerial view of the putting green, and the screen displays conditions and information appropriate to the putting green (direction of break, steepness, etc.). Running score summations are listed for each hole after the final shot of the hole is played, and individual handicaps are also displayed.

Par For The Course

Country Club's overall performance is true to the characteristics of real golf. Real golf can be a slow game—even boring for some. But *Country Club* moves along at a satisfying pace, with no undo pauses or delays. Although adjusting all parameters before making a shot can take some time, there is nothing illogical or frustrating in the process. In fact, to the real golf enthusiast, breaking down a swing into all of its components can be very interesting and informative.

Documentation

The user's guide is easy to follow and understand and could well serve as an introductory tutorial on the many elements and strategies involved in a basic game of golf. In addition, game features on scoring, club choice and use, and insights to game play are clearly outlined.

The Editor subroutine provided for changing the formats of the different fairways lacked documentation at the time this review was written. User-Happy Simulations will be issuing a "hackers" guide supplement to the *Country Club* manual in the near future.

All in all, *Country Club* is a real golfing adventure for you TI owners; its value goes beyond the mere fun of a video game. If you're still working on that swing, this program just may teach you something.



Name: Pole Position
 Program Type: Arcade Game
 Machines: TI-99/4A, VIC-20, C-64, PCjr, Apple II Family
 Distributor: AtariSoft
 Atari, Inc.
 P.O. Box 61657
 Sunnyvale, CA 94086
 Price: \$34.95 disk, \$44.95 cartridge
 System Requirements: joystick optional
 Poor Fair Good Excellent
 Performance: TI-99/4A
 C-64, VIC-20
 Engrossment: TI-99/4A
 C-64, VIC-20
 Documentation: TI-99/4A
 C-64, VIC-20

Pole Position

A review
 by Dana M. Campbell
 HCM Staff

Ladies and Gentlemen, start your engines, cries the announcer, and you rev your engine, keeping a firm, gloved grip on the gear shift knob. It's the last week in May, and your moment has come to attempt to qualify for a position in the Indy 500. You wait for what seems a million seconds before the signal light turns green, and then in an explosion of dust and noise . . . you're gone.

This is the scenario that was successfully simulated by Atari in the arcade game *Pole Position*, now available for the home. The game's basic concept is easy to understand but challenging to perform: guide a turbo-charged Formula 1 racer around a track without hitting any obstacles or other cars, and do it fast enough to qualify for the coveted pole position (first row, inside lane) in the main event. Of course, various factors have been thrown in to make the game more stimulating and realistic, and it is these touches that make *Pole Position* the best of the driving-simulation games.

For instance, if you round a corner too fast, your car will skid, slowing you down while the clock ticks away. Also, unlike similar games, the other cars on the road do not remain stationary—they will move around on the track as you approach, and ruthlessly attempt to cut you off as you pass them. Nerves of steel and a steady hand are all that will get you through some of those tight squeezes.

Staying On Track

Although a keyboard can be used to play the game, the ease of a joystick is akin to power steering. Pressing a (fire or keyboard) button will toggle you from low to high gear and back again. Three levels of difficulty test

your skill, with the only apparent difference between them being that the number of curves and cars on the track increases with each succeeding level. Accumulate points by completing a lap, by passing a car, and for each second of time left on the clock after you've crossed the finish line. If you finish the qualifying race in 73 seconds or less, you receive a position (from one through eight) in the main race. Finishing the big race within specified lap times will garner you some Extended Play.

The differences between the TI-99/4A and the Commodore versions of *Pole Position* are quite noticeable. On the TI, players can choose from one to eight laps per race, but are limited to a maximum speed of 195 miles per hour (mph). Players on the VIC-20 or C-64 can accelerate up to 244 mph, but must contend with a set number of laps every race.

The TI game is also not as well crafted as the Commodore games. In the TI version there is no signal light to begin the race—players must rely on the sound alone. The car responds slowly on turns, and loses speed a bit when shifting from low to high gear—even when shifting at the recommended 100 mph. The background scenery changes in jerky movements which supposedly depict motion. This "motion", however, reminded me more of oldtime movies than a high-

speed race. In addition, cars passed on the track aren't actually "passed"—they simply disappear just before your car is opposite them. I actually ran over a few such cars in my path and did not crash. Too bad I can't do that on the highway . . .

The Commodore versions feature brighter, more detailed graphics, and the other racers on the track provide more of a challenge—avoiding them requires some pretty intricate weaving among the cars. The traditional signal light is there to start the race, as are the appropriate engine sounds. Shifting and motion occur smoothly. The C-64 game even offers an option that allows players to pause during their race. You can't pass on the grass in these versions, but you're better off not trying to pass on the outside anyway—you'll be quickly cut off and will likely crash.

Too Much of a Good Thing

The game's instruction manual is simple but complete. In fact, the Strategy Tips almost tell you too much, for if you stay on the center stripe as suggested, you can bypass everything, taking away most of the game's

"Unlike similar games, the other cars on the road do not remain stationary—they will move around on the track as you approach, and ruthlessly attempt to cut you off as you pass them."

challenge and fun. (You can't get as good a lap time though by staying on the middle line. It's faster to hug the curves.)

The home game's sound effects exactly mimic the original arcade game, from the starting tune to the crashes.

As mentioned above, *Pole Position* is a game that is easy to learn, which is also its main drawback. Once you've taken a few spins around the track, the track and its scenery become predictable and boring. Teenagers on up may tire of the game after a few plays, while younger kids may spend more time with this simulation of something they are not yet allowed to try in real life.

Pole Position would be more interesting if a few slick spots were placed on the track so that they couldn't be seen from a distance. Adding another gear or two would also keep things hopping, and why not require some quick pit stops after every few laps? Finally, a few scenery changes would help keep the race lively. The arcade version of *Pole Position II* features this last suggestion, so perhaps we will see it added to a new home version.

In the meantime, practice hugging those curves, and try not to hit any billboards this time . . .

Versions for IBM PCjr and Apple II Family had not yet been released at press time.

HCM



Name: Bermuda Race
 Program Type: Sailing race/simulation
 Machines: IBM PC, Apple II
 Distributor: Howard W. Sams & Co. Inc.
 4300 W. 62nd St.
 Indianapolis, IN. 46268
 Price: \$29.95 diskette

System Requirements: Apple II—48K RAM, disk drive; IBM PC—128K RAM, disk drive

Poor Fair Good Excellent

Performance:
 Engrossment:
 Documentation:

Bermuda Race

A review
 by Steve Nelson
 HCM Staff

Two general types of computer games dominate the market today. One style relies on visual stimulation through colorful graphic displays. The other type uses less graphics, and relies more on text to create a world or a situation inside the mind of the game player. Games that use brilliant graphics are generally well-received and are usually quite entertaining—but a picture on the screen may never equal what you can envision in your mind.

Bermuda Race falls into the second category—a text-oriented adventure with interesting but not astounding graphics. It allows you to experience a technically-correct race, in great detail, without ever leaving your house. You do, however, have to have a good imagination in order to visualize the sailboat slicing through the deep blue water, the salty spray flying up over the bow, stinging your eyes and crusting up in your beard.

an overview of the race, and a description of the boat you will be commanding. It also describes how to access the different screens that allow you to change the speed or direction of the boat.

In *Bermuda Race*, the computer simulates a sailboat race from Newport, Rhode Island to Bermuda—a distance of 635 miles. The game can be played by one or two players; its object is to try and best the 1982 race record set by the sloop Nirvana, a boat of the same general class as the one you are commanding. Even if you can't beat the record, a fast time can make you eligible for the "Players Hall of Fame" (all the best scores are kept on disk). A successful race via computer can take 15 to 20 minutes.

When the status screen appears, the computer gives you only 20 seconds to set your course before the start of the race—so you should be ready to "read the wind" as soon as the starting gun

screen, which shows your compass heading, boat speed, wind speed, and direction (along with other useful information); and the chart screen, which shows you a map-like view of the course with a dotted line indicating your sailing direction. Other screens can be accessed when you wish to adjust your sails or the centerboard.

After each heading change, the boat's response is displayed on the status screen. It tells you how fast you are going, and whether the direction you chose is correct. If it isn't, your speed will decrease and your sails may start to luff (the boat will stall). If this happens, you must attempt to turn the boat away from the wind, fill your sails, and continue racing.

As the race progresses, you will have to adjust your course often because the wind's direction is constantly changing. The wind is, of course, different each race—there may be times when you can almost sail in a straight line to Bermuda, and then the very next race you may be meandering all over the Sargasso Sea.

Using the keyboard to maneuver your boat may seem like an unusual method for a sailing game, but the system works rather well. Just a press of a key causes an immediate change of screen, allowing you to set your course, adjust your sails or centerboard, or view your boat's progress as you race toward Bermuda. This swift feedback is one aspect that makes this game enjoyable, even without spectacular graphics.

The unexpected can happen in *Bermuda Race*, just as it can in a real race situation. You can lose satellite navigation, or your bilge pump may break, or some of your sail may blow out. And if you don't watch where you're going, you can even crash on the rocks and sink. This threat becomes especially dangerous as you near the end of the race, because the finish line is surrounded by reefs.

Visual Impact Missing

A real rookie, I was somewhat out of my element when I first tried *Bermuda Race*. But after some initial floundering, I soon began to feel like a seasoned veteran. For although this game is billed as a sailing simulation, it is more like a teaching aide, and, as mentioned earlier, an exercise in imagination. Once under way, you don't actually see your boat at all; the chart display is your only real indication of movement—a dotted line that slowly beeps across the screen. So, while the game is technically realistic, it lacks the visual impact inherent in other simulation games such as *Pole Position* [reviewed in this issue—Ed.].

For some people, the lack of great graphics in *Bermuda Race* may be disappointing. But for others, myself included, the game is challenging and quite realistic. And best of all, I don't have to worry about getting seasick.

"... there may be times when you can almost sail in a straight line to Bermuda, and then the very next race you may be meandering all over the Sargasso Sea."

But *Bermuda Race* is much more than just a game—it is a tutorial that helps you learn sailing terminology as well as the basics of good sailing. I found this aspect to be extremely helpful, as I knew nothing about sailing before playing this game. Even if you are an experienced sailor, I recommend that you go through all of the manual's prescribed steps before beginning; you will need to know such things as how much sail your boat can safely carry, and what the boat's proper luff angle is.

The documentation that comes with *Bermuda Race* is quite good. You get

sounds. With each change you make to the course, sails, or centerboard, the computer automatically calculates the new figures and adjusts the status screen accordingly.

Facing The Elements

Once under way, you must alternate between several screen displays to make course changes, adjust your sails, or observe your progress on the navigation chart (the only graphics screen available during game play). Two basic displays demand most of your attention in order to maintain a steady course and speed: the status



Name:	Star Trek	Buck Rogers
Program Type:	Arcade game	Arcade game
Machine:	TI-99/4A	TI-99/4A
Distributor:	Triton	Triton
	P.O. Box 8123	P.O. Box 8123
	San Francisco, CA 94128	San Francisco, CA 94128
Price:	\$29.95	\$29.95
System Requirements:	speech synthesizer and joysticks optional	joysticks and speech synthesizer optional
Performance:	Poor Fair Good Excellent	Poor Fair Good Excellent
Engrossment:	=====	=====
Documentation:	=====	=====

Both games are available through Sega Enterprises 360 N. Sepulveda, Suite 3000, El Segundo, CA 90245 for the following machines: Commodore-64 and the VIC-20 (\$34.95/\$39.95 cartridge), and for the Apple family including the IIc (\$39.95 diskette) and for the IBM PC and PCjr (\$39.95 diskette). Unfortunately, we were unable to obtain copies of these versions in time for this review.

Buck 'n' Kirk

A review of
BUCK ROGERS—PLANET OF ZOOM and STAR TREK
by Steve Nelson
HCM Staff

Over the years there have been several attempts to launch a new sci-fi series on prime-time television, but other than *Star Trek*, only *Buck Rogers in the 25th Century* and *Battlestar Galactica* were able to make any kind of impact. Now that these series are gone from the airwaves (except for reruns), sci-fi fans have been forced to turn to video arcades to see any interplanetary adventure outside of movies. But wait! Thanks to some like-minded video game programmers, the *Enterprise* and *Buck Rogers* once again are on the TV screen—this time with you at the helm.

Hoppers And Klingons?

Like most space-theme games, these two provide lots of alien spaceships and other neat things to blast, but they do it in different ways. On the *Enterprise*, the familiar front viewscreen is your window to deep space. Alien saucers, Klingon battle cruisers, and Nomad (who scatters bombs and mines in all directions) all attack you furiously as soon as you begin play. In *Buck Rogers*, you must pilot a spaceship over the three-dimensional surface of the planet Zoom, negotiate a series of tall electron posts (one touch is instant destruction), shoot your way through waves of creatures called hoppers—who, by the way, are very difficult to hit—then leave the planet and attempt to fight your way through flying saucers defending their mother ship. Whew!

Once you destroy the mother ship, you advance to the next level of play and begin the game again. As you move from one level to the next, the game's speed increases.

Buck Rogers is somewhat disappointing in terms of its scenario, as the game seems to rely more on its 3-D effect to keep you interested, rather than create anything more than a picture on the screen. The spaceship itself, complete with shadow on the planet

Zoom's surface, is an excellent model, but the hoppers and the flying saucers are not nearly as well depicted. Once I left the planet to attack the mother ship, I expected a spectacular battle with a huge spaceship; instead, the mother ship looked tacky, and it was much too easy to destroy.

The ship's response to your joystick is spectacular. You can bank to the right or left, and the planet's surface curves away beneath you. The closer to the ground you fly, the slower your ship moves, giving you more control as



"The ship's response to your joystick is spectacular. You can bank to the right or left, and the planet's surface curves away beneath you."



you maneuver between electron posts, but causing you to burn more fuel. When you fly higher, the opposite happens—your ship burns less fuel, but you lose some control because it is more difficult to negotiate the electron posts at faster speeds. And, at higher speeds, the hoppers and flying saucers attack much faster. You will have to have incredibly fast reflexes, or luck, or both, in order to play this game at full speed.

Watch For Energy Bolts

I also tested *Buck Rogers* on the Commodore 64. The graphics on the C-64 version aren't as crisp as those on the TI version, but the game is more difficult, with a few more obstacles. (For instance, when you advance to the second level of difficulty, the electron posts begin emitting bolts of energy that can destroy your ship unless you fly between them.)

Star Trek is a much less graphic game, but more challenging. You must keep your mind on more than just blasting the aliens. The *Enterprise* viewscreen is split into three sections: a view of the *Enterprise* out in space with the Klingons, flying saucers and Nomad around you; a close-up view of what is in front of you, along with your gun sight; and indicators that keep track of how many shields, photon torpedos, and warp drive units you have left. You must monitor all three, keeping in mind that your main worry is losing your shields.

Each time the *Enterprise* suffers a hit, it loses a shield. To replenish, you must rendezvous with a green starbase which will give you one shield, one photon torpedo, and one warp drive. Your starship comes armed with phasers and photon torpedos, energy shields, and can maneuver at either warp speed or on impulse power. Ten levels of play challenge you, each level continuing at a faster pace.

The only drawback I ran into while playing *Star Trek* was that after about five minutes of playing time my hand went numb from holding the joystick.

It's Them Or You

The documentation for both games is adequate, and you really don't need to know much to play either game: just blast them before they blast you.

Both of these games for the TI-99/4A have a speech option. If you have a speech synthesizer hooked up to your computer, you can hear a limited amount of speech.

I really liked the response of the starship in *Buck Rogers*, but the game itself tended to get boring after a few minutes of play, and the 3-D effect is very hard on the eyes. *Star Trek* is a more challenging game, but less exciting visually. While neither one of these games are as good as a *Star Trek* rerun, they are both fun to play, and can hold their own against the hordes of other star-struck video games now on the market.

Junior Addition:



Photo 1: The Tecmar jrCaptain peripheral comes with three manuals, a diskette of software programs, its own power source (the small transformer is not shown here), and the special screws to mount it to the PCjr.

A Review of the Tecmar jrCaptain Peripheral

by David G. Brader

HCM Staff

Did you buy an IBM PCjr with a disk drive and RAM? Have you been frustrated in your efforts to do serious work on the machine? Are you lusting for more computing power? Tecmar may just have the answer for you . . .

The Tecmar jrCaptain peripheral for the IBM PCjr sports two major attractions for the Junior owner—a battery-backed-up clock (to remember the date and time even when the PCjr power is off), and additional Random Access Memory (RAM). The amount of additional RAM depends on the type of memory ICs (integrated circuits) installed. The jrCaptain sample that we tested came equipped with the 64K-bit ICs (16 ICs for a total of 128K bytes of RAM) mounted in sockets. Tecmar's jrCaptain installation manual contains instructions for replacing these ICs with the new 256K-bit ICs, upgrading the peripheral to 512K bytes of RAM.



Photo 2: The jrCaptain mounts neatly to the side of the PCjr, just like the the IBM parallel printer port. In fact, it can perform the same functions.

The jrCaptain looks much the same as the IBM PCjr parallel printer module that fits on the side of the computer. Indeed, the jrCaptain module has a compatible parallel printer port as one of its elements. The jrCaptain does, however, have its own power supply (yes, you must find another socket for the AC power pack to plug into . . . With 128K of RAM, it uses less than one amp from its internal 5 volt supply).

More Than Just a Printer Port

The parallel printer port on the Tecmar jrCaptain is software compatible to the IBM version. The connector for the port (DB25 type) matches the IBM parallel printer cable used to connect to an IBM (or compatible) printer. While writing this review, we have our jrCaptain connected to an Epson MX80F/T. This port can also be used for other general-purpose parallel input or output. (Analog/digital or digital/analog converters anyone?)

By changing a jumper inside the Tecmar jrCaptain, the parallel port may be switched from device name "LPT1:" to device name "LPT2:." Although only one jrCaptain can be attached to Junior, the Tecmar product can be used with the IBM parallel port adapter to drive two printers from the one PCjr. "LPT1:" is the only device name available with the IBM parallel port ("LPT2:" cannot be accessed with IBM PCjr equipment, even with two IBM parallel ports attached). What does this mean?

You could connect a high-speed dot-matrix printer to the IBM port as "LPT1:" and a slower letter-quality printer to the jrCaptain port as "LPT2:." Because the default device name for the system printer is "LPT1:," all listings, reports, and rough drafts will automatically be printed on the high-speed printer. When you have a business letter or final draft document ready to print, it is simple to redirect the output to device "LPT2:."

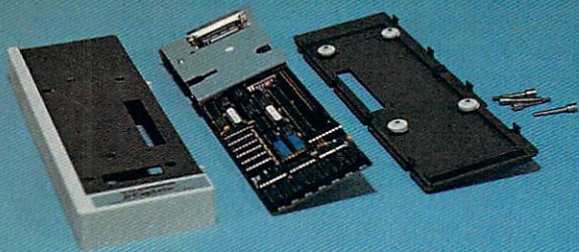


Photo 3: The jrCaptain case "pops" apart allowing access to the memory chips, selector switches, and clock battery.

Timing Is Everything

Do you usually skip entering the date and time when powering up your Junior? The jrCaptain will not only remove this hassle, but (with the aid of Tecmar software included in the package) will also automatically place the correct date and time on the screen every time you power up the system!

The jrCaptain's battery-powered digital calendar-clock can be used to trigger messages to the screen at preset times using a Tecmar software utility called CRON. When Junior is on, it can remind you that it is time to pick up the kids from school or to call Aunt Mary while the rates are lower—better than a simple alarm clock.

A major drawback to the clock feature is, however, its battery.

The battery must be changed by a local factory-trained technician, or the entire jrCaptain must be returned to Tecmar. Considering the fact that Tecmar printed extensive instructions in the installation manual for changing the memory ICs (a more complex task), requiring technical assistance to change the battery seems ludicrous!

128K More RAM, But . . .

Unfortunately, the additional memory doesn't mean that a PC application that was memory-bound on Junior will now work just fine. This is due to Junior's design, which uses the memory address space differently from the PC.

In the PC, the video memory is located on a video adapter board, separate from the system memory where our PC application program is located. This video memory, which is used to keep track of what is displayed on the screen, is addressed starting at location 704000 in the PC. The system memory, for application programs, can take up almost all of the address locations below the video memory—more than 640K bytes of contiguous memory if all PC memory expansion is installed. The PCjr hardware, however, places the video memory in the lower 128000 locations of system memory. So if your application program needs more

"The jrCaptain will not only remove the hassle . . . it will automatically place the correct date and time on the screen every time you power up the system!"

than about 100K bytes of memory, it will not work on the PCjr without major reprogramming—even with the jrCaptain attached.

So What Good Is The Additional Memory?

The answer: **RAMdisk**. With the aid of another special program supplied with the jrCaptain (called MEMDISK), the additional memory can be configured to function like a disk drive to DOS. Most of the DOS commands such as DIR, COPY, DEL, and TYPE work just the same with **RAMdisk** as they do with your built-in real disk drive (only noiselessly and with far greater speed). DANGER! Any data that you store on the **RAMdisk** will be lost when you turn off the power to the PCjr. If you are storing data that you want to keep—copy it back onto a real diskette before shutting down for the night.

It is wiser to use **RAMdisk** to hold a copy of the programs you are currently using, and to put all of your data on a diskette in the real drive. Imagine putting a full word processor including spelling checker on the **RAMdisk** during a writing session—with the documents stored on a data diskette in drive A . . . Sorry, there isn't enough **RAMdisk** memory space with the 128K version of the jrCaptain to do this. For example, the new **IBM Writing Assistant** package, including the spelling checker, fills one diskette. Our suggestion: Buy the 512K version of the jrCaptain if you are going to use heavy-

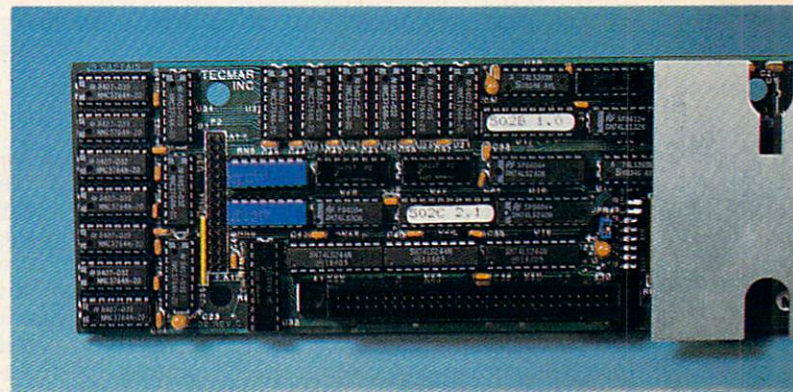


Photo 4: All 64K RAM chips are socketed for easy replacement when the price and availability improve on the 256K RAM chips (increasing on-board memory to 512K-bytes).

Name:	Tecmar jrCaptain
Description:	jrCaptain with 128K of RAM memory, clock, printer port peripheral with soft ware and manuals, 1 year limited warranty
Machine:	IBM PCjr
Distributor:	TECMAR INCORPORATED Personal Computer Products Division 6225 Cochran Road Solon (Cleveland), Ohio 44139 (216) 349-0600
Price:	With Memory, \$395. Without Memory, \$235.
System Requirements:	IBM PCjr with 128K RAM and the IBM disk drive
	Poor Fair Good Excellent
Performance:	██████████
Ease of Set-up:	██████████
Documentation:	██████████

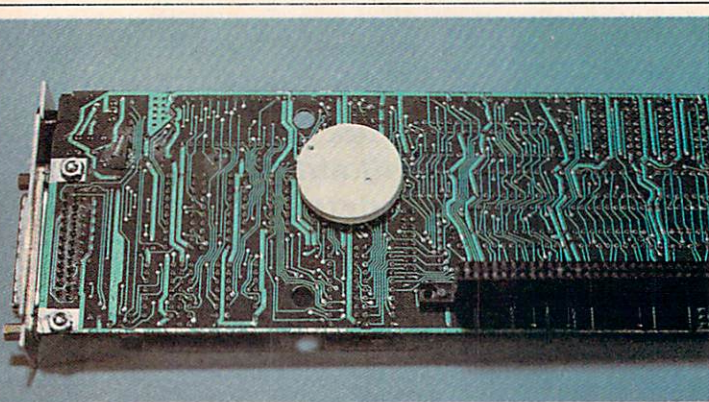


Photo 5: The round disk in the center of the back of the jrCaptain board is special battery for the on-board clock/calendar. It must be replaced by a factory-trained technician.

duty business packages. [At the present time, due to the cost of 256K RAM chips, Tecmar is offering an add-on product called the jrCadet to mount on the jrCaptain. The jrCadet uses 64K RAM chips to add 384K of addressable memory to the 128K jrCaptain. The suggested retail price of the jrCadet with 384K is \$595.—Ed.]

Printer Buffer

Another immediate use for the jrCaptain's memory is to temporarily hold documents or listings being sent to the parallel printer. This "spooling" of data to be printed means that you can continue working at the keyboard while the printer is doing its thing—nice feature, especially if you are doing production typing. (Don't laugh, many people are able to type fast on the PCjr keyboard. Plus, there are now several office-quality keyboards available.) The software to configure the memory for this purpose is also supplied by Tecmar with the jrCaptain.

Basic Usage Of The Expansion Memory

PCjr Cartridge BASIC programmers can use the additional memory with many of their applications. The DEF SEG command is used to locate BASIC's data segment pointer anywhere in the addressable memory range. Then BLOAD, BSAVE, and CALL may be used to load, save, and call assembly language programs from the addressed segment of memory.

If you have defined a RAMdisk in the jrCaptain, you can use BLOAD and BSAVE to do rapid video memory paging—quickly switching many screen images opens a new area of animation on the PCjr.

Installing The Sidekick

Installing the jrCaptain is a two-step process. Attaching the unit to the right side of the PCjr requires a medium-sized flat-blade screwdriver and a clear workspace. The Tecmar manual is very clear on the steps involved to accomplish this task—it takes less than 15 minutes. The second stage of installation deals with the Disk Operating System. A special program (yes, it too is provided) called CONPCJR must first be run when the system is initialized or reset. Complete the installation by copying the program to your DOS 2.1 "boot" diskette and creating an AUTOEXEC.BAT file that calls the CONPCJR program first. To find out how to create batch files, check your manual or read the *Home Computer Magazine* Tech Note for IBM in the August, 1984 issue.

The CONPCJR program has several options that can change the default hardware configuration as seen by DOS. Of primary interest is the option that lets you

"This 'spooling' of data to be printed means that you can continue working at the keyboard while the printer is doing its thing . . ."

designate additional disk drives on-line. By placing CONPCJR -D2 in the first line in your AUTOEXEC.BAT file, disk drives A and B are recognized by the system. By making the next line in the file MEMDISK B:, all disk drive B-addressed functions are directed to the RAMdisk.

For those of you who have already read the article in this issue describing the addition of a second *real* disk drive to Junior: yes, this software trick works with those hardware modifications, too. If you have made the modifications to your system as described in that article and you also wish to use the jrCaptain RAMdisk, the first two lines in AUTOEXEC.BAT should be entered as: CONPCJR -D3 and MEMDISK C: -APL128. Now the two real drives will be device addresses A and B while the RAMdisk becomes device address C.

Other Jewels From The Treasure Chest

The diskette that comes with the jrCaptain contains all the little program gems that we have covered, plus a bunch more. Some are rather frivolous—a banner letter-printing program or Tic-Tac-Toe—but others are very useful. My favorite has to be the MEMDISK program mentioned above because of the speed and flexibility that a RAMdisk adds to Junior.

Tecmar calls the diskette of programs their "Treasure Chest of Software." Although this is definitely advertising hype, the software certainly adds value to the jrCaptain package.

How About the Documentation?

Three manuals come with the unit: *Treasure Chest User's Guide*, *Treasure Chest Technical Reference*, and *jrCaptain Installation Manual*. The first two manuals cover all software programs supplied except the CONPCJR routine, which is covered in the third. This third manual is the one to read carefully prior to attempting to install or use the jrCaptain. It also contains much data of use to programmers, regarding hardware addressing and control of the unit. All three manuals are colorless and simple, but they have all the information a jrCaptain owner is likely to desire.

So, Should I Buy It Or Not?

If you have not purchased the IBM parallel printer port adapter and you feel the features described above fit your needs (and budget), yes—buy it. It is a well-built handsome unit and comes with a one-year limited warranty. Just remember, however, that in light of the recent across-the-board IBM price decrease, plus the imminent (as we go to press) introduction of a more powerful and flexible Junior, the cost of the Tecmar accessories should be weighed very carefully. **HCM**



RAZZLE DAZZLE

by **W.K. Balthrop**
HCM Staff

**Unlock some of the TI-99/4A's
graphics potential with this
short graphics routine.**

If you're not yet convinced of the 99/4A's graphics capabilities, then you will change your mind after watching this next program light up your screen. With Extended BASIC, the TI computer is capable of performing startling effects with very little effort. This short program, *Earth Sprites*, is proof. With *Earth Sprites*, a small figure of the Earth rotates across a large field of stars. (The photo at the top of the page is an enlargement of this screen.)

Movies are made up of a lot of still frame photos which, when shown together rapidly, give the illusion of motion. Computers can do this too, but the procedure has always required a large amount of memory and a very powerful processor. All you really need are a couple of sprites and a little imagination.

In this program, we have re-defined the characters in the computer. Sprites get their shape from the shapes of these characters, but we can also tell a sprite to, at any time, start getting its shape from a different set of characters. This concept is what this program is based on. We have created a series of different shapes for a sprite, which when shown in succession rapidly, gives the illusion of motion. Not just motion across the screen, but of the planet Earth, with all of its continents, rotating on an axis through space.

***"Movies are made up of a lot of still frame
photos which, when shown together
rapidly, give the illusion of motion.
Computers can do this too . . . all
you really need are a couple of
sprites and a little imagination."***

Line 170 of *Earth Sprites* reads the graphics patterns from the DATA statements, and assigns them to characters 40 through 79. The sprite is set to a magnification factor of 4 in line 180, which means that each sprite will use four characters for its shape. The two sprites for this animation are then placed on the

color cyan, which represents the oceans. This color will remain behind sprite #1, which is used for the continents. Line 190 changes the shape of character 84, which is the letter T, to a single point. It then places 100 of these characters on the screen randomly to create a star field. Line 200 puts in motion across the screen the two sprites which make up the earth.

Line 210 is the key to the whole operation. The Extended BASIC command **CALL PATTERN** lets us reassign a new shape to an existing sprite. By cycling through the patterns it seems as if the Earth is actually rotating on the screen. The rest of the program consists of the graphics data for the sprite shapes.

Now that you see how easy it is to create graphics on the TI home computer with your own character patterns and the use of sprites, you will want to experiment with your own ideas. To get started, try enhancing the *Earth Sprites* program. The rotation is still a little jerky because only 9 different shapes are used, so try increasing the resolution of the Earth's rotation by using more shapes, for example.

TI-99/4A

```

100 REM *****
110 REM * EARTH SPRITES *
120 REM *****
130 REM BY WILLIAM K BALTHROP
140 REM HOME COMPUTER MAGAZINE
150 REM VERSION 4.4.1
160 REM TI EXTENDED BASIC
170 CALL CLEAR :: FOR Z=1 TO 10 :: READ
  AS :: CALL CHAR((Z-1)*4+40,AS):: N
  EXT Z
180 CALL SCREEN(2):: CALL MAGNIFY(4)::
  CALL SPRITE(#28,40,8,100,100,#1,48,
  3,100,100)
190 CALL CHAR(84,"01"):: CALL COLOR(7,1
  6,2):: FOR Z=1 TO 100 :: CALL HCHAR
  (RND*23+1,RND*27+3,84):: NEXT Z
200 CALL MOTION(#1,0,2,#28,0,2)
210 FOR Z=76 TO 48 STEP -4 :: CALL PATT
  ERN(#1,Z):: FOR TD=1 TO 25 :: NEXT
  TD :: NEXT Z :: GOTO 210
220 DATA 030F1F3F7F7FFFFF7F7F7F3F1F0F
  03C0F0F8FCFEFEFEFEFEFEFEFEFEFEFE
230 DATA 00060F07070703000000101000000
  00000E8E8C0C0808040E0F0F0E06010
240 DATA 00081F3F3E3C1C02070F0F07030000
  0000404006060302030716120000080
250 DATA 0000003A7A70F0E0E010387C3C1808
  00000081C3E3E1F111E3F3E1E0C08
260 DATA 00000105101800000814160200000
  00003078FCFEFEFEFEFEFEFEFEFEFEFE
270 DATA 000103070F0F0704070F0F03030301
  0000F0F8F8FCFEFEFEFEFEFEFEFEFEFE
280 DATA 000F1F3F7F7F3F233C7F7E1E1C1C08
  00009C0C0E0E0C0C0000001C14
290 DATA 000C1E3E7F7FFFE1EE0F8707020000
  0000F01804060603000000E0A0
300 DATA 000710307878F0F0000C00705000000
  0000F0F83C3E3E1C0404020700E0E04
310 DATA 000F0701414180800003828000000
  000080F8F8F0F0E02010387C7C3818

```

HCM

SST BASIC COMPILER SYSTEM

A review
by Tom Green
HCM Staff

SST Software's new *Expanded BASIC Compiler System* enhances TI BASIC by giving the BASIC programmer an expanded instruction set, while greatly increasing program execution speed. This, however, doesn't come without a price: With the *SST System*, programs have to be re-written in a highly structured format, and carefully edited and compiled—resulting in greater program development time. As is the case with nearly all microcomputers, the TI-99/4A's resident BASIC is an "interpreted" language. When the Central Processing Unit (CPU) RUNs a BASIC program, it's like trying to read a book in a foreign language—looking up every word in a dictionary to find out what it means—instead of reading a translation. A BASIC compiler translates the program into the CPU's "native tongue" (machine language) *before* the program RUNs—so that it can RUN many times faster.

SST Expanded Compiler

Three principal steps are involved when using the *SST Compiler System*.

The flow chart in Figure 1 demonstrates the entire process: **(1)** Create and edit an SST BASIC program by modifying an existing TI BASIC program, or writing a new one to conform to SST BASIC

format. RUN-test it using one of the two editor programs and prepare it for compiling. **(2)** Compile and SAVE the program to disk using the compiler program. **(3)** LOAD and RUN the program using one of the two loader programs.

Last year SST Software broke the Sound barrier with a cassette-based compiler system for the 99/4A. Now the expanded system goes to Mach 2.

Preparing To Compile

The first step is to prepare your program in SST format. Figure 2 (taken directly from the SST manual) shows the outcome of such a conversion. The "at" symbol (@) after each of the variable names designates the variable as an integer. For the inexperienced programmer, this next requirement will be the most difficult part of using the *SST System*: Any variable or constant you wish to use in your program must be defined at the beginning as a one- or two-character variable name. No actual numbers are allowed in the body of the program, so if you want to add two numbers together, they must be defined. These requirements defeat the easy (although unstructured) features that make BASIC such an attractive language. This drawback, however, is somewhat offset by the fact that novices will be introduced to structured programming in the familiar BASIC environment.

Still, we do feel that defining constants (something not required in even a highly-structured language like Pascal) is a little hard to justify. Once you've written the SST BASIC version of your program, it is combined with one of the two editor programs. The

BASIC Editor requires either the Mini-Memory or Editor/Assembler Module, and the

Editor/ex requires

the TI Extended BASIC module.

The major advantage of the *Editor/ex* program is that it allows use of the Merge option to combine a program with the editor. To use the *BASIC Editor* your program must be keyed in with the *Editor* in memory. Next, RUN-test your programs from

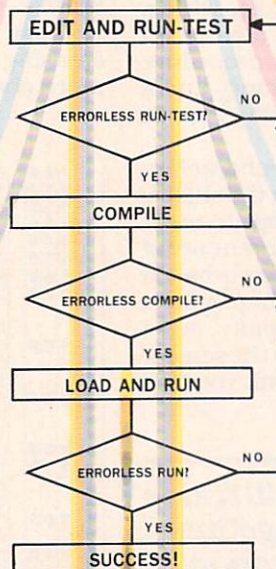


Figure 1



either editor. Every instruction is checked for correct syntax, and errors are noted so that modifications are relatively easy to locate. You modify errors found by the editor by using regular TI BASIC or Extended BASIC commands in the conventional manner. The *Expanded SST Compiler System* also includes many new commands. During the RUN-test these SST BASIC commands are printed to the screen as a means of tracking your program.

Compiling, LOADING and RUNing

Once you have completed the Editor phase of debugging, you RUN the editor to create your source file, and you're ready to begin compiling. Program debug-

ging is a trial-and-error process between the *Editor* and *Compiler* programs. The *Compiler* will catch errors that get by the Editor. Yet, it is not 100% effective—a program may successfully pass through the *Compiler* and not RUN properly, leaving the developer with "run-time" errors. This "hit-and-miss" meth-

Name: SST Expanded BASIC Compiler System
 Program Type: Utility
 Machines: TI-99/4 or TI-99/4A
 Distributor: SST Software, Inc.
 P.O. Box 26
 Cedarburg, WI 53012
 Price: \$95, Disk; \$85 User Group fee; \$50 with proof of purchase of SST BASIC Compiler System.

System Requirements: Memory Expansion, Disk Drive, and Editor/Assembler or Mini-Memory Command Module.

Poor Fair Good Excellent

Performance: ██████████
 Ease of Use: ██████████
 Documentation: ██████████

od of editing can be frustrating, and is always time-consuming. For instance, the process of editing, compiling, and running the program in Figure 1 took approximately 15 minutes to complete. You can see that a longer, more complicated program would require much more time and patience to develop. The increased performance of the program, however, makes up for the frustration that arises from implementing these procedures. (Try writing machine code from scratch sometime—you'll appreciate the power of the *Compiler*.)

When you finish compiling, the *Loader* program then accesses code generated and saved to disk by the *Compiler*, and places it into memory for execution. You may link and RUN several compiled programs using the *Loader*. There is also a Fast Load option that prepares programs so they don't require the *Loader* each time they RUN. Once prepared, the programs RUN directly using the commands CALL INIT, CALL LOAD, and CALL LINK, all of which are available from TI BASIC with either the Mini-Memory or Editor/Assembler modules. This creates an executable file that can be RUN on any TI computer with Memory Expansion and the particular command module used during the Fast Load procedure.

Pixels Accessable

One of the more powerful commands (unique to SST BASIC) is the hi-resolution screen graphics mode, specified as PLOTMODE (Bit-Map Mode). This function allows access to each of more than 49,000 pixels (dots) on the monitor screen. You have the ability to place or remove a character anywhere on the screen, designate foreground and background color, or apply the same parameters to individual pixels. This is the first time such access has been available to programmers on the TI-99/4A, outside of assembly language or TI Forth.

The *SST BASIC Compiler* can only use the TI Extended BASIC module when running the Editor/ex program. But no matter which editor you use, the SPRITEMODE commands give you access to sprites. Generally, these commands are similar to those available in TI Extended BASIC, with only minor variations.

"A BASIC compiler translates the program into the CPU's "native tongue" (machine language) before the program RUNs—so that it can RUN many times faster."

These new enhancements, however, do not come without certain limitations. Because of differences in the TI-99/4A's video-processor memory modes, SPRITE and PLOT modes are not compatible in the same program. Either command can be CALLED only once from a program, and both have further restrictions on accessing certain codes.

Expanded SST Is Expandable

Another plus of the *SST System* is access to user-defined routines with the CALL USERA . . . E command. Up to 6 assembly language or compiled BASIC routines may be included in the *Loader* program as a sort of user-defined "library" of functions. Thus, as you become more adept at programming your TI-99/4A on an assembly language level, you can customize the *SST System* to include your routines.

Documentation

The first word that comes to mind when describing the user's manual of the *SST Expanded BASIC Compiler System* is "beefy." Guidelines for program development are thoroughly outlined in step-by-step fashion, and include remedies for common pitfalls in the procedures. This product however, is not for the casual programmer. But, those who are willing to invest the time to learn the *SST System* will be rewarded with programs that run many times faster than ordinary BASIC programs.

HCM

The next program is similar to one which appeared in the March, 1980 BYTE Magazine. It is a program designed to generate prime numbers, and is often used as a benchmark. The program was originally run in Basic on the TRS-80 computer. It took 7 hours, 12 minutes to check the first 10,000 integers for prime numbers. The program written here checks only the first 1,000 integers.

```
100 LET L@=6
110 LET E@=1
120 LET M@=1000
130 LET Z@=5
140 LET A@=1
150 LET N@=10
160 LET D@=1
170 LET B@=2
180 LET C@=2
190 FOR A@=L@ TO M@
200 A@=A@+E@
210 D@=A@/C@
220 FOR Z@=B@ TO D@
230 Z@=Z@+E@
235 REM FOR T.I. BASIC LINE 240 SHOULD BE
236 REM N@=INT(A@/Z@)
240 N@=A@/Z@
250 N@=N@*Z@
260 N@=A@-N@
270 IF N@ <=0 THEN 300
280 NEXT Z@
290 PRINT A@
300 NEXT A@
310 STOP

TIME, BASIC: 1535 seconds
TIME, SST COMPILER: 18 seconds
```

If line 120 is changed from M@=1000 to M@=10000, the program will check the first 10,000 integers. The *SST EXPANDED COMPILER* completes the program in 11 minutes, 20 seconds. In T.I. BASIC, it took 4 hours and 15 minutes to check the first 5500 integers. The *SST EXPANDED COMPILER* took 4 minutes to check the first 5500 integers.

Figure 2

Group Grapevine

News, information and upcoming events of home computer users groups around the world.

Looking to join a users group, exchange newsletters or software, increase your users group's membership or pep up your next meeting's agenda? For the latest users group news, put your ear to the Group Grapevine. And if you have a message to put out to other groups, if you are starting a new group, or have an interesting item to share, send a note or picture—or better yet, a group newsletter—to the Users Group Editor, Home Computer Magazine, 1500 Valley River Drive, Suite 250, Eugene, OR 97401, (503) 485-8796.



Need help in forming a Commodore user group? If so, call the **New Mexico Commodore Users Group** in Albuquerque. They will send copies of *On-Line* (their newsletter), their bylaws, and any other pertinent information upon request. They will also answer questions regarding becoming an affiliate of NMCUG. This is just one of many aspects of this group which offers monthly meetings, hardware and software reviews, classes, a public-domain software library, and a newsletter. Individual membership dues are \$12 per year, \$15 for family and out-of-state individuals. If you would like to volunteer some time helping the group, you can earn credit toward the membership fee. For more information, write: NMCUG, P.O. Box 37127, Albuquerque, NM. 87176.

The **San Luis Obispo Commodore Computer Club** was formed a year ago with just a handful of interested people, and today they count a membership of more than 130 members. The group has a large club library containing programs for the VIC-20 and Commodore 64, produces a newsletter, and operates a bulletin board service. According to Alan Heminger, outgoing president of the group, the club meetings are well-attended and informative. The club even has its own computer and disk drives, which allows them to provide many of their own services. If you live in the San Luis Obispo area and are interested in finding out more about this club, contact: Gary Bissell, 1766 Ninth St., Los Osos, CA. 93402, (805) 544-2924.

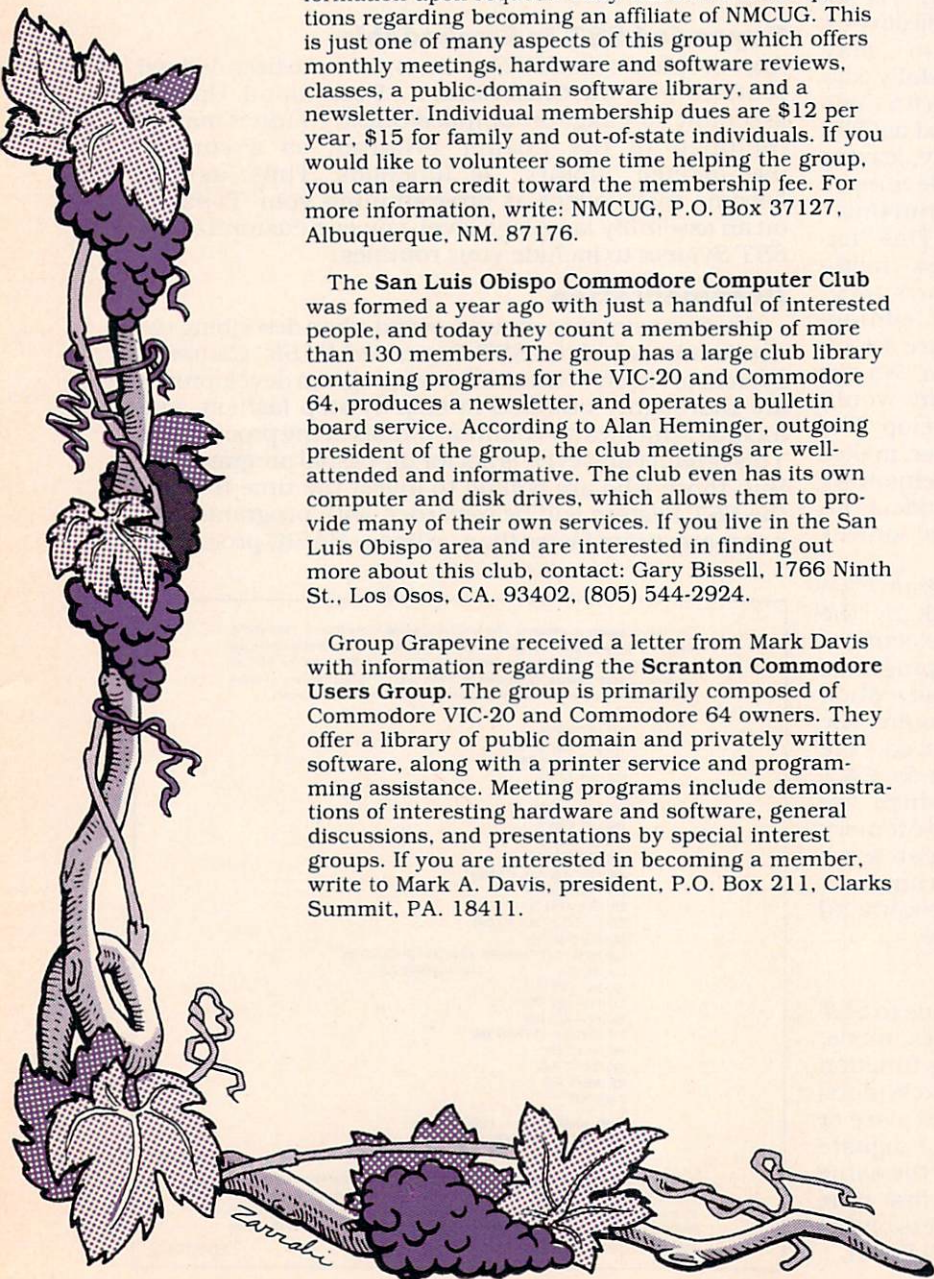
Group Grapevine received a letter from Mark Davis with information regarding the **Scranton Commodore Users Group**. The group is primarily composed of Commodore VIC-20 and Commodore 64 owners. They offer a library of public domain and privately written software, along with a printer service and programming assistance. Meeting programs include demonstrations of interesting hardware and software, general discussions, and presentations by special interest groups. If you are interested in becoming a member, write to Mark A. Davis, president, P.O. Box 211, Clarks Summit, PA. 18411.



David Frye, a 16-year-old high school sophomore from Alden, New York, has initiated a TI-99/4A users group at his high school. The idea seems to have caught on well—new members from all over the Clarence School District are joining up, and so far no one is over 17. The group's library consists of 78 BASIC and 31 Extended BASIC programs. They are now looking for other users groups run by kids with which to correspond and exchange ideas. If you are interested in becoming a member, contact David Frye, 1132 Boncliff Drive, Alden, NY. 14004.

Congratulations! **TISHUG** (Texas Instruments Sydney Home computer User's Group) in Australia celebrated their third birthday in May. **TISHUG** founder, Shane Andersen, has watched the club grow to an unbelievable 700 members since that first meeting in May 1981. The group's dream of establishing a bulletin board service has finally come true. **TISHUG**'s bulletin board is for downloading software, up-to-date news and views, electronic mail, and programming hints. This electronic bulletin board will soon become the very first bulletin board in the Southern Hemisphere to have clear, spoken text as displayed on the screen. The group is also making plans to provide low-cost RS232 interfaces and modems so that as many club members as possible can take advantage of the bulletin board. In addition, each month **TISHUG** conducts a software competition open to everyone. Prizes are awarded for the best in the following divisions: Best Award of the Month; Junior Award of the Month; and Rookies Award of the Month. There must be many TI-99/4A users "down under" who would like to join up and share their talents. Upon glancing through the Sydney News Digest (**TISHUG**'s newsletter) one notes that there are several active regional groups affiliated with **TISHUG** in Sydney—Blaxland, Newcastle, Illawarra, Mosman, Nepean, Bulkem Hills, and Marrickville/Ashfield. If you are interested in becoming a member, contact: John Robinson, P.O. Box 149, Pennant Hills, NSW, Australia 2120.

Group Grapevine received its first issue of **Channel 99 User Group's** newsletter from Tom Arnold, coordinator. This group of 180 members—several of whom live as far away as 200 miles—hails from Hamilton, Ontario (Canada). They are interested in exchanging newsletters and user-written programs with other groups. Channel 99 recently held a *Munchman* competition at their monthly meeting, and they are currently planning for a lending library consisting of computer-related books. Also in the planning stage is a programming tutorial to be held at the meetings and augmented with lessons and problems in their newsletter. If you think you live too far away from a users group to become a member, this is the group to contact. Write: Tom Arnold, 77 Lavina Crescent, Hamilton, Ontario L9C 5S8.



Group Grapevine just received some unhappy news from Ed York of the **Cin-Day Users' Group** in Cincinnati. In Vol. 4, Issue 2 of Home Computer Magazine, Ed announced that the group was preparing to sponsor the first annual Midwest Computer Fest in the fall. According to Ed, the group's new officers feel that it is no longer feasible for the **Cin-Day Users Group** to sponsor this project. Therefore, the first Midwest Computer Fest has regretfully been cancelled. If you have any questions, contact Ed York, P.O. Box 519, West Chester, OH. 45069, (513) 777-0010.

A small group of TI-99/4A owners in Dimona, Israel is very interested in joining with another user group. Kitlaru Beny writes Group Grapevine that the group is extremely hungry for all kinds of information regarding software, peripherals, etc. If you would like to contact this faraway group, write: Kitlaru Beny, P.O. Box 565, Dimona 86104, Israel.



Wow! Here's music to Group Grapevine's ears! An IBM PCjr computer club is forming out there in the Washington, DC area. The **Capital Area IBM PCjr Computer Club** is the place to learn more about hardware and software for Junior, including: keyboards, memory expansion, magazines, games, and business software, as well as IBM compatibility, DOS 2.10, BASIC programming, home/office applications, telecommunications, word processing, spreadsheets, and much more. The club is open to all persons who have PCjrs, or who are merely interested in computers. The club is geared toward less technically-experienced people who want to know more about home computing. For more information, contact: Lowell Denning, 12611 Beechfern Lane, Bowie, MD. 20715, (301) 262-8275 or (202) 566-4801.

The **Portland IBM PC Club** of Portland, Oregon has a very active group, with special interest groups in the following areas: spreadsheet, business applications, C language, word processing, data base management, novice, and hackers. The club's efforts to establish a bulletin board have finally come to fruition and it should be in place and operational by the time Group Grapevine goes to press. The Portland club has members who are willing to rent their personal computers for a fee set by each individual member. If you live in the Portland area and would like more information about this very active and interesting group, contact: Rich Rohde, P.O. Box 2068, Beaverton, OR. 97075, (503) 620-6862.

According to the **Quad-City Personal Computer User's Group's** newsletter, this group provides a forum for the exchange of information by PC users, a software library, outstanding public-domain and user-supported programs, educational programs, and group purchasing of hardware and software. If you live in the Bettendorf, Iowa area and are interested in becoming a member, contact: John Dannenfeldt, P.O. Box 464, Bettendorf, IA. 52622, (319) 752-0245.

Group Grapevine just received a news release from IBM with information regarding support to PC user groups. User groups can now receive support and information directly from IBM's Entry Systems Division. The support department offers a newsletter distributed on a diskette that includes selected technical articles from user group publications,

items of general interest to user groups, and information about recent IBM PC product announcements. The division also offers a bulletin board to assist user groups in communicating with other groups and to provide general product information. It also includes answers to questions frequently asked on the support department phone line. Officers of PC user groups can call the support department for general information about user group activities, IBM PC products, and forming a new user group. If your group would like to register with IBM, write to Gene Barlow, IBM PC User Group Support (2900), P.O. Box 3022, Boca Raton, FL. 33432.



A group focusing on practical applications for Macintosh has taken root in New York City. **New York MacUsers' Group** meets monthly and anyone can become a member—not just New Yorkers. The meetings include software exchanges, speakers, demonstrations, and many more topics of interest, as dictated by the membership's interests. If you're a new Mac owner and would like more information, contact: Cheryl Sandler, NYMUG, P.O. Box 6686, Yorkville Station, New York, NY. 10128, (212) 535-1943.

Group Grapevine just had a phone conversation with Dave Hoffman, founder of the **Wenatchee Valley Apple User Group**, established in September, 1981. The group consists of 67 members from many professional fields, including engineering, medicine, law, and education. The group library consists of approximately 1,000 titles, which can be sent out by mail to any interested club member—as well as other users groups. They have donated about one-half of the library titles to the regional library. This group also helped an Apple group in the Tri-Cities area get off the ground, and because there are more and more Macintosh users becoming interested in the club, they are considering forming a Mac group. If you are interested in this group or its possible offshoot and live in the Wenatchee area, contact: Dave Hoffman, 535 Highland Drive, Wenatchee, WA. 98801, (509) 662-7317.

Some news comes from a relatively new Apple group, the **Roseville Apple Users Group** in Roseville, CA. This group has been in existence for only four months and has 55 members. After talking with Otto Haiungs, president of the group, it sounds as if they have plans for several community projects, as well as in-house club projects. Activities that include the handicapped in the community is one idea they plan to pursue, and any help from people knowledgeable in this area would be greatly appreciated. The group meets at the Roseville Main Library the second and fourth Tuesday of each month (the meeting on the fourth Tuesday is a Special Interest Group meeting). Currently, the group has SIGs covering the Macintosh, business, and games. Membership dues are \$12 per year, and include a subscription to the Roseville Apple Core Bulletin, the group's monthly newsletter. If you are interested in becoming a member, or have information regarding computers and the handicapped, please contact: Otto Haiungs, P.O. Box 1377, Roseville, CA. 95651, (916) 783-0364.

Okay Apple users, it's tree-shaking time! We really would like to hear from all of you Apple groups via letter or newsletter (or even a phone call), so we can keep up with all the exciting things that are blossoming out there in Apple land. Some Macintosh seedlings to plant amongst the other Apple varieties in this orchard would be extremely welcome.

HCM

Imagine, 360K bytes of diskette storage in the TI Expansion Box—magic made possible by installing two, half-height, dual-sided disk drives in the TI box.

2 for TI

A review
by **Steve Nelson**
HCM Staff

CompuAdd Corporation is marketing two Shugart SA455s as a dual-disk drive that slips nicely into your peripheral expansion box with no modifications. Although two disk drive units would overtax the power supply, they claim to have solved this problem by utilizing drives that require half the normal power. So while you are in fact running two drive motors, they require approximately the same amount of power as one regular drive motor—with no loss of performance. Because these items are sold separately by the distributor, we felt our readers would appreciate some added documentation explaining installation procedures.

Getting Ready

When your dual-disk drive arrives, the first thing you will need to do is take the dual-disk drive unit out of the box and make certain that all the necessary parts are there (see Photo 1). Next, you must turn off the power to your peripheral expansion box and wait a couple of minutes before removing the old disk drive unit. The only tool required is a Phillips screwdriver.

Removing The Old Disk Drive Unit

First, remove the screws holding the old disk drive in place. I found it helpful to first disconnect the signal cable and remove the disk controller card from the peripheral expansion box before removing the drive unit. Gently slide the disk drive unit out until the back is visible. Detach the power cable, the ribbon cable, and slide the unit free. Once the disk drive unit is removed, you are ready to install the Shugart dual-disk drive assembly in its place.

Installing The Shugart

You must be aware of two important things when installing the disk drive system. First, be sure you understand which drive is Drive 1 and which is Drive 2. Second, determine whether a special resistor pack is in its proper position. The manual (which is a bit on the sparse side and




could be more specific) explains what position the resistor pack should be in according to the number and type of drives you are using.

Start by opening the package of cables (34-pin signal cable and power cable). Refer to the instructions in the

Product: Shugart 455 Disk Drive
Machine: TI-99/4A
Distributor: CompuAdd Corp.
13010 Research Blvd. #101
Austin, TX. 78750
Price: \$189 per drive
\$15 2-drive cable kit

System Requirements: TI Disk Manager 2
command cartridge, TI disk controller card,
TI Peripheral Expansion System

Poor Fair Good Excellent

Performance: 
Ease of Set-Up: 
Documentation: 



"While you are in fact running two drive motors, they require approximately the same amount of power as one regular drive motor—with no loss of performance."

documentation, which show you the proper set-up of the jumper plugs on the jumper block at the back of the dual-disk drive unit. Drive 1 requires the plug to be in position 1. Drive 2 requires the plug to be in position 2 (see Photo 2). Plug the two ends of the 34-pin cable into the connectors on the drive units, and do the same with the power cable. Because there are two drives, both units need to be connected (see Photo 3). Place the Shugart disk drive unit in front of the opening in the peripheral expansion box and thread the 34-pin cable back through this opening, connect it to the disk controller card, and re-install the card into its proper slot. Now connect the power cable to the supplied adaptor and to the drives. Gently slide the Shugart disk drive assembly back into the peripheral expansion box until it is flush with the front of the box. (Note: the new unit cannot be fastened in place—the screw holes do not align.) Now you are ready to turn on the power and test it.

How It Checked Out

After making certain that everything was plugged in correctly, I turned on the power and tried to load a program from Drive 1. My first attempt failed; neither indicator light came on, and the program didn't load. Needless to say, everything *wasn't* connected properly—I had the 34-pin ribbon cables reversed.

**"One important thing isn't mentioned in the documentation:
The dual drive system
can read both sides of a disk."**

As I mentioned earlier, the documentation is not as specific as it could be. I turned off the power, switched the cables, and tried again. This time the program loaded fine. Next, I tried to save the program to DSK2 (the second drive). The program saved perfectly. Since then, I have used the dual-disk system every day for the past few weeks with absolutely no problems. It works great, and saves me a lot of time.

The dual-disk drive unit fits nicely inside the TI peripheral expansion box, and CompuAdd recommends that you leave the units loose in there. However, you may want to slide a thin piece of cardboard between the drive units and the side of the box to give it a "friction" fit. I found that unless you plan on regularly moving the peripheral expansion box, there is no reason to worry about securely bolting in the drives—a piece of cardboard works fine.

One important thing isn't mentioned in the documentation: The dual drive system can read *both* sides of a disk. However, if you are using a *Disk Manager 1* command module, you will be able to access only *one* side of a disk (for a total of 180K bytes of storage between the two drives). In order to access both sides of a disk, you will need the *Disk Manager 2* command

Photo 1:

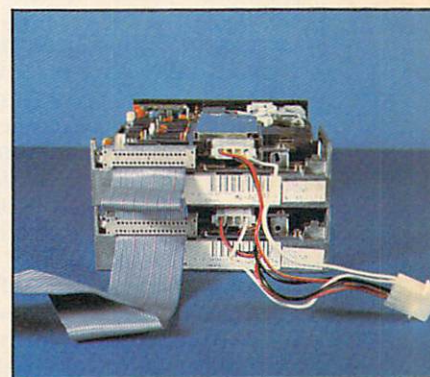
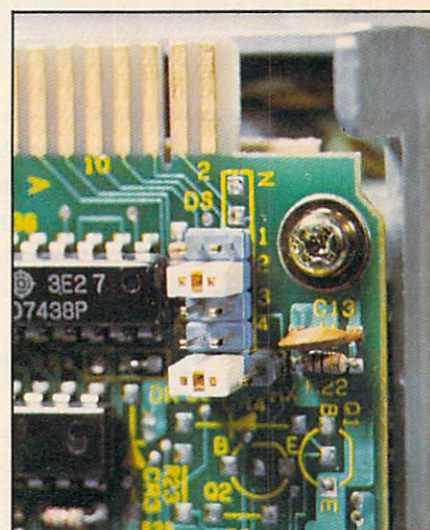
Complete kit, showing both drives, power cable and signal cable unassembled.

Photo 2:

Close up of Drive 2's jumper block with jumper plug shown in position 2.

Photo 3:

Rear of unit showing power cable and signal cable properly connected.



module. I recently called Texas Instruments, and they informed me that *Disk Manager 2* is still available. If you call them at 1-800-842-2737, they will refer you to a distributor who can get one for you.

A Few Complaints

The documentation should be more thorough and specific to be of any real assistance, but because the system is so easy to install, I'll say that the documentation is marginally adequate. CompuAdd has informed me that they will be amending their documentation to discuss the *Disk Manager* cartridges. Hopefully, they will make the installation procedure a little clearer at the same time. *CompuAdd* does provide you with a toll-free number 1-800-531-5475 for assistance if you have problems installing the drives. There is one other thing: Even though the dual-disk drive unit fits snugly, you really ought to be able to securely mount the unit in the peripheral expansion box.

Otherwise, CompuAdd's system performs great with either *Disk Manager 1* or 2 (giving you 180K or 360K respectively). This system does what it is supposed to do, and does it well. It's easy to install, and best of all, it's priced right. I recommend it. **HCM**

MOUSING AROUND

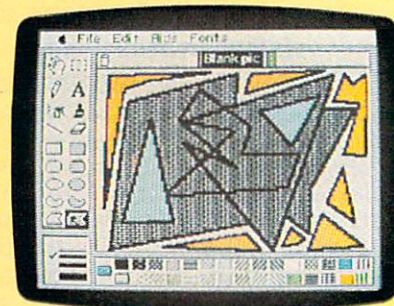
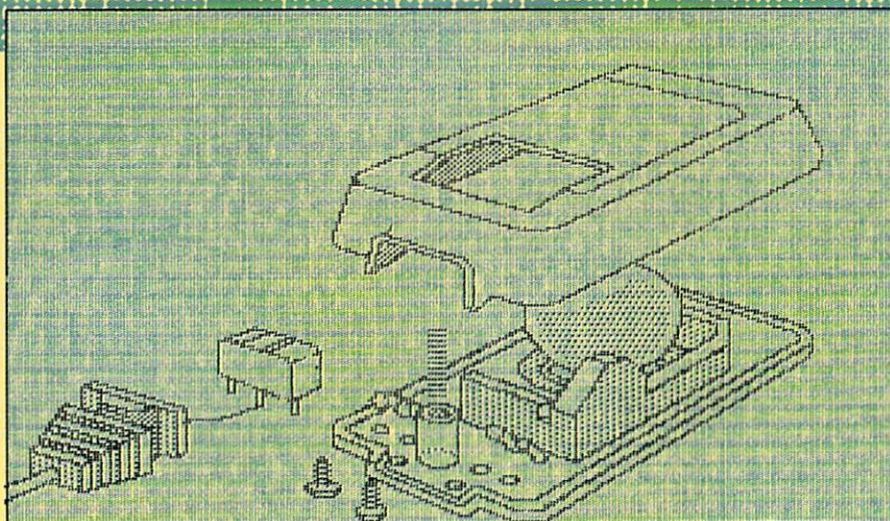


Photo 1

Love MacPaint but can't afford a Mac?
A new product now gives the rest of the "rest of us"
a chance to mouse around without blowing
our whole bankroll.

A review
by Roger Wood
 HCM Staff

MACINTOSH... maybe I'll get one. But I've got so much tied up in my Apple II I hate to spend the money. If you find yourself in this quandary, don't feel alone. Many Apple II owners would like to try out a mouse without splurging on a whole new system. *AppleMouse II* may just be the answer for someone with this dilemma. The expansion card with its support software, *MousePaint*, gives your Apple II, II+, or IIe access to a mouse for a fraction of the price of a Macintosh—without making all of your hard-earned software and hardware additions obsolete. The new Apple IIc already has the necessary internal hardware and firmware (ROM-based software) to

accommodate a mouse—so for about \$100, you IIc owners can easily get into mouse graphics.

In addition, some third-party, mouse-based software has already been released for the Apple II family (see insert), with more becoming available all the time. This means that a mouse for your Apple will be a long-term enhancement, not just a novelty.

The mouse device of *AppleMouse II* is identical to the one used with the Macintosh and Lisa computers. This small, plastic box has a single push-button on top, and contains a ball which protrudes from the bottom, held in place by a locking ring. The ball allows you to move the Mouse around on your desk top while a pointer on the screen echoes your movements, thus fostering man-machine communication in a new visually-oriented way.

The *AppleMouse II* card is easy to install and even comes with a small wrench, so no tools are needed (although a small nut-driver works better than the wrench). The package also includes a small connecting box to provide strain relief when installed on the Apple II or II+. The card may be installed in any of the expansion slots, but for maximum software compatibility, it is best installed in slot 4. This is because the new, "slotless" Apple IIc uses an address scheme for its

mouse firmware based on an *AppleMouse II* installed in slot 4.

Once your mouse is installed, you're ready to enter the world of mouse graphics using the *MousePaint* software. If you are new to the mouse, you can try the tutorial that is included; it takes you through clicking, dragging, pulling down menus, and the related mouse functions. Although the tutorial is clear and easy to follow, it is limited in scope. It covers mouse operations, but not *MousePaint* functions. A similar tutorial covering the finer points of *MousePaint* would have been a welcome addition.

In Color, Or Not?

When we first started working with *MousePaint*, our Apple IIe was hooked up to a color monitor. Our immediate observation was "Oh boy, *MacPaint* with color." After just a few minutes, however, we found that without a thorough knowledge of Apple high-resolution (hi-res) graphics, color with *MousePaint* doesn't work as you would expect. Then, upon careful examination of the manual we found no mention of color at all. After checking with Apple we learned that *MousePaint* was not written to support color, and that the color performance was a function of the Apple hi-res hardware—not by design of *MousePaint*. In the words of one Apple technician, "It's the nature of the beast..." (the beast being the Apple motherboard, and not a ferocious mouse).

We also discovered that the black used in *MousePaint* is hi-res black1—never black2. Thus, drawing a black line on an orange (or

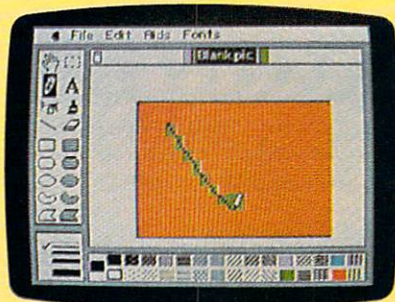


Photo 2

on the Apple II

blue) shape results in a strange green (or purple) swatch on either side of the line (see Photo 2). [For more information about color on the hi-res screen, see the appropriate "Home Computer Tech Note" on the subject in the August 1984 issue of *Home Computer Magazine*—Ed.] By carefully placing shapes on the screen, beautiful color graphics can be obtained using MousePaint—but an in-depth knowledge of Apple graphics is required.

Mouse Files from BASIC

MousePaint is a ProDOS-based utility; all of your drawings must therefore be saved on ProDOS-formatted disks. Apple has foreseen that many people buying the mouse do not have ProDOS, so they have provided a Format Disk option in the MousePaint file menu so that you can save your pictures. If you do have a ProDOS BASIC disk, you can

MousePaint an extremely useful graphics utility for any BASIC programmer—allowing the mouse to be used in creating whole screens of graphics for user-written programs.

To look at a MousePaint picture on the hi-res screen, type in the following lines from command mode with your file disk in the disk drive:

```
BLOAD /MY.DISK/MY.PICTURE
PRINT PEEK(49232),PEEK(49239)
```

The first line loads the picture into page 1 of the hi-res screen memory area, and the second line turns on the two "soft switches" to display the hi-res screen.

This opens the door to making hard copies of your mouse-created drawings using any printer interface that can print the hi-res screen. For printing your pictures directly from MousePaint, the only peripherals that the documentation mentions are the Apple Super Serial Card with

accessing the mouse from either Applesoft BASIC or assembly language programs. Apple has built its business on keeping its architecture and systems accessible to users, and these appendices demonstrate that Apple is continuing this tradition. With this easy-to-follow programming support, all you programmers can start writing mouse routines from the very first day.

"Once you discover that MousePaint files are simply BINARY files that can be loaded back into the hi-res screen area, it becomes an easy-to-use and very powerful graphics utility."

not only format disks, but can also do a great deal of manipulation of your MousePaint files. File management from MousePaint might confuse someone who is accustomed to DOS 3.3 instead of ProDOS. In ProDOS each disk has a distinctive volume name. To save a picture, MousePaint requires you to refer to a file by its complete "pathname"—i.e. volume name plus its particular file name. For example, to save your latest mouse-work to a disk named MY.DISK as a file called MY.PICTURE, you must enter /MY.DISK/MY.PICTURE. Considering these long pathnames, the MousePaint file menu is deficient in its omission of a disk catalog. You must remember both the volume name and the file name of a picture in order to call it up for editing.

Unfortunately, the MousePaint documentation does not go into detail about how MousePaint files are saved (they are simply BINARY files). For instance, we found that once saved, the files can be BLOADED into the hi-res screen memory area from Applesoft BASIC and displayed from a BASIC program. This makes

the Apple Dot Matrix Printer, or the ImageWriter. But once back in Applesoft, we were able to print graphics with a Buffered Grappler+ and an Epson MX-80 dot-matrix printer. All we did was BLOAD the picture, as shown above, but instead of setting the soft switches, we entered the following command—causing the Buffered Grappler+ to print the hi-res screen:

```
PRINT CHR$(9); "G"
```

By using similar techniques with other graphics printer interfaces, you can undoubtedly get similar results.

MousePaint is particularly handy in putting text into hi-res graphics displays. The Text option on the MousePaint screen, identified by the large letter A icon, lets you include text (in one of five different fonts, or type-styles) with your mouse drawings. By creating the text with MousePaint, it can be saved as a MousePaint file and then BLOADED by a basic program back into the hi-res screen.

Programming The Mouse

The best parts of the documentation are the appendices which cover

Name:	AppleMouse II and MousePaint
Program	
Hardware Type:	Graphics Creation
Machines:	Apple II, IIe, II+ (w/64K RAM), IIc.
Distributor:	Apple Computer 20525 Mariani Avenue Cupertino, CA. 95014 (408)996-1010
Price:	\$99 for Mouse and MousePaint (for Apple IIc) \$149 for Mouse, AppleMouse II expansion card, and MousePaint
	Poor Fair Good Excellent
Performance:	██████████
Ease of Use:	██████████
Ease of Setup:	██████████
Documentation:	██████████

Even with its deficiencies in documentation (the lack of detail about MousePaint features, no mention of color, and limited information about MousePaint files), AppleMouse II is an excellent product. Once you discover that MousePaint files are simply BINARY files that can be loaded back into the hi-res screen area (either in command mode or from an Applesoft program), it becomes an easy-to-use and very powerful graphics utility. We've become quite fond of the little critter and look forward to wheeling him around our Apple II for some time to come.

HCM

A sampling of third-party AppleMouse software:

Jane IIc/ Arktronics
Bank Street Writer/ Broderbund
Home Accountant/ Continental Software
Crypto Cube/ Designware
Cut and Paste/ Electronic Arts
Pinball Construction/ Electronic Arts
Graphics Magician/ Penguin Software
Catalyst IIc/ Quark
Grandma's House/ Spinnaker Software
Rocky's Boots/ The Learning Company
Dollars and Sense/ Tronix/ Monogram
Micro Cookbook/ Virtual Combinatics
Stickybear Shapes/ Xerox Educational Pub.

HCM Review



Name: Axiom GP-100 TI II Graphic Printer
 Product Type: Impact Printer
 Machine: TI-99/4A
 Distributor: Axiom Corporation
 1014 Griswold Avenue
 San Fernando, CA. 91340
 Price: \$299 for Printer with Parallax TI Expansion Interface, \$99 for Parallax TI Expansion Interface alone.

System Requirements: None

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

A Shortcut to 99/4A Printing:

A Review of the Axiom GP-100 TI II Graphic Printer

by Tom Green

HCM Staff

The Axiom Corporation—which markets a high-tech line of color graphic printers—has not turned its back on older, yet still-functional systems. A case in point is the Axiom GP-100 TI II—a printer equipped with a module (the Parallax TI Interface) that plugs directly into the side expansion port of the TI-99/4A. The module's function is threefold: (1) it is a quick-connect parallel interface for the Axiom GP-100 TI II (without requiring the TI Peripheral Expansion System), (2) it serves as a parallel print interface for "any" parallel-input printer for the TI-99/4A when ordered as a separate unit, and (3) it features a built-in edge connector for other standard peripherals, such as the TI Peripheral Expansion System.

Print Control Options

The general specifications for the Axiom GP-100 TI are listed in Figure 1. Interface (software) control options for text formatting include suppression of line feeds or carriage returns (thus overriding these automatic functions), added line feeds for double (or triple, or . . .) spacing, line length designation, and left margin setting.

Special printout formatting features (executed from BASIC using ASCII codes) give the programmer some powerful options. For instance, the TAB functions of TI BASIC operate with the printer, augmented by a POSition function that allows the printer to type at any column width (00-79), in any sequence. This means that you can enter characters at the end of a line, and backtrack to print more characters on the same line. All text can be changed to double-width characters; for bold type, there is an overprint option.

So far, we've concentrated on text options, but the real strength of the

Axiom GP-100 TI II is its graphics printing capabilities. The dot matrix impact head is arranged as 7 rows (or 7 needles). Each needle is addressable and can be controlled to print at any column using the ESC POSition

Axiom's GP-100 TI II printer with parallax interface is a triple bypass operation—providing lowcost printing without the usual expansion system.

command. To join dot patterns between lines, there is a control code that compresses the line spacing. With these options, you can customize printouts with special symbols or high-resolution pictures.

A Trouble-Free Package

The Axiom GP-100 TI II is marketed as an inexpensive printer for the TI-99/4A, operating with or without the TI Peripheral Expansion System. It fulfills its promise, operating trouble-free and installing easily. In addition, when daisy-chained to the Peripheral Expansion System, the RS232 option is still functional.

Printer-specific errors (indicated by a light that appears on its front panel) include detection of abnormal timing between machines, and carriage return malfunctions. It's too bad there are no error detection messages like "Out of Paper" or "Change Ribbon" for those of us who don't pay attention to such details.

As indicated in Figure 1, lower- and upper-case characters are standard with the Axiom GP-100 TI II. The lower-case print, however, is of a style that some people find annoying—there are no "descenders"—that is, the let-

ters g, j, p, q, and y are printed level with the baseline. According to one representative of the Axiom Corporation, there is a trade-off between offering printers with lower-case descenders and minimizing the final retail price. They chose to keep the price at a minimum, and still offer a competitive package. And, technically, the strongest element of this printer is its graphics production, not its text.

The printer is sold with a one-year warranty, including parts and labor. If a printer requires repair, Axiom will fix and return the printer within 48 hours, or send you a new machine free of charge. Normally, the print head and ribbon wear out first on a dot-matrix printer. The replacement cost of the print head is \$49.50. Under normal usage, it has an estimated life span well beyond one year. The ribbon will last an average of 500 pages of type, and costs \$9.95 to replace.

Documentation

The Axiom GP-100 TI II manual includes set-up instructions, testing sequences, and explanations of all print execution commands. This user's guide comes complete with detailed diagrams of the interface process, and program examples of all command features. The only documentation error that came to my attention is on

page 24—a reference is made there to the replacement of character sets, which does not apply to this printer.

Considering its dual-purpose print capabilities, its expedient installation procedure, and the compactness of its interface module, the Axiom GP-100 TI II proves to be a valuable addition to the TI home system.

HCM

Figure 1.
Axiom GP-100 TI II Printer Specifications

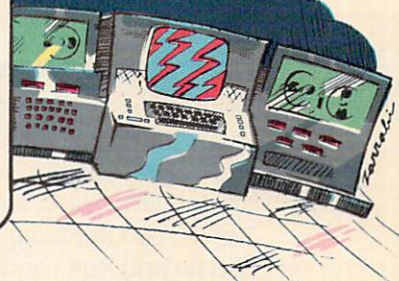
- | | |
|----------------------|--|
| 1. Character matrix | 5 by 7 dot matrix standard, with double width capability. |
| 2. Characters | Full upper/lower case characters, numerals and symbols. |
| 3. Graphics | Dot addressable 7 vertical dots per column, 480 columns maximum. |
| 4. Print speed | 30 characters/second—left to right, unidirectional. |
| 5. Maximum width | 80 columns |
| 6. Character spacing | 10 characters/inch. |
| 7. Linefeed spacing | 6 lines/inch—Character mode.
9 lines/inch—Graphic mode. |
| 8. Linefeed space | 5 linefeeds/second—Character mode.
7.5 linefeeds/second—Graphic mode. |
| 9. Paper feed | Pin feed with manual advance for fanfold-style paper. |
| 10. Paper width | 4.5 to 10 inches acceptable. |
| 11. Multiple copies | 2 including original. |



simon sez:

Lessons on Using Simon's BASIC

by **W.K. Balthrop**
HCM Staff



Simon's BASIC is an exciting development in the Commodore world. This plug-in cartridge enhances the resident BASIC in the Commodore 64 by adding 114 new commands. It's a programming language that instructs the computer to do many marvelous things—tricks that you may have thought only possible with assembly language programs.



Generate BASIC line numbers automatically with the AUTO command. Example: `AUTO 100,10`



When you enter the sample line above, the computer will display the number 100 followed by the blinking cursor. This is your program's first line number, and anything you type on this line, until you press [RETURN], will be accepted as line 100. The first number after the AUTO command tells the computer to start with that line number. The second number tells the

computer how much to increment each following line number. You should always have the automatic line numbering sequence start after the last line in your program. If you specify that AUTO start at an existing line number, that number will be erased from memory and replaced by whatever you type following the new line number.

Control the number of lines printed to the screen with the PAGE command. Example: `PAGE 20`

Entering the command above will cause only 20 lines at a time to be printed on the screen when you use the LIST command to look at your program. Just press the [RETURN] key to see the next 20 lines on the screen. The number of lines specified is in screen lines not program lines. Some of the program lines may be longer

than a single screen line. If this occurs at the bottom of the screen, only the first part of a program line may be shown on the screen. In this case, when the next group of lines is requested (by pressing the [RETURN] key), that entire program line will be displayed at the top of the screen.



Find all of the commands in your program requiring Simon's BASIC by using the OPTION command. Example: `OPTION 10`



Enter the above command after your program is loaded into memory. Then when you LIST the program, all the special Simon's BASIC commands will be written to the screen in inverse video. If you have a printer compati-

ble with the Commodore graphics characters, LISTing to the printer will cause the Simon's BASIC commands to be printed inversely (white characters on a black background).

Change the speed that the computer LISTs your program to the screen with the DELAY command. Example: `DELAY 50`

The DELAY command allows you to change the LISTing speed to the screen by using the [SHIFT] key. First, load your program into memory. Then enter the above command. Now, when you LIST to the screen, holding down the [SHIFT] key will cause the LIST-

ing speed to slow. The larger the number following the DELAY command, the slower the program will LIST while the [SHIFT] key is held down. Releasing the [SHIFT] key will cause the LISTing to go full-speed to the screen.



HCM

Industry Watch

SUNWARE PROVIDES CARTRIDGE CONVERSIONS FOR TI SOFTWARE

In an effort to increase software development for the TI-99/4A, two ex-TI employees have formed a firm, SunWare, to convert disk-and cassette-based software into cartridges. The firm will either market them or sell them back to the developers to market themselves. SunWare will produce two types of cartridges—a Peripheral Port cartridge which can duplicate the functions of a floppy disk (up to 48K ROM or RAM), and a Command Port cartridge, which has 32K ROM or RAM memory.

APPLE OPENS MACCOLLEGE FOR SOFTWARE DEVELOPERS

A new program has been put together by Apple Computer to assist experienced, independent programmers in developing software for the Macintosh. Dubbed MacCollege, it will be located at the company's headquarters in Cupertino, California and will provide resources and instructions to programmers certified under Apple's Certified/Registered Developers program—a support program for Apple-compatible products.

LEADING EDGE GOES AFTER IBM WITH A PEANUT OF ITS OWN

Going after the Apple IIc/IBM PCjr market, Leading Edge Products of Needham, Massachusetts is reportedly preparing to mount its attack using—Peanuts. The name chosen for the PC-compatible manufactured for Leading Edge by Matsushita of Japan is (coincidentally?) the PCjr's common nickname. Declining to give specific details, Leading Edge did say that its Peanut will have a better keyboard than the Junior, and that it will be aimed at first-time home computer buyers.

TEXAS INSTRUMENTS RELEASES DEBUGGER

Texas Instruments recently announced that it has relinquished to the public, for free use, any and all proprietary claims to the Advanced Assembly Language Debugger. Copies have been mailed to all 99/4A user groups. TI further stated that this software is not covered by warranty in any fashion, and that it assumes no responsibility for its use.

RADIO SHACK GOES DOOR-TO-DOOR

Banking on the assumption that there is a large number of people who are intimidated by the entire computer-buying process, Tandy/Radio Shack is planning to bring their Color Computer 2 right into prospective buyers' homes for a demonstration. Included in this package is the Color Computer 2, educational software, joysticks, modem, and disk drive. Beginning with 13 markets, using a direct mail and magazine advertising campaign as well as 250 salespeople, Radio Shack reportedly plans to cover the entire nation within three years.

COMMODORE PLANS TO INTRODUCE A NEW COMPUTER

Relying on their flagship, the Commodore 64, Commodore International Ltd. is predicting substantial sales and profits for 1984. The company is also planning to introduce a new home computer with the power of Apple Computer's \$2,495 Macintosh at a price under \$1,000. The computer will be based on a more powerful 32-bit microprocessor, giving it better graphics and a friendlier user interface. Next year, Commodore will offer expanded memory and a higher resolution 80-column display for the Commodore 64.

Any Questions?

The present global ignorance of computing may come, in part, from our natural aversion to asking simple questions—for fear of revealing only a shallow knowledge of vital topics.

Why not let someone else ask the questions while we sit back and benefit from the reply? That's the purpose of this column.

Q. I keep hearing the term "buffer." What is it and what does it do?

A. A buffer is an interface between the Central Processing Unit (CPU) of a computer and any peripheral equipment such as a printer, disk, plotter, etc. The term is most often used to refer to a memory area that is used as temporary storage space for data until the peripheral device is ready for it. For example, printers operate at a much slower speed than microcomputers; to assure that the data to the printer is not lost, it is stored in a buffer and transmitted at a speed that the printer can handle. Similarly, most printers have buffers (memory storage areas) as part of their circuitry, so they can store data while the actual printing operation is in progress. In some instances, buffering also refers to taking care of any voltage level differences between a computer and a peripheral device.

Q. What is the difference between parallel and serial transmission?

A. Serial transmission means transmitting data one bit at a time over one signal path (wire). With parallel transmission all of the bits (usually as an 8-bit byte) are transmitted simultaneously using a separate signal path for each bit of data.

The major advantage of serial transmission is that it is generally less susceptible to electronic noise than parallel. It works better over long distances and is the method used to interface to "modems" for transmission over telephone lines.

The major advantage of parallel transmission is that it can be much faster because several bits are transmitted simultaneously. This method is very flexible and is excellent for use with digital to analog (D/A) and analog to digital (A/D) applications. Although the parallel method is faster, as the length of cable gets longer, the data transmitted may pick up electronic noise and become garbled.

Q. What is the difference between machine language and assembly language?

A. There are two basic types of languages used in microcomputers—low-level and high-level languages. Low-level languages communicate more directly with the hardware of the computer. High-level languages are easier for people to understand and act as translators between programmers and a machine's hardware. The lowest-level language is machine language. It tells your computer what to do using binary numbers. But working with binary numbers is time-consuming and error-prone. Programmers developed assembly language to speed up writing machine language programs.

Assembly language is also a low-level language, but it uses symbols called mnemonics (two or three letter abbreviations) that stand for machine language instructions that are closer to English than binary numbers. It is, therefore, much easier to work with. In order for the computer to use the assembly language program, it must be translated into machine code by a program called an "assembler."

Q. I'm thinking about buying a home computer. How much memory do I need?

A. This is a good question. The answer will vary depending on how you plan to use your computer. If you are planning to use it to store large amounts of data, run memory-intensive applications, or play complicated games like chess, then you will need more memory than someone who plans on using his or her computer to begin learning simple programming or to play video games. The amount of memory space required is directly related to the number of instructions needed to do the job assigned.

Many small computers come with less than 64K of memory. This can be enough if the computer has cartridge slots for games or specific applications, and the user is just starting out in computing—as long as the computer can be expanded to meet future needs. Examples of this type of computer are the TI-99/4A or VIC-20. Computers with 64K such as the Commodore 64, or the unexpanded Apple IIe or IBM PCjr, allow for more extensive programming applications, and should prove quite adequate for most home applications.

Finally, those computers that contain 128K or more memory are best for people who wish to use fancier word processors and more extensive business applications. One thing to remember is that as you become more proficient at computing and programming, you will eventually need more memory. You could either buy it now in the form of a more powerful computer, or add on memory as needed.

Q. Why won't programs written for the TI-99/4A work on any other machine and vice versa? If all microcomputers use BASIC as their language, why aren't programs interchangeable between systems?

A. The main reasons have to do with the different Central Processing Units (CPUs) in the various computers, and the way memory is managed by the different operating systems.

All of the data, commands, and functions needed for operation of a computer are controlled by the CPU. Because different computers use different CPUs, they require different machine-level instructions to process, store, and channel information. Even though two machines use BASIC, how a CPU interprets the BASIC instructions depends upon the way the computer translates the instructions into machine-level commands. Computers with different CPUs will interpret and even store identical BASIC statements in different formats best suited to their particular method of processing.

Another reason is in the way that memory is allocated. Even though two computers may use the same CPU, their operating systems can differ drastically. Some systems have more memory than others, and some may require a specific amount of memory to be set aside for specific functions. These differences all contribute to non-interchangeable programs.

Machine-specific functions and commands are one more reason. Every machine has commands and functions that do not work on other systems. Because of these differences, most programs are not interchangeable.

HCM



TAX DEDUCTION FILER

Tax time, 1984, has come and gone—but it's not too late to get organized for 1985. Here's a program that can help you get ready for next year's income tax preparation right now.

by Roger Wood
HCM Staff

If you itemize your deductions using Schedule A, often you can save yourself a lot of money. But keeping track of a year's worth of deductible items can indeed be a tremendous chore—especially if you, by default, rely on the “Shoebox Method” of data management (alternately referred to as “Pitch Now—Worry Later”). But hark! The Mighty Mouse of 1040 Land has come to save the day—the day's receipts, that is . . .

By saving your deductible data with the *Tax Deduction Filer* every few weeks, you can make itemizing a more pleasant, manageable, and profitable task. You should be aware that IRS forms tend to change yearly, so some minor modifications of the program may be necessary for 1984 taxes. This program is not a substitute for reading the IRS instructions, but it will help keep your records straight.

Tax Deduction Filer was inspired by a program entitled *Schedule A*, submitted by Marty Casado of Eugene, OR.

Using The Program

Here's how the program can help simplify your record keeping. Let's say you send a check for \$50 to your local public television station. Tax deductible, right? By the time next year's taxes are due though, you may have trouble remembering in which month you sent the check, let alone its amount. So you have to plow into your old checkbook registers and cancelled checks to find that deduction. With this program, however, you can sit down with your computer every few weeks, add new deductions to your files, and have all of your deductions neatly compiled for Schedule A when tax time rolls around.

The program is easy to use and totally menu-driven. The first menu asks you which part of the program you wish to access:

1. ADD DATA
2. CHANGE DATA
3. DISPLAY DATA
4. TOTALS
5. PRINT REPORT
6. LOAD DATA FILE
7. SAVE DATA FILE
8. EXIT PROGRAM

If you select any of the first three options, you are presented with a menu of the 17 deduction categories available:

1. MEDICINE AND DRUGS
2. DOCTORS, DENTISTS, ETC.
3. MEDICAL TRANSPORTATION
4. OTHER MEDICAL
5. STATE AND LOCAL INCOME TAX
6. REAL ESTATE TAX
7. MOTOR VEHICLE SALES TAX
8. OTHER TAXES
9. HOME MORTGAGE INTEREST
10. CREDIT CARD INTEREST
11. OTHER INTEREST
12. CASH CONTRIBUTIONS
13. OTHER CONTRIBUTIONS
14. CASUALTY AND THEFT
15. UNION AND PROFESSIONAL DUES
16. TAX PREPARATION FEES
17. MISCELLANEOUS

If you examine Schedule A, you will find that these categories generally follow the schedule's format. This makes it convenient to transfer your data from the program to the form at tax time.

The fourth option on the first menu displays the totals of each of the 17 categories. If you have a printer connected to your computer, the Print Report option (number 5) gives you a hard copy of your deductions.

Options 6 and 7 are for loading and saving your data files for long-term storage, and Option 8 is for exiting the program. This last option has a built-in safety feature, which inquires whether you are ready to halt the program and wipe out all the data in memory, or, if you want to first save the data before halting.

Program Implementation

When you enter data into the program you are asked to give a description of the item and an amount. The description could include to whom you paid the expense, as well as the date; its only limitation is that it cannot exceed 27 characters. (This keeps the displays easy to read.) Any time you press [ENTER] or [RETURN] without making an entry, the program returns you to the main menu to select another option.

After you have made an entry, the program stores the information in one member of the string array `AS()`. This array is DIMensioned to 500, which should be plenty for most households.

Data is stored in the array in a unique fashion. When a category is selected, the program takes its number (between 1 and 17, see list above) and adds it to 100. This value is stored as the initial ASCII character in the

array element (the value would be between 101 and 117). Next, the length of the description is found using the LEN function, and 100 is added to it. This ASCII value (somewhere between 100 and 127) is placed in the array as the second character. The description and the STR\$ of the amount are "concatenated" with the first two ASCII characters to form the total string value of the element. The program can then identify the type of data in any element by simply taking the ASCII function of any member. In addition, the ASCII value of the second character is used to index the length the description and the starting location of the amount in the string.

Whenever you use the program, you don't need to worry about how many items will eventually be entered in a given category, because all of the different categories

are stored in one array. The total number of items is stored in the variable R—itself being the first one stored in the data file.

Because the pro-

gram adds each item as the next available array element, it automatically keeps track of the order in which the items are entered.

For C-64, IBM PC & PCjr, and TI-99/4A see next page.



Apple's new ProDOS operating system removes certain bugs that were present in the old DOS 3.3 system. The Convert program that comes on the ProDOS master disk makes converting a program from one system to the other relatively easy—but the old DOS 3.3 bugs don't make it totally fool-proof.

Whenever you access the printer in a program from ProDOS, you must include `PRINT CHR$(4);` in front of the `PR#1` statement. Preceding `PR #1` with `PRINT CHR$(4)` under DOS 3.3 doesn't work with many printer interfaces (VIDEX and Grappler+, for example). However, a `PR #1` without the additional `CHR$(4)` works just fine.

To ensure that this program can be loaded under one operating system and then be converted to the other without difficulty, we included a prompt (see lines 200-230) so the user can tell the computer under which system the program is running. A flag (OP) is set at this point in the program so that when the printer is accessed later, the proper commands are selected (see lines 1070 and 1090).

Tax Deduction Filer (Apple) Explanation of the Program

Line Nos.	Explanation of the Program
100-170	Program header.
180-190	DIM arrays and initialize variables.
200-230	Title screen and operating system prompt.
240-280	Main menu and keyboard input.
290-300	Display categories subroutine.
310-320	Extract one record from array subroutine.
330-400	Display category subroutine.
410-480	Add data to array.
490-660	Change data in array.
670-730	Subroutine to find a specific record.
740-800	Display a category.
810-860	Figure and display totals.
870-1100	Send data to printer.
1110-1190	Load data file.
1200-1300	Load data error trapping routine.
1310-1420	Save data file.
1430-1520	Save data error trapping routine.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

The C-64 version of *Tax Deduction Filer* printer routine is designed to use the VIC-1525 printer. This serial printer is treated as a sequential output file using serial channel 4. The printer does not support the regular TAB function that you use to format data on the screen. Instead, a special control code, CHR\$(16) is used. A good example of this occurs in line 1180:

```
1180 PRINT#4,"CATEGORY";CHR$(16)"10DESCRIPTION";
CHR$(16)"40AMOUNT";CHR$(13)
```

Here, the word CATEGORY is printed at the left edge of the paper. CHR\$(16) tells the printer that the next two-digit number will specify the column where the next entry is to be printed. Notice that this two-digit number must be part of a string. In this instance, the word DESCRIPTION will begin in column 10. Likewise, the word AMOUNT will begin in column 40. Notice that the numbers 10 and 40 will not be printed, even though they appear in the same quotes as the words DESCRIPTION and AMOUNT. Because the strings are preceded by CHR\$(16), the printer knows that these numbers are for formatting purposes.

For further information on this and the rest of these printer control codes, refer to the VIC-1525 manual. If you are using an RS232 interface for your printer,

then the above control codes may not be supported by your printer, and you will need to modify the Print Report section (lines 1060-1380) of the program to conform to your printer's specifications. [Would anyone who tries this please send in a Letter-to-the-Editor with specific details. It would be helpful to other readers who wish to follow in your footsteps.—Ed.]

Tax Deduction Filer (C-64) Explanation of the Program

Line Nos.	
100-170	Program header.
180-190	DIM arrays and initialize variables.
200-240	Title screen.
250-290	First menu.
300-310	Display categories subroutine.
320-340	Extract one record subroutine.
350-450	Add data to array.
460-710	Change data in array.
720-810	Subroutine to find a specific record.
820-970	Display a category.
980-1050	Figure and display totals.
1060-1380	Send data to printer routines.
1390-1550	Load data file.
1560-1730	Save data file.
1740-1800	Disk error routine.
1810-1870	End program routine.
1880-1950	Data statements for menus.

HCM

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.



IBM's BASIC file management system is so versatile that disk and cassette options can share much of the same code in the program. For example, because both cassette and disk options use the basic DOS input and output scheme, we used the same OPEN statement for both options (see lines 1020 and 1130). F1\$ is determined by the user's input (lines 980-100 or lines 1090-1110), and no other special disk or cassette instructions are required. Most other computer systems (e.g., TI, Apple, C-64) require separate routines for each type of device. If you run the program on an IBM PC, be aware that your data disk must be in Drive A when you save or load.

Line 500 contains a useful trick for centering text on the screen—such as titles, headings, etc. It is based on a 40-column screen.

```
500 CLS:LOCATE 2,(20-LEN(N$(C))/2):PRINT N$(C).
```

Here, the horizontal location of the cursor (the start-

ing column) is determined by the LENGTH of N\$(C), where N\$(C) is the category to be displayed. Specifically, half of the length of the category name is subtracted from one-half of the width of the screen. This causes the title of the particular category to be centered on the screen.

Tax Deduction Filer (IBM PC and PCjr) Explanation of the Program

Line Nos.	
100-160	Program header.
170-180	DIM arrays and initialize variables.
190-230	Title screen.
240-290	Main menu and first input.
300-450	Add data routine.
460-610	Change data routine.
620-730	Search for record.
740-780	Totals routine.
790-950	Print report routine.
960-1060	Load data file routine.
1070-1170	Save data file routine.
1180-1200	End program.

HCM

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.



The menu-driven nature of *Tax Deduction Filer* demands quick changes between text screens. Extended BASIC's DISPLAY AT and ACCEPT AT statements make it much more suitable to this application than TI BASIC.

In TI Extended BASIC, we use the SEG\$() command to extract the characters to help implement *Tax Deduction Filer's* unique file structure. You can see the string constructed in line 430. The subroutine in line 810 takes the string apart, returning the value of the category in AC, the item name length in AL, the item name in D\$, and the item value in V\$. This technique is very useful in simplifying the code for the program, and in reducing the amount of memory required for data storage.

This same technique is handy for a multitude of purposes: You could insert a character that represents the previous item or the next item in a search chain, thus eliminating the need for manipulating whole records when you sort. You could also use several characters as qualifying categories when searching for items. Now see if you can come up with some original

ideas for this technique. And don't forget to let us know about them in a Letter to the Editor.

Tax Deduction Filer (TI-99/4A) Explanation of the Program

Line Nos.	
100-170	Program header.
180-190	Initialize program variables and arrays.
200-220	Display the main menu and input choice.
230-280	Add data routine.
290-440	Change data routine.
450-500	Display data routine.
510-520	Calculate and display totals.
530-680	Print reports.
690-740	Load data into memory from cassette or disk.
750-800	Save data to cassette or disk.
810	Routine to retrieve information from the data string.
820	Display mode option screens.
830-860	Input subroutines.
870	Data for the main menu screen.
880-930	Option screen data for display.
940-950	Subroutine to catch the user when trying to exit the program. Give opportunity to back out and save data first.

HCM

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.



KALEIDO

COMPUTER

by **Melody Covington**

and the HCM Staff

I recently came across a kaleidoscope in a store. Fascinated by the twirling imagery, I played with it for several minutes. "I could do that on my computer," I mused. The symmetry could be mathematically controlled, and a feeling of movement would be easy to generate. I had formed rough ideas for the programs, even before I got home.

Little Stars

My first Kaleidoscope program, *Little Stars*, wasn't very difficult. My main task involved creating a symmetrical pattern, which was fairly easy to do because the characters are placed on the screen with coordinates. If I knew one set of coordinates, then three more could be calculated to produce a mirror image in each corner. A short subroutine accomplished that. The next step was to randomly select a location for the first set of coordinates.

The only task left was to design the algorithm (the method by which a task is solved). The problem was simple: For every location selected on the screen, find three more sets of coordinates that define mirror image locations of the first. By mirror image, we mean for every character in the upper left corner of the screen, there will be a similar character in the corresponding lower left, lower right, and upper right corners of the screen.

For example, let's look at a screen with 32 columns and 24 rows. If a character is placed in column 2, row 2, then we will also need a character at column 2, row 23, and so on. Now it's easy to come up with a quick solution. To find the mirror of a column, subtract the column number from 33. To find a mirror row, subtract the row number from 25. You can then use just the new column (and old row), just the new row (and old column), and both the new row and new column to find the three mirror positions. The values you subtract from the row and column will vary with the number of rows and columns on your monitor.

Now that I had an algorithm for obtaining the mirror images, I needed to find a spot on the screen to place the first character. The random number generator does this simply; its command varies slightly from machine to machine, but it always performs the same function. On most machines the `RND` or `RND(1)` function will return a number between 0 and 1. If you multiply that number times your screen width (or number of lines), it will generate a random number between 0 and your screen width (or number of lines). This way you can generate random screen coordinates. Once the coordinates for one character are generated, you can then use the mirroring algorithm to plot the other characters.

Color Kaleidoscope

This program is an extension of *Little Stars*. I just added color and graphics shapes to the previous program—the same algorithm is used to find the mirror images. Color is an important factor in graphics design. As you compare the first program with the spectacular effects of this one I'm sure you will agree. Continued

Linear Kaleidoscope

In this program I deviated slightly from the previous two, and added an enhancement. Instead of simply plotting random dots all over the screen, *Linear Kaleidoscope* shoots graphics characters across the screen in lines of color. The increased complexity is minor compared to the fantastic effects we can create.

APPLE II Family

```

100 REM *****
110 REM * LITTLE STARS *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM APPLE II FAMILY APPLESOFT
180 REM
190 HOME
200 REM PLACE 25 SETS OF * THEN PLACE
  25 BLANKS
210 FOR X = 1 TO 2
220 IF X = 1 THEN MS = "*"
230 IF X = 2 THEN MS = " "
240 FOR I = 1 TO 25
250 GOSUB 290
260 NEXT I
270 NEXT X
280 GOTO 210
290 REM SUB ROUTINE
300 REM RANDOMLY GENERATE COORDINATES
310 R = INT (RND (1) * 23 + 1)
320 C = INT (RND (1) * 32 + 1)
330 REM PLOT CHARACTER PLUS 4 REFLECTI
  ONS
340 VTAB R: HTAB C: PRINT MS
350 VTAB 24 - R: HTAB 33 - C: PRINT MS
360 VTAB R: HTAB 33 - C: PRINT MS
370 VTAB 24 - R: HTAB C: PRINT MS
380 RETURN
390 END
  
```

HCM

COMMODORE 64

```

100 REM *****
110 REM * LITTLE STARS *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM C-64 BASIC
180 REM
190 PRINT CHR$(147): SC=1023: CS=55295: J=1
  3
200 POKE 53280,0: POKE 53281,0
210 REM MAIN LOOP
220 FORM=42 TO 32 STEP -10
230 FOR I=1 TO 25
240 GOSUB 280
250 NEXT I
260 NEXT M
270 GOTO 210
280 REM SUBROUTINE
290 REM RANDOMLY GENERATE COORDINATES
300 R=INT(RND(1)*12)+1
310 C=INT(RND(1)*20)+1
320 REM PLOT CHARACTER + 3 REFLECTIONS
330 Q1=40-R+C
340 Q2=40-R+(40-C)
350 Q3=40*(24-R)+C
360 Q4=40*(24-R)+(40-C)
370 POKE SC+Q1,M: POKE SC+Q1,J: POKE SC+Q2,M
  : POKE SC+Q2,J
380 POKE SC+Q3,M: POKE SC+Q3,J: POKE SC+Q4,M
  : POKE SC+Q4,J
390 RETURN
  
```

HCM

VIC-20

```

100 REM *****
110 REM * LITTLE STARS *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM VIC 20 BASIC
180 A=7680
190 PRINT "SHIFT CLR"
200 POKE 36879,104
210 FOR CH=42 TO 32 STEP -10
220 FOR T=1 TO 15
230 X=INT(RND(1)*11)
240 Y=INT(RND(1)*11)
250 P=2*(10-X)
260 Q=2*(10-Y)
270 POKEA+X+22*Y,CH
280 POKEA+X+P+22*Y,CH
290 POKEA+X+P+22*(Y+Q),CH
300 POKEA+X+22*(Y+Q),CH
310 FORTD=1 TO 50
320 NEXT T,D,CH
330 GOTO 210
  
```

HCM

This program is actually a very primitive line-drawing routine in BASIC.

The algorithm used to draw a line is much simpler than you might suspect. I just needed to generate two random numbers, which serve as the horizontal and vertical slope of the line. An inner loop is responsible for the length of the line. Each time the loop repeats, the horizontal and vertical slope factors are added to the current position on the screen. If the same factors are added every time, the line will continue in the same direction. Remember, all we need to do is draw one line, and let the mirror-plotting routine draw the others. The result is four lines racing across the screen to create a beautiful symmetrical pattern.

"Instead of simply plotting random dots all over the screen, Linear Kaleidoscope shoots graphics characters across the screen in lines of color. The increased complexity is minor compared to the fantastic effects we can create."

IBM PC & PCjr

```

100 *****
110 * LITTLE STARS *
120 *****
130 BY MELODY COVINGTON
140 AND THE HCM STAFF
150 HOME COMPUTER MAGAZINE
160 VERSION 4.4.1
170 IBM PCjr WITH CASSETTE & CARTRIDGE
  BASIC
180 IBM PC WITH CASSETTE BASIC & BASIC
  A
190
200 SET UP
210 RANDOMIZE TIMER
220 CLS: SCREEN 0: WIDTH 40
230 PLACE 25 SETS OF * THEN PLACE 25
  BLANKS
240 FOR M=42 TO 32 STEP -10
250 FOR I=1 TO 25
260 GOSUB 300
270 NEXT I
280 NEXT M
290 GOTO 240
300 REM
310 RANDOMLY GENERATE COORDINATES
320 R=INT(RND*23)+1
330 C=INT(RND*39)+1
340 PLOT CHARACTER PLUS 4 REFLECTIONS
350 LOCATE R,C: PRINT CHR$(M);
360 LOCATE 24-R,40-C: PRINT CHR$(M);
370 LOCATE R,40-C: PRINT CHR$(M);
380 LOCATE 24-R,C: PRINT CHR$(M);
390 RETURN
  
```

HCM

TI-99/4A

```

100 REM *****
110 REM * LITTLE STARS *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM HOME COMPUTER MAGAZINE
150 REM VERSION 4.4.1
160 REM TI BASIC
170 REM
180 REM SET UP
190 RANDOMIZE
200 CALL SCREEN(2)
210 CALL CLEAR
220 CALL COLOR(2,16,2)
230 REM PLACE 25 SETS OF * THEN PLACE
  25 BLANKS
240 FOR M=42 TO 32 STEP -10
250 FOR I=1 TO 25
260 GOSUB 300
270 NEXT I
280 NEXT M
290 GOTO 240
300 REM SUBROUTINE
310 REM RANDOMLY GENERATE COORDINATES
320 R=INT(RND*24)+1
330 C=INT(RND*32)+1
340 REM PLOT CHARACTER PLUS 4 REFLECTI
  ONS
350 CALL VCHAR(R,C,M,1)
360 CALL VCHAR(25-R,33-C,M,1)
370 CALL VCHAR(R,33-C,M,1)
380 CALL VCHAR(25-R,C,M,1)
390 RETURN
  
```

HCM



Little Stars

Two FOR-NEXT loops in this program control the action. The first (outer loop) starts in line 240, and determines whether an asterisk or a space is to be plotted. The second (inner) loop starts in line 250 and dictates the number of characters plotted. These two loops will first plot 25 asterisks, then 25 blanks, then 25 more asterisks, and so on.

Color Kaleidoscope

Here, the TI computer assigns color to groups of eight characters so that the program can draw with up to eight different colors at the same time. Each group can have its own foreground and background color. (They are assigned in lines 400 through 470.) The first (outer) loop has been altered slightly from the first program. It still decides which characters are plotted, but the characters now have an ASCII value in the range of 96 to 152. These are the characters which were assigned colors earlier.

APPLE II Family

```

100 REM *****
110 REM * COLOR KALEIDOSCOPE *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM APPLE II FAMILY APPLESOFT
180 REM
190 HOME
200 GR
210 POKE -16302,0
220 FOR Y=0 TO 47
230 FOR X=0 TO 39
240 COLOR=0: PLOT X,Y
250 NEXT X: NEXT Y
260 REM LOOP TO CONTROL COLORS
270 FOR A=1 TO 15
280 COLOR=A
290 REM LOOP TO CONTROL NUMBER OF LINE
300 S DRAWN
310 FOR D=1 TO 25
320 REM RANDOMLY SELECT BEGINNING COOR
330 R=INT(RND(1)*39)
340 C=INT(RND(1)*47)
350 REM PLOT SPOT PLUS 3 REFLECTIONS
360 PLOT R,C
370 PLOT R,47-C
380 PLOT 39-R,C
390 PLOT 39-R,47-C
400 NEXT D
410 NEXT A
420 REM DELAY ROUTINE
430 FOR M=1 TO 1000
440 NEXT M
450 GOTO 200

```

VIC-20

```

100 REM *****
110 REM * COLOR KALEIDOSCOPE *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM VIC 20 BASIC
180 A=233:B=223:C=105:D=95:K=7680:J=384
190
200 PRINT "SHIFT CLR"
210 POKE 36879,104
220 FOR T=1 TO 15
230 X=INT(RND(1)*11)
240 Y=INT(RND(1)*11)
250 P=(2*(10-X))+1
260 Q=(2*(10-Y))+1
270 POKEK+X+22*Y,A
280 POKEJ+X+22*Y,CC
290 POKEK+X+P+22*Y,B
300 POKEJ+X+P+22*Y,CC
310 POKEK+X+P+22*(Y+Q),C
320 POKEJ+X+P+22*(Y+Q),CC
330 POKEK+X+22*(Y+Q),D
340 POKEJ+X+22*(Y+Q),CC
350 FOR TD=1 TO 50
360 NEXT TD
370 CC=INT(RND(1)*8):IF CC=6 THEN 370
380 IF A=32 THEN A=233:B=223:C=105:D=95:GO
390 A=32:B=32:C=32:D=32:GOTO 210

```

Linear Kaleidoscope

We now add the final stage to this program. A slope to draw the line is determined in lines 370 and 380. The loop in line 400 causes the line to continue drawing for 15 characters. Lines 410 and 420 add the slope values to the current position of the line, to continue its path.

TI-99/4A

```

100 REM *****
110 REM * COLOR KALEIDOSCOPE *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM HOME COMPUTER MAGAZINE
150 REM VERSION 4.4.1
160 REM TI BASIC
170 REM
180
190 CALL SCREEN(16)
200 CALL CLEAR
210 RANDOMIZE
220 REM DEFINE CHARACTERS
230 SS="0103070F1F3F7FFF"
240 TS="80C0E0F0F8FCFEFF"
250 US="FF7F3F1F0F070301"
260 VS="FFFEFCF8F0E0C080"
270 FOR A=96 TO 152 STEP 8
280 CALL CHAR(A,VS)
290 NEXT A
300 FOR B=97 TO 153 STEP 8
310 CALL CHAR(B,TS)
320 NEXT B
330 FOR C=98 TO 154 STEP 8
340 CALL CHAR(C,US)
350 NEXT C
360 FOR D=99 TO 155 STEP 8
370 CALL CHAR(D,SS)
380 NEXT D
390 REM DEFINE COLORS
400 CALL COLOR(16,14,15)
410 CALL COLOR(15,12,10)
420 CALL COLOR(14,9,16)
430 CALL COLOR(13,7,11)
440 CALL COLOR(12,15,14)
450 CALL COLOR(11,10,12)
460 CALL COLOR(10,16,9)
470 CALL COLOR(9,11,7)
480 REM THESE LOOPS PLACE 25 SETS OF E
490 ACH COLOR COMBO
500 FOR M=96 TO 152 STEP 8
510 FOR I=1 TO 25
520 GOSUB 550
530 NEXT I
540 NEXT M
550 REM SUBROUTINE
560 REM RANDOMLY GENERATES COORDINATES
570 R=(RND*12)+1
580 C=(RND*16)+1
590 REM PLOT CHARACTER PLUS THE REFLECT
600 IONS
610 CALL VCHAR(R,C,M)
620 CALL VCHAR(25-R,C,M+1)
630 CALL VCHAR(R,33-C,M+2)
640 CALL VCHAR(25-R,33-C,M+3)
650 RETURN

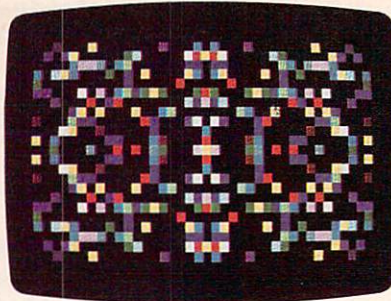
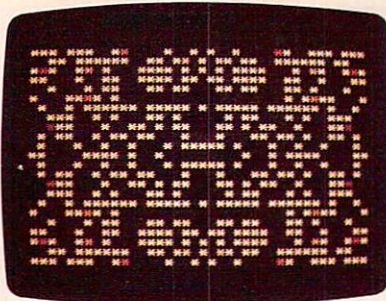
```

IBM PC & PCjr

```

100 REM *****
110 REM * COLOR KALEIDOSCOPE *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM IBM PCjr: CASSETTE & CARTRIDGE BASI
180 REM IBM PC CASSETTE BASIC & BASICA WIT
190 REM COLOR/GRAPHICS MONITOR ADAPTER
200 REM AND COLOR MONITOR
210 REM SET UP
220 REM RANDOMIZE TIMER
230 CLS:SCREEN 0:WIDTH 40
240 REM PLACE 25 SETS OF *, THEN PLACE
250 FOR M=219 TO 32 STEP -187
260 COLOR INT(RND*16)
270 FOR I=1 TO 10
280 GOSUB 320
290 NEXT I
300 NEXT M
310 GOTO 250
320 REM
330 REM RANDOMLY GENERATE COORDINATES
340 R=INT(RND*23)+1
350 C=INT(RND*39)+1
360 REM PLOT CHARACTER PLUS 4 REFLECTIO
370 LOCATE R,C:PRINT CHR$(M);
380 LOCATE 24-R,40-C:PRINT CHR$(M);
390 LOCATE R,40-C:PRINT CHR$(M);
400 LOCATE 24-R,C:PRINT CHR$(M);
410 RETURN

```

At far left is a representative screen photo of the Little Stars program. The photo at near left shows what can result from the Color Kaleidoscope routine.



Little Stars

Two FOR-NEXT loops control the asterisk plotting. The first loop, in line 220, determines whether the character to be placed on the screen is an asterisk or a space. The second loop repeats 25 times, printing a set of asterisks with each pass. Line 240 branches to a subroutine that places the characters on the screen. In this subroutine, two random numbers are selected in lines 300 and 310. These two numbers tell the computer where to place the first character on the screen.

Color Kaleidoscope

This program is the next step in the evolution of the Kaleido Computer. The asterisks have been replaced with colored shapes.

Linear Kaleidoscope

The first outer loop in line 250 changes the color with each pass. Line 280 starts two inner loops, which control the number of lines drawn, and the character used for drawing. Lines 300 and 310 select a starting point for the line. This is where the first character of the line will be drawn. Lines 330, 340, and 350 select a slope for the line.

COMMODORE 64

```

100 REM *****
110 REM * COLOR *
120 REM * KALEIDOSCOPE *
130 REM *****
140 REM BY MELODY COVINGTON
150 REM AND THE HCM STAFF
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 4.4.1
180 REM C-64 BASIC
190 REM
200 PRINT CHR$(147):SC=1023:CS=55295
210 REM ASSIGN COLORS FOR EXTENDED BACK
    GROUND COLOR MODE
220 POKE53265,PEEK(53265)OR64:POKE53282
    ,7:POKE53284,10:POKE53283,3
230 POKE53280,0:POKE53281,0
240 REM THESE LOOPS PLACE 25 SETS OF EA
    CH COLOR COMBO
250 REM J DETERMINES COLOR OF CHARACTER
260 FOR J=2 TO 16:FOR L=1 TO 3
270 REM M DETERMINES THE BACKGROUND COL
    OR FOR EACH ASTERISK
280 ON LGOTO 290, 300, 310
290 FORM=106 TO 32 STEP 74:GOTO 320
300 FORM=170 TO 32 STEP 138:GOTO 320
310 FORM=234 TO 32 STEP 202
320 FOR I=1 TO 25
330 GOSUB 380
340 NEXT I
350 NEXT M
360 NEXT L:NEXT J
370 GOTO 260
380 REM SUBROUTINE
390 REM RANDOMLY GENERATE COORDINATES
400 R=INT(RND(1)*12)+1
410 C=INT(RND(1)*20)+1
420 REM PLOT CHARACTER + 3 REFLECTIONS
430 Q1=40-R+C
440 Q2=40-R+(40-C)
450 Q3=40*(24-R)+C
460 Q4=40*(24-R)+(40-C)
470 POKE SC+Q1,M:POKE SC+Q1,J:POKE SC+Q2,M
    :POKE SC+Q2,J
480 POKE SC+Q3,M:POKE SC+Q3,J:POKE SC+Q4,M
    :POKE SC+Q4,J
490 RETURN

```

HCM

Little Stars

Two FOR-NEXT loops control the program. The first of these is in line 210. This loop controls whether the character to be placed on the screen is an asterisk or a space. The second loop causes 15 of those characters to be plotted. The routine that places the characters on the screen is quite easy: Two random numbers are selected in lines 230 and 240. Lines 250 and 260 calculate the screen address offset for the first character, based on the pair of coordinates. Four characters are then poked onto the screen in lines 270 through 300 using the mirror technique.

Color Kaleidoscope

This program is the next step in the evolution of the Kaleido Computer. The asterisks have been replaced with colored shapes. This version of the program is hard-ly different from Little Stars, except for lines 190, 370, 380, and 390.

TI-99/4A

```

100 REM *****
110 REM * LINEAR *
120 REM * KALEIDOSCOPE *
130 REM *****
140 REM BY MELODY COVINGTON
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM TI BASIC
180 REM
190 CALL CLEAR
200 CALL SCREEN(2)
210 REM ASSIGNS COLORS
220 CALL COLOR(14,8,8)
230 CALL COLOR(13,5,12)
240 CALL COLOR(12,4,3)
250 REM REDEFINE CHARACTERS
260 CALL CHAR(120,"33CC33CC33CC33CC")
270 CALL CHAR(128,"0466FFFF99FFFF6620")
280 REM LOOP FOR DIFFERENT COLORS
290 FOR A=136 TO 120 STEP -8
300 RANDOMIZE
310 REM LOOP TO CONTROL NUMBER OF LINE
    S DRAWN
320 FOR D=1 TO 10
330 REM RANDOMLY SELECTS BEGINNING COO
    RDINATES
340 R=(RND*24)+1
350 C=(RND*32)+1
360 REM RANDOMLY SELECTS SLOPE
370 E=(2*RND)
380 F=(2*RND)
390 REM LOOP TO COMPUTE COORDINATES OF
    THE LINE
400 FOR G=1 TO 15
410 R=(R+E)
420 C=(C+F)
430 REM STOPS LINE AT EDGE OF THE SCRE
    EN
440 IF R>24 THEN 540
450 IF C>32 THEN 540
460 IF R<1 THEN 540
470 IF C<1 THEN 540
480 REM DRAW LINE PLUS 3 REFLECTIONS
490 CALL VCHAR(R,C,A,1)
500 CALL VCHAR(R,33-C,A,1)
510 CALL VCHAR(25-R,C,A,1)
520 CALL VCHAR(25-R,33-C,A,1)
530 NEXT G
540 NEXT D
550 NEXT A
560 REM FOR DELAY
570 FOR M=1 TO 1000
580 NEXT M
590 GOTO 190

```

HCM

Line 190 sets up several variables to contain the POKE character code for the four characters to be used in each area of the screen. Lines 370 through 390 cycle those characters between the graphics shapes and a space character. Line 370 also selects a random color for plotting.

Linear Kaleidoscope

This final program takes graphics one step farther and places a form of control over drawing direction. In Linear Kaleidoscope the computer draws a line across the screen with a character.

Continued

VIC-20

```

100 REM *****
110 REM * LINEAR *
120 REM * KALEIDOSCOPE *
130 REM *****
140 REM BY MELODY COVINGTON
150 REM AND THE HCM STAFF
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 4.4.1
180 REM VIC 20 BASIC
190 PRINT "SHIFT CLR":A=7680:K=38400
200 POKE 36879,104
210 FOR T=1 TO 8
220 E=INT(RND(1)*3)
230 F=INT(RND(1)*3)
240 X=INT(RND(1)*11)
250 Y=INT(RND(1)*11)
260 P=2*(10-X)
270 Q=2*(10-Y)
280 POKEA=X+22*Y,160
290 POKEK=X+22*Y,CC
300 POKEA=X+P+22*Y,160
310 POKEK=X+P+22*Y,CC
320 POKEA=X+P+22*(Y+Q),160
330 POKEK=X+P+22*(Y+Q),CC
340 POKEA=X+22*(Y+Q),160
350 POKEK=X+22*(Y+Q),CC
360 FORTD=1 TO 50:NEXT
370 IF E=0 AND F=0 THEN 410
380 X=X+F:IF X>20 OR X<0 THEN 410
390 Y=Y+E:IF Y>20 OR Y<0 THEN 410
400 GOTO 260
410 CC=CC+1:IF CC=6 THEN 410
420 NEXT T
430 CC=0
440 FORT=1 TO 2000:NEXT:PRINT "SHIFT CLR"
      GOTO 210

```

HCM

IBM PC & PCjr

```

100 '*****
110 ' * LINEAR KALEIDOSCOPE *
120 '*****
130 ' BY MELODY COVINGTON
140 ' AND THE HCM STAFF
150 ' HOME COMPUTER MAGAZINE
160 ' VERSION 4.4.1
170 ' IBM PCjr WITH CASSETTE & CARTRIDGE
    BASIC
180 ' IBM PC WITH CASSETTE BASIC & BASIC
    A WITH
190 ' COLOR/GRAPHICS MONITOR ADAPTER
200 ' AND COLOR MONITOR
210 '
220 ' SET UP
230 RANDOMIZE TIMER
240 CLS:SCREEN 0:WIDTH 40
250 ' PLACE 25 SETS OF *, THEN PLACE 25
    BLANKS
260 M=219
270 FOR D=1 TO 20
280 COLOR INT(RND*16)
290 R=INT(RND*23)+1
300 C=INT(RND*39)+1
310 ' RANDOMLY SELECTS SLOPE
320 E=2*RND
330 F=2*RND
340 ' COMPUTE COORDINATES OF LINE
350 FOR G=1 TO 20
360 R=R+E
370 C=C+F
380 IF R>23 OR R<1 OR C>39 OR C<1 THEN
    450
390 ' DRAW LINE PLUS 3 REFLECTIONS
400 LOCATE R,C:PRINT CHR$(M);
410 LOCATE 24-R,40-C:PRINT CHR$(M);
420 LOCATE R,40-C:PRINT CHR$(M);
430 LOCATE 24-R,C:PRINT CHR$(M);
440 NEXT G
450 NEXT D
460 ' DELAY LOOP
470 FOR TD=1 TO 5000:NEXT:GOTO 230

```

HCM

COMMODORE 64

```

100 REM *****
110 REM * LINEAR *
120 REM * KALEIDOSCOPE *
130 REM *****
140 REM BY MELODY COVINGTON
150 REM AND THE HCM STAFF
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 4.4.1
180 REM C-64 BASIC
190 REM
200 PRINTCHR$(147):SC=1023:CS=55295
210 REM ASSIGN COLORS FOR EXTENDED BACK
    GROUND COLOR MODE
220 POKE53265,PEEK(53265)OR64:POKE53280
    ,0:POKE53281,0
230 POKE53282,3:POKE53283,10:POKE53284,
    7
240 REM LOOP FOR DIFFERENT COLORS
250 FOR J=2 TO 16
260 PRINTCHR$(147)
270 REM LOOP TO CONTROL NUMBER OF LINES
    DRAWN
280 FOR D=1 TO 4:FOR M=106 TO 234 STEP
    64
290 REM RANDOMLY SELECTS BEGINNING COOR
    DINATES
300 R=INT(RND(1)*12)+1
310 C=INT(RND(1)*20)+1
320 REM RANDOMLY SELECTS SLOPE
330 E=INT(RND(1)*3)
340 F=INT(RND(1)*3)
350 IF (F=0) AND (E=0) THEN E=1:F=1
360 REM LOOP TO COMPUTE COORDINATES OF
    LINE
370 FOR G=1 TO 15
380 R=R+E
390 C=C+F
400 REM STOPS LINE AT EDGE OF SCREEN
410 IF R>24 THEN R=24
420 IF C>39 THEN C=39
430 IF R<1 THEN R=1
440 IF C<1 THEN C=1
450 REM DRAW LINES PLUS 3 REFLECTIONS
460 Q1=40*R+C:POKESC+Q1,M:POKECS+Q1,J
470 Q2=40*R+(40-C):POKESC+Q2,M:POKECS+Q
    2,J
480 Q3=40*(24-R)+C:POKESC+Q3,M:POKECS+Q
    3,J
490 Q4=40*(24-R)+(40-C):POKESC+Q4,M:POK
    ECS+Q4,J
500 NEXT G
510 NEXT M:NEXT D
520 REM FOR DELAY
530 FORN=1 TO 1000
540 NEXT N
550 NEXT J
560 GOTO 200

```

HCM

APPLE II Family

```

100 REM *****
110 REM * LINEAR KALEIDOSCOPE *
120 REM *****
130 REM BY MELODY COVINGTON
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM APPLE II FAMILY APPLESOFT
    HOME : GR
180 POKE -16302,0
190 FOR Y=40 TO 47
200 FOR X=0 TO 39
210 COLOR=0:PLOT X,Y
220 NEXT X:NEXT Y
230 REM LOOP TO CONTROL COLORS
240 FOR A=1 TO 15
250 COLOR=A
260 REM LOOP TO CONTROL NUMBER OF LINE
    S DRAWN
270 :
280 FOR D=1 TO 10
290 REM RANDOMLY SELECT BEGINNING COOR
    DINATES
300 R=INT(RND(1)*39)
310 C=INT(RND(1)*47)
320 REM RANDOMLY SELECT SLOPE
330 E=2*RND(1)
340 F=2*RND(1)
350 REM LOOP TO COMPUTE COORDINATES OF
    THE LINE
360 FOR G=1 TO 15
370 R=(R+E)
380 C=(C+F)
390 REM STOP LINE AT THE EDGE OF THE S
    CREEN
400 IF R>39 THEN 510
410 IF C>47 THEN 510
420 IF R<0 THEN 510
430 IF C<0 THEN 510
440 REM DRAW LINE PLUS 3 REFLECTIONS
450 PLOT R,C
460 PLOT R,47-C
470 PLOT 39-R,C
480 PLOT 39-R,47-C
490 NEXT G
500 NEXT D
510 NEXT A
520 REM DELAY ROUTINE
530 FOR M=1 TO 1000
540 NEXT M
550 GOTO 180

```

HCM



Little Stars

Applesoft BASIC has many built-in graphics commands. This program makes use of the **PRINT** statement, the **VTAB**, and the **HTAB** commands to display graphics on the screen. The first (outer) loop begins in line 210. This loop decides whether an asterisk or a space is printed. Lines 220 and 230 check the value of **X**. If **X** is set to 1, then an asterisk is printed—otherwise a space is printed. The next loop (inner) controls the number of characters printed to the screen. In this case 25 asterisks will be printed, then 25 spaces, then 25 more asterisks, and so on.

The **VTAB** and **HTAB** commands are used to position the print statement on the screen. The algorithm is the same as the one used on other systems, except the commands have changed.

Color Kaleidoscope

In this program we decided to make use of Apple's low resolution graphics screen, which perfectly fits this application. We want to plot little boxes of color all over the screen. Line 200 selects this mode. The short routine in lines 220 through 250 clears out the bottom of the low-res screen.

Linear Kaleidoscope

This program also uses the low resolution graphics screen. It plots a low-res line across the screen, and mirrors it in all four corners. The result is well worth the short amount of time it takes to key in. Of course, the real beauty of this program is its simplicity—the line-drawing algorithm couldn't be any easier to implement.

Lines 340 and 350 select two random numbers to be the horizontal and vertical slope for the line, and determine the direction of travel. In the loop that starts in line 370, the two slope values are added to the current position of the line. This process is repeated either 15 times or until the line runs into the side of the screen.



Cassette BASIC, BASICA, Cartridge BASIC

Little Stars

This program displays its graphics with the simple **PRINT** statement. The printing location is handled with the **LOCATE** statement, which lets you position the cursor at any location on the screen. Because Text mode is used, all 16 colors are available, even with Cassette BASIC.

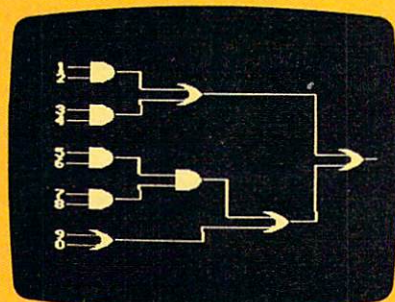
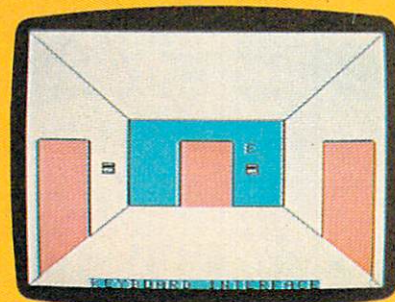
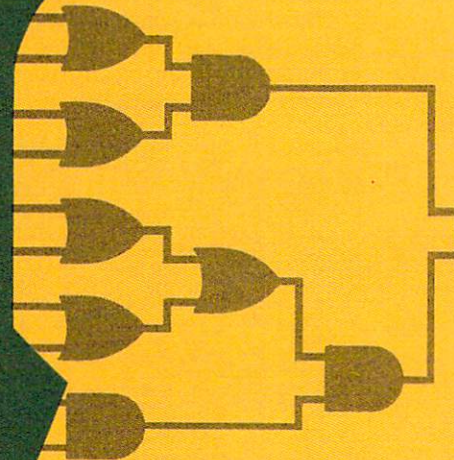
Color Kaleidoscope

Only one line was added and one line changed to turn *Little Stars* into *Color Kaleidoscope*. Line 260 is new—it changes the color of the characters being printed. Line 250 was modified to print either a solid block or a space. The rest of the program remains the same.

Linear Kaleidoscope

This program is computer graphics at its best. The code is so elegantly simple, yet it yields a spectacular result. There are two primary loops in this program. The first loop, which starts in line 270, controls the number of lines drawn before clearing the screen to start over. The second (inner) loop controls the length of the lines. A line will continue to a length of 20 characters, unless it reaches the side of the screen first. If that happens, then the line will terminate, and the next line will start.

HCM





THE BOOLEAN BRAIN

by **W.K. Balthrop**
HCM Staff

*Captured by your computer,
you wind up exploring
its inner workings.*

[Editor's Note: In the August issue of *Home Computer Magazine* we featured "The Boolean Brain" for the IBM PCjr and Apple II family. We bring it back this month for those readers with TI-99/4A and Commodore 64 computers.]

It is late. Another night of blasting aliens draws to a close. You reach to turn off the computer—but it's not through playing! Suddenly, before your hand can touch the switch, the screen flashes bright red. Then the message CPU Error appears briefly and vanishes, leaving the screen totally blank.

What has happened? Has your computer died? You pounce on the keyboard, hoping to save your system before it's too late. Instead, a tingling surge of electricity grabs and holds your arms fast. With horror, you realize something is pulling you in, in . . .

This must be a nightmare you think. But when you open your eyes, there's a new shock awaiting: You are in a strange, brightly lit room—a room that looks remarkably like the inside of a computer. Thus, stranded in the *Keyboard Room*, you suddenly recall the message about a CPU failure. Perhaps if you can make it to a room with the Central Processing Unit, you can fix the problem and get out of your silicon cell.

The Program

The *Boolean Brain* program is a combined adventure game and "logical" learning experience. Your goal is to find the computer's CPU. To do this, you will have to open the locked doors of each room, and gain access to other rooms. Each door is secured with a logic lock.

As the game begins, you start out in the *Keyboard Room*. Here you will see a three-dimensional picture on the screen of three of the four walls. In each wall is a door, and to the right of each door is a control panel. On the wall in the center of the screen is one of four letters which indicate the direction you are facing. To move in any of the four directions, simply press either E, W, N, or S. Closed doors are red, and when you try to go through one you will be taken to another screen. This screen will display the computer logic gates that you must activate to open the door.

*"This must be a nightmare you think.
But when you open your eyes,
there's a new shock awaiting . . ."*

The two types of logic gates used in the lock look and operate quite differently. The AND gate—with its left side squared off—requires both of its inputs to be turned on before it will pass its output. The OR gate—resembling an arrow head—will pass its output when either of the two inputs are turned on. The output of the first AND and OR gates will feed the input of other AND and OR gates. The logic paths that are turned on will become green, and the lock will open when you have succeeded in completing a logic path to the right side of the screen.

On the left side of the screen are 10 input lines to the five gates. To activate an input, you simply press the number on the keyboard for the input line you desire. You want to open a path with the fewest number of inputs possible. The computer keeps track of how many inputs you use throughout the game, so if you don't learn to be a "Boolean Brain," you may not be able to escape from the computer.

(One word of warning to those who venture carelessly: There is one trap hidden in the game . . . beware bad disk sectors.)

HCM

Boolean Brain (TI-99/A) Explanation of the Program

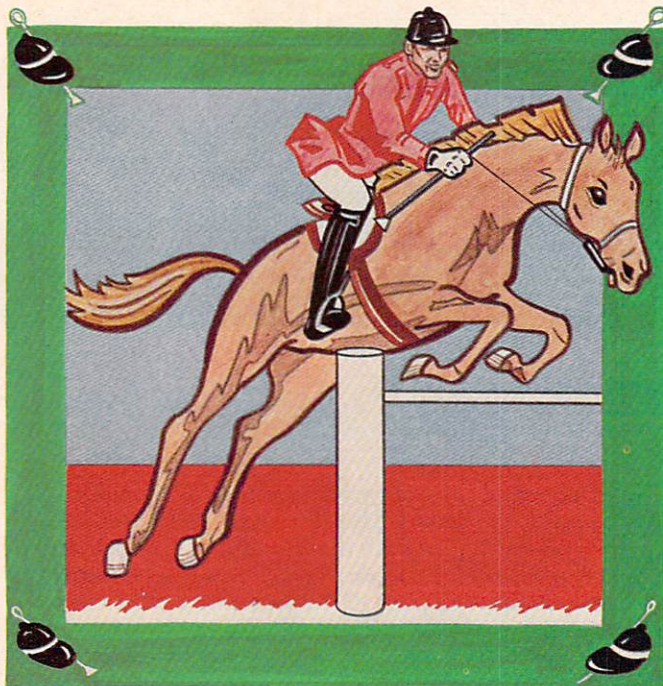
Line Nos.	
100-170	Program header.
180-250	Initialize program variables.
260-270	Input direction.
280-310	Hit a bad disk sector.
320-350	Go through open door.
360-420	Display computer rooms.
430-490	Display logic gates, and get input.
500-770	Gate control logic.
780-910	Found control room. End of game and option to play again.
920	Subroutine to read the keyboard.
930-1310	Program DATA.

For the Key-In listing see HCM PROGRAM LISTINGS Contents on page 93.

Boolean Brain (C-64) Explanation of the Program

Line Nos.	
100-170	Program header.
180-220	Initialize program variables.
230-910	Draw the computer rooms, and input direction.
920-940	Check for an open door.
950-1190	Draw the logic gates, and get input.
1200-1610	Gate control logic.
1620-1970	Subroutines to draw the gates.
1980-2340	Found the control room. End of game, and option to play again.
2350-2420	Sound routines.
2430-2470	Program DATA.

For the Key-In listing see HCM PROGRAM LISTINGS Contents on page 93.



STADIUM

by Kent and Kathy Gemmel
and the HCM Staff

*This BASIC program captures
an age-old sport in a
simple simulation game.*

its right, meaning, as the horse sees right. If the horse is heading down toward the bottom of the screen, the horse's right will be your left. This may seem confusing at first, but after you have played a few games it will come as second nature.

This is no rodeo. Riders sit erect and without expression, their caps and uniforms spotless and perfect in every detail. Their well-groomed horses stand in formal posture. As a fine horse and its rider glide through the obstacle course of high fences, every movement is precise, deliberate, and subject to judgement.

Stadium Jumping is an equestrian simulation game in which you must ride your horse through a pre-planned course and jump fences—preferably without knocking down the poles or yourself in the process. You are scored on how many "faults" you have. Each time you knock down a pole you will receive four faults. A perfect score is 0, or no poles knocked down. If you fall from your horse, you won't be able to get back on to complete the course, and will have to start over again. Here are the general rules of *Stadium Jumping*:

1. All fences must be jumped in the proper order. Fences are numbered according to the order in which they must be jumped.
2. All fences must be jumped in the correct direction. A flag is posted on the side of every fence and, in some systems, this flag may be the sequence number itself. You must jump the fence squarely, with the flag to your horse's right.
3. Attempting to make the horse jump without a fence will cause you to fall from your horse. Simply missing the fence will also cause you to fall.
4. If you fail to jump a fence and instead collide with it, you will be faulted. You also run a high risk of falling from your horse.
5. After you have jumped the last fence, your score will be displayed.
6. You may run your horse around the screen as much as you like without penalty.

Three Skill Levels

Three skill levels are available in this game. The first requires only four jumps to complete a round. You have 7 fences to jump on the second level, and the third level has 11 fences and is for masters only.

Horse's Point Of View

Only two keys on the keyboard, or two directions on the joystick, turn the horse. When you press the right button or pull the joystick right, the horse will turn to

When the game begins, your horse will be in the upper left corner of the screen. To start him off, press K for Kick whether you are using the keyboard or a joystick. The horse will start running, and will not stop until the game is over.

KEYBOARD	ACTION
S.....	Turn horse to its left.
D.....	Turn horse to its right.
J.....	Make horse jump.
K.....	Kick horse to start the course.
JOYSTICK	ACTION
STICK LEFT.....	Turn horse to its left.
STICK RIGHT.....	Turn horse to its right.
FIRE BUTTON.....	Make horse jump.

Stadium Jumping (TI-99/4A) Explanation of the Program

Line Nos.	
100-180	Program header.
190-290	Initialization.
300-380	Input skill level.
390-520	Get graphics shapes and color assignments.
530-1780	Display initial arena and fences for each skill level.
1790-1910	Start of game. Wait for K to be pressed.
1920-2000	Scan keyboard, check for a jump.
2010-2180	Check for a direction change.
2190-2460	Change direction and horse shape routines.
2470-2740	Routine for fouls.
2750-2880	Rider has fallen.
2890-3380	Routine to jump the horse.
3390-3580	Finished round.
3590-3700	Option to play again.
3710-3820	Input new level, start new game.
3830-3870	Routine to print without scrolling.
3880-3910	Graphics character and color assignment data.

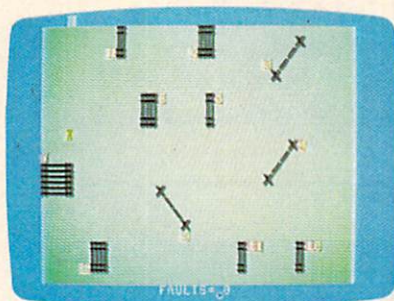
For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.



Once you have selected one of the three skill levels, the arena will appear on the screen, with your horse waiting in the upper left corner. To start the horse, press any key—the horse will continue across the arena by itself.

Turn the horse by using the left and right cursor control keys. Once you hear the beep, you can let go of the key. However, the horse may not turn instantly, because

JUMPING



the program will still have to go through its paces before it can update the horse's shape and direction.

If you prefer to play the game without sound effects, press (F1) on the PC,

and (Fn)(1) on the PCjr. Pressing this key again will turn the sound effects back on.

KEY	ACTION
LEFT Cursor	Turn horse to the left.
RIGHT Cursor	Turn horse to the right.
UP Cursor	Make horse jump.
(Fn)(1)	Turn sound off or on.
Any Key	Kick horse to start game.

Stadium Jumping (IBM PC/PCjr) Explanation of the Program

Line Nos.	
100-200	Program header.
210-350	Initialize program graphics and variables.
360-370	Get skill level. Branch to display arena.
380	Wait for a key to be pressed. Start the game.
390-420	Main control loop.
430-460	Interrupt routines.
470-670	Routine to display the three arenas.
680-1010	Routine to jump the horse.
1020-1140	Routine to turn the horse.
1150-1250	Move horse and check for collisions.
1260-1330	Check to see if jump went over the right fence.
1340-1350	Foul.
1360-1390	Rider has fallen.
1400-1410	Option to play again.
1420-1440	Round has been completed.
1450	Subroutine to read the keyboard.
1460	Get information for the skill level from the data statements.
1470-1510	DATA for fences.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.



The Apple version of Stadium Jumping has an added feature not found in the other versions: While playing the game, you can change the speed of the horse by pressing keys 1 through 5.

KEY	ACTION
LEFT Cursor	Turn horse to its left.
RIGHT Cursor	Turn horse to its right.
SPACE BAR	Make the horse jump.
K	Kick horse to start the game.
1 through 5	Change horse's speed. Default=2.

JOYSTICK	ACTION
Stick left	Turn horse left.
Stick right	Turn horse right.
Fire button	Make horse jump.

Stadium Jumping (Apple II Family) Explanation of the Program

Line Nos.	
100-190	Program header.
200-330	Initialize program and save machine code routines to memory.
340-380	Input options.
390-440	Get skill level.
450-1040	Set up initial arena and display fences for each level. Branch to display fences.
1050-1070	Start game. Wait for K to be pressed.
1080-1180	Read keyboard and joystick.
1190-1260	Routines to turn the horse.
1270-1390	Foul routine.
1400-1420	Rider has fallen.
1430-1710	Routine to jump the horse.
1720-1740	Round complete.
1750-1770	Option to play again.
1780-1790	Display number of fouls on the high-res screen.
1800-1810	Display a flashing border on the screen.
1820	Display the current number of gates completed.

**NOTE: A word of caution to anyone who may want to resequence this program in the future: A machine language routine is in use which restores DATA statements to a particular line number so that they can be read again. This routine is called RESTR, and it starts at address 2138. If you resequence this program, you will need to change any line references where CALL RESTR is used. The value passed in the routine is the line number containing the DATA statement being tested.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.



When the arena is first displayed, your horse will be waiting in the upper left corner of the screen. To start the game and make your horse enter the arena, press K to kick the horse. Once the horse starts moving, it will not stop until the end of the round.

KEY	ACTION
S	Turn horse to its left.
D	Turn horse to its right.
J	Make the horse jump.
K	Kick the horse to start the round.

JOYSTICK	ACTION
Stick left	Turn horse to its left.
Stick right	Turn horse to its right.
Fire button	Make the horse jump.

Stadium Jumping (C-64) Explanation of the Program

Line Nos.	
100-170	Program header.
180-250	Title screen.
260-310	Get skill level.
320-400	Define graphics characters.
410-1050	Draw arena and fences.
1060-1110	Start game. Wait for K to be pressed.
1120-1350	Read keyboard and joystick.
1360-1470	Update position pointers and character shape.
1480-1680	Crash routines. Do faults.
1690-2010	Jump routine.
2020-2120	End of the round. Option to play again.
2130-2180	Enter level for a new game.
2190-2270	Graphics data for character shapes.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

HCM

MARKET

A

D

N

E

S

S

Shouts arise everywhere from the boisterous—even frantic—crowd. Many wave small sheets of paper, others yell into telephones as the numbers change on the lighted board.

This is the world of profit and loss—the instant buy and sell decisions that amass paper fortunes and shatter dreams . . .

Could you survive here?

by Brian Lee
and the HCM Staff

Sit down to a real investment—your computer—and put your market skills to the test. *Market Madness* is a game that simulates stock transactions for six companies. Up to ten people can play at one time, each with their own personal portfolio. Each player begins the game with \$5000 cash, and one turn is the equivalent of one week of time. The game can last from two to 999 weeks. Players can buy and sell stock from the exchange, at exchange rates, or they can buy and sell to each other, setting their own prices. There is even a bank that will loan you money, if you're not a bad credit risk.

Playing The Game

The main game screen displays the stock prices and a menu for the first player. The stock price section of the screen includes the names of the stocks, their current value, and the

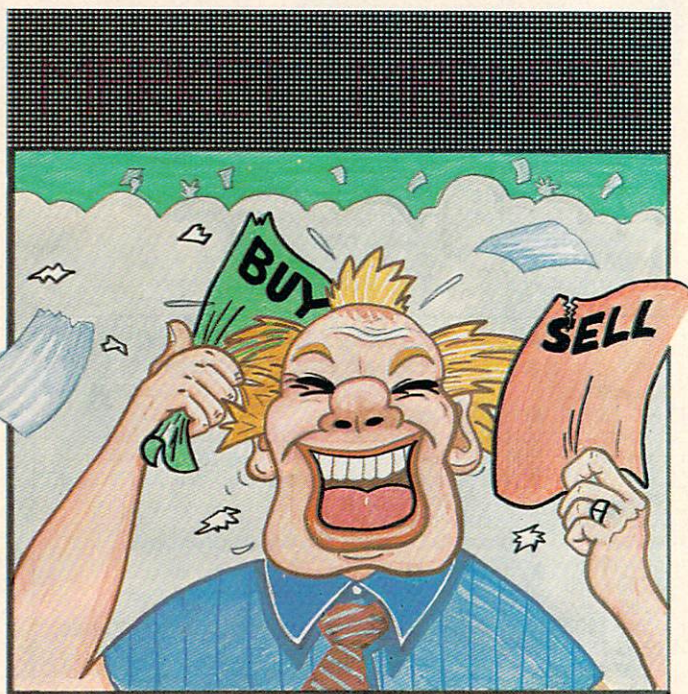
number of shares of each stock you currently own. The menu at the bottom of the screen lists six options, which can be selected by typing the first letter of the desired option:

B)uy	S)ell
T)rade	L)oan
N)ext	P)ortfolio

B)uy Stocks

This option lets you purchase stocks at the going rate displayed on the screen. The first prompt will ask you which stock you wish to buy. Enter a number from 1 through 6 to indicate the desired stock. The only time you will not be allowed to buy a stock under any circumstances is when the company is bankrupt. (See the section on bankruptcy). The second prompt will ask you how many shares you wish to buy. If you have enough money to make the purchase, the message TRANSACTION COMPLETE will be displayed.

Any time you are prompted to enter a number, you can also type 0 (zero) and return to the menu screen. At that point, you may notice that there was more



money deducted from your account than what you should have paid for the stock. This extra charge is the broker's fee. The only time you don't have to pay a broker's fee for exchanging stocks is when trading with other players or cashing in stocks at the bank.

S)ell Stocks

This option works the same as the Buy Stocks option. You are prompted for the stock you wish to sell; if the company is bankrupt or if you don't own any of that stock, you will not be able to sell it. If you can sell the stock, then you are asked for the number of shares you wish to sell. You can enter all or part of your holdings in that stock, but of course, you can't sell more shares than you own. The money from the sale, minus the broker's commission, is then transferred to your cash assets.

T)rade Stocks

This option takes you to another menu screen where you can select one of three options:



1. Trade to another player.
2. Trade for another stock.
3. Cash shares into the bank.

—1. Trade to another player.

If you are the only player in the game you can't use this option. After selecting the trade option you will be asked to choose someone to trade with. Sorry, you can't trade with yourself. After selecting someone to trade with, you will be asked if you want to sell to that person, or buy from that person. When trading stock, players can set their own price for it, within certain limits. The price can't exceed twice the market value, or be less than one half the market value. After the seller enters a price for the stock, the buyer will be asked whether he or she agrees with the transaction. The players then bargain, and if they don't agree, they are taken back to the main menu.

—2. Trade for another stock.

In this option, you will be able to trade stock you currently own for any other stock straight across, as long as the company is not bankrupt.

When trading stock, the value of the stock you trade will not always be evenly divisible by the cost per share of the stock you want to receive. Thus, any money left over from the trade will be transferred to your cash assets. You must always trade enough stock to receive at least one share of the new stock.

"The old adage holds true: buy low and sell high. However, this practice is not without its price."

—3. Cashing into the bank.

You may sell your stock to the bank once per turn. You will be asked to enter the name of the stock you want to cash in, and the number of shares you wish to sell. The bank will then make an offer on the stock which may be a little higher or lower than its market value. If you decline the offer, then the program will return to the trading menu screen. If you accept the offer, the money you receive will be added to your cash assets.

L)ans

Selecting this option takes you to another menu screen where you can select one of four options:

1. Take out a loan
2. Pay back a loan
3. Compound interest on a future loan
4. Main menu (Apple and C-64)
0. Main Menu (TI and IBM)

—1. Take out a loan

If you select this option, you will be prompted for the amount you wish to borrow. You can't borrow more than your credit limit, which is calculated to be your total net worth or \$5000, whichever is greater. If your credit limit is \$5000, and you already have a loan out for \$4000, then you would only be eligible to borrow another \$1000. You will never have a credit limit below \$5000, and are not penalized if your total net worth (credit limit) drops below your current loan balance. This simply means that you will not be able to take out any more loans until your credit limit once again exceeds your loan balance.

Each week during the game you will have to pay

interest on the balance of your loan. The interest you pay is automatically deducted from your cash assets at the end of the week. If you don't have any cash assets, the interest will be tacked on to your loan balance. This is the only instance in which your loan balance can be increased above your credit limit. No automatic payments are made on your loan balance; however, it is in your best interest to pay off your loans as soon as possible. Your outstanding loans will be deducted from your net worth at the end of the game when the scores are displayed.

—2. Pay back a loan

If you are not in debt when you select this option, you will be advised that you do not owe any money. Otherwise, you will be asked for the amount you wish to repay. You can only use your cash on hand or cash assets to pay back a loan. If you overpay your debt, only the amount you owed will be deducted from your cash assets. This is a handy feature for those who may be a little overzealous to cancel their debts.

—3. Compound interest on a future loan.

This option lets you calculate how much interest must be paid on a loan for the remaining weeks in the game. The calculation assumes that you intend to keep the loan at its present balance for the duration of the game. You will not be taking out a loan with this option—it is meant as a calculator only, to estimate future interest payments.

WALL STREET		WEEK: 1
PORTFOLIO		
STOCK PLUS	CURRENT PRICE	NO. OF SHARES / ORIGINAL PRICE
1. US STEEL	\$45	0 / \$0
2. PAK AM	\$17	0 / \$0
3. FORD	\$24	0 / \$0
4. SANVO	\$42	0 / \$0
5. XEROX	\$44	0 / \$0
6. AT&T	\$46	0 / \$0
CASH: \$5000		INVEST: \$0
LOANS: \$1000		TOTAL: \$5000
PRESS ANY KEY: N		

N)ext

This option from the main menu terminates your turn, but you will get one chance to back out first and continue your turn. When your turn is over, it will become the next player's turn; if you are the last

player in the round, then it will also be the end of the week, and the first player's turn will occur next, starting a new week.

P)ortfolio

The Portfolio screen displays a player's current financial status. Included in the display are the current market price, the number of shares invested, and the net worth of your investment in each stock. Also displayed are your current cash assets, total investment worth, loan balance, and total net worth. Your total net worth is actually your cash assets plus the value of your investments, minus any loans you may have. This value is used to adjust your credit limit when you apply for a loan.

There are two ways to display the Portfolio screen. If you press P while at the main menu, the Portfolio screen will be displayed. To return to the main menu, press either [RETURN] or [ENTER], depending on your system.

Whenever you buy or sell stock directly with the market, the Portfolio screen will be displayed after a successful transaction. If you have a TI or IBM computer, you may return to the main menu by pressing [ENTER]. On the Apple and C-64 you will get a prompt saying ANOTHER TRANSACTION (Y/N)? If you

enter Y to this prompt, you will be taken back to the menu screen. If you enter N your turn will end, and the next player's turn will start.

The Smart Marketeer

After playing the game for awhile you may notice a pattern or trend for certain stocks. The program is written to simulate real market situations as closely as possible. Stocks generally do not wildly fluctuate back and forth at random—they generally exhibit short term trends.

The two important influences in the program are the general trend adjustment, and the activity adjustment. The general trend indicates in which direction the stock is likely to move (up or down). All trends have a life span of four weeks. After four weeks, a stock's trend will take on totally new random values.

The activity factor indicates the fluctuation of a stock (how much it will change each week). The most any stock can change in one week is 20 points (or \$20 in value).

Stock Splits

When a stock reaches a value of more than \$150, the stock will split. This means that the value of the stock will be cut in half, and you will receive double the number of shares for that stock. You will lose nothing in value when this happens—in fact, it's to your advantage that it does happen every once in a while. Because the stocks are limited to how much they can

fluctuate, expensive stocks do not make very much profit. If you have 10 shares of a stock worth \$150 per share, you would have \$1500 invested.

If that stock went up \$10 per share, your \$1500 investment would earn you \$100 in profit. However, if you had 10 shares of a stock worth \$20, your investment would only be \$200. Then if the stock went up \$10, you would make \$100 in profit on a \$200 investment.

Bankruptcy

Occasionally a company will go bankrupt. This happens when a stock's value reaches \$0 (zero dollars). If you have any shares in a company at the time it goes bankrupt, they will be lost and your holding in the stock will be reset to zero shares. When a company is bankrupt you can't buy or sell its stock. It may recover from bankruptcy, but you will not get back those shares lost when bankruptcy occurred.

Strategy

The old adage holds true: buy low, sell high. However, this practice is not without its price. Keep in mind that stocks with a value of \$20 or less could conceivably go bankrupt in only one week. There is a good chance that you could lose everything you invested in such a stock. When investing in speculative stock it is a good idea to diversify your investments. (Don't put all your eggs in one basket.) Then, if one stock goes under, you will still have several other investments to keep you going.

HCM

Market Madness (Apple II Family) Explanation of the Program

Line Nos.	
100-170	Program header.
180-290	Initialize program.
300-600	Display the main menu.
610-710	Buy stock routine.
720-810	Sell stock routine.
820-920	Main menu for trading.
930-1370	Trading with other players.
1380-1480	Cash stock into the bank.
1490-1610	Trade for another stock.
1620-1730	Main menu for loans.
1740-1820	Take out a loan with the bank.
1830-1930	Pay back a loan.
1940-2020	Calculate interest on a loan.
2030-2070	Routine for the Next option.
2080-2300	Display the portfolio.
2310	Routine for the continue prompt.
2320-2350	Update for end of the week.
2360-2500	End of the game routine.
2510-2550	Routine to PEEK the keyboard.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

Market Madness (C-64) Explanation of the Program

Line Nos.	
100-160	Program header.
170-360	Initialize program.
370-820	Main game menu.
830-970	Buy stock routine.
980-1110	Sell stock routine.
1120-1240	Main menu for trading.
1250-1820	Trade with other players.
1830-1970	Cash stock into the bank.
1980-2130	Trade for another stock.
2140-2260	Main menu for loans.
2270-2360	Take out a loan from the bank.
2370-2520	Pay back a loan to the bank.
2530-2640	Calculate interest on a loan.
2650-2690	Routine for the Next option.
2700-2980	Display portfolio.
2990-3200	End of turn, week, and game routines.
3210-3330	Cursor routine—products rotating cursor.
3340-3740	Display instructions.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

Market Madness (IBM PC/PCjr) Explanation of the Program

Line Nos.	
100-190	Program header.
200-290	Initialize the program.
300-320	Main game menu.
330-400	Buy stocks routine.
410-470	Main menu for trading.
480-860	Trade with other players.
870-980	Trade for another stock.
990-1080	Cash into bank.
1090-1190	Next turn.
1200-1270	Sell stocks routine.
1280-1460	Routines for the loans section.
1470-1540	Display the portfolio.
1550-1640	End of the game.
1650-1710	Keyboard routines.
1720-1780	Display the main game menu screen.
1790-1820	Program data.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

Market Madness (TI-99/4A) Explanation of the Program

Line Nos.	
100-180	Program header.
190-290	Initialize the program.
300-310	Main control loop for main menu screen.
320-410	Buy and sell stock.
420-450	Trading main menu.
460-780	Trade with other players.
790-870	Trade for another stock.
880-960	Cash into bank.
970-990	Loans main menu
1000-1030	Borrow from the bank.
1040-1100	Pay back loan.
1110-1140	Interest calculation.
1150-1280	End of turn, and week routines.
1290-1320	Display Portfolio screen.
1330-1390	End of the game.
1400-1450	Display main menu screen.
1460	Key input subroutine.
1470	Routine to clear part of the screen.
1480	Image format.
1490	Game data.
1500-1510	Time delay subroutine.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

HOME COMPUTERTM

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:

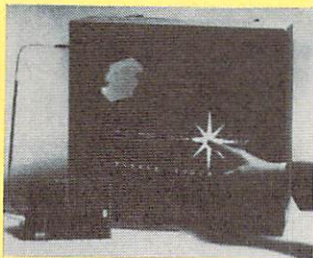
Product News Editor
Home Computer Magazine
1500 Valley River Drive., Suite 250
Eugene, OR 97401



A Touch of Glass

Kit Converts Monitors to Touch Screens

Interaction Systems, Inc. has announced a digitized X-Y Touch Sensor for Sony-type CRTs and monitors. Designated the TK-2000 Series Kits, the units provide an X-Y coordinate output with 100 part resolution when a person touches the tempered glass faceplate. Output is through an RS232 serial port. The Digitized X-Y Touch Sensor is available in 12, 15, and 19 inch sizes starting at \$660.



15, and 19 inch sizes starting at \$660.

Interaction Systems, Inc.
24 Munroe St.
Newtonville, MA. 02160
(617) 964-5300



More Utility from Your TI

Three New Assembly Language Programs for the TI-99/4A

Three high-speed assembly language programs were released by StarSoft for the TI-99/4A. Microkey provides ten user-defined function keys in TI BASIC or Extended BASIC. Each of ten control keys may be assigned a 28-character string consisting of a BASIC command that will be automatically entered when the corresponding key is pressed. Microkey's list price is \$19.95. Nibbler is a fast sector-by-sector disk copier and formatter. It

contains options to copy only certain sectors, to write to a different sector number than the one read from, and to format the destination disk. Nibbler's list price is also \$19.95. Unprotector allows the user to "unprotect" protected TI Extended BASIC programs while in memory. This allows users to backup and edit protected programs, and to transfer protected programs from tape to disk. Unprotector lists for \$14.95.

StarSoft
601 Alleghany St.
Blacksburg, VA. 24060



Increasing Your Power

Expanding Memory, I/O, & Disk Storage on TI Systems

Myarc, Inc. has released their newly developed MPES/50 systems. The MPES/50 System is a mini peripheral expansion system for the TI-99/4A, with 32K bytes of expansion memory, an RS232 serial port and parallel I/O port, a floppy disk controller, and a double-density, single-sided disk drive. It retails for \$595.

A two drive model, the MPES/50-2, comes equipped

Myarc, Inc.
P.O. Box 140
Basking Ridge, NJ 07920
(201) 766-1700

with two double-density, single-sided disk drives. It retails for \$785. A double-sided drive option is available for an additional \$50. For cassette memory storage systems, the MPES/50-RPM comes with all of the above except the disk controller and drive. This system can be upgraded later to full MPES/50 capability. It retails for \$299.



Extensions for the Home Accountant

Popular Financial Software Comes to Mac & Junior

The best-selling program The Home Accountant by Continental Software is now available for the IBM PCjr and the Apple Macintosh. On the PCjr, the program tracks up to five checkbooks, all cash and credit card transactions, and up to 100 budget categories. It will also print checks and a variety of reports. The Home Accountant jr costs \$74.95. On the Macintosh, the program will track any number of check-

Continental Software
11223 South Hindry Ave.
Los Angeles, CA. 90045
(213) 417-8031

book accounts, record 25 monthly automatic transactions, flag tax items, and enter monthly budgets for assets, credit cards, liabilities, income, and expense categories. It includes a financial calculations module which allows the user to calculate loans or determine the future value of a specific monthly investment. The Home Accountant for the Macintosh costs \$99.95.



Tiny Turtles

Cassette-Based LOGO for Bare-Bones 99/4A

Microcomputers Software has announced TINY LOGO on cassette for the TI-99/4A. No extra memory is required. Like bigger versions of LOGO, TINY LOGO uses turtle graphics to teach principles of programming. The software Microcomputers Software
34 Maple Ave.
Armonk, NY. 10504
(914) 273-6480

package comes with a 32-page instruction booklet featuring samples of simple and recursive procedures and a summary of TINY LOGO terminology. It is priced at \$19.95. Versions for other home computers will soon be released.



A Guide Ilc You Through

A Book for the Portable Apple

Bantam Books has published *The Apple Ilc Book* by Bill O'Brien. Written for new Apple Ilc computer buyers and experienced Apple users, the book answers users' questions

Bantam Books
666 Fifth Ave.
New York, NY. 10103
(212) 765-6500

about compatibility, configuring the system, and adding peripherals. It also includes information on DOS 3.3 and the new ProDOS. The paperback book is priced at \$12.95.



Muppet-Friendly Keyboards for Kids

Apple/Commodore Accessory Simulates School Desk

Children age three and up can learn letters, numbers, and colors with the assistance of the Muppets, featured on Muppet Learning Keys, Kids' Computer Keyboard. Developed by Koala Technologies Corp., the keyboard simulates the contents of a child's school desk to help children learn basic skills. The 14" x 15" three-pound keyboard connects to a computer display screen through the paddle port on the Apple IIe or Ilc, or the joystick port on the Commodore 64. Each section of the desk—penmanship slate, paint set, arithmetic exercise book, etc.—can be activated by the touch of a child's finger. Miss Piggy,



Gonzo, Fozzy Bear, and Kermit help provide instruction. Muppet Learning Keys is priced at \$79.95.

Koala Technologies Corp.
3100 Patrick Henry Drive
Santa Clara, CA. 95052-8100
(408) 986-8866



Software Explosion from Great Lakes

Games and Graphing for TI Users

Super Bargraphs, Lunar Cavern, and Funhaus are three new programs for the TI-99/4A released by Great Lakes Software. Super Bargraphs features graphing of up to 15 items at a time with labels, automatic scale adjustments, accuracy to one pixel, printout capability,

and more. It is available in BASIC or Extended BASIC versions, on cassette for \$14.95 and on disk for \$16.95. Lunar Cavern and Funhaus are arcade-style games for use with Extended BASIC, and are the same prices as Super Bargraphs.

Great Lakes Software
P.O. Box 241
Howell, MI 48843



Take Stock of Your Investments

Portfolio Management Software for Home Machines

Basic Byte, Inc. has introduced a portfolio management system of three volumes that work independently. Stock Management Vol. I is for the investor's personal stock portfolio. It allows the instant update of current value of up to 100 individual stocks, calculates long and short term capital gains and losses, and records dividends. Stock Management Vol. I for the Commodore 64, VIC-20, and Atari retails for \$39.95. The IBM PC version retails for \$59.95, and an Apple version is forthcoming.

Options Management Vol. II is for the speculative

options trader, and incorporates the record-keeping features of Stock Management Vol. I for use with an options portfolio. It also determines the fair market values of any stock option using the "Black-Scholes" model. It is available for the C-64 for \$39.95, and versions for the IBM PC, Apple, and Atari are in the works.

Graphic Analysis Vol. III, scheduled for fall release for the C-64, IBM PC, Apple, and Atari computers, is a graphics program designed to track stock market trends as well as an individual stock's performance.

Basic Byte, Inc.
P.O. Box 924
Southfield, MI. 48037-0924
(313) 540-0655



Details Sketchy On Graphic Add-On

Drawing Tablet Hooks Up To All Popular Machines

Personal Peripherals, Inc. has announced the release of Super Sketch—a graphics tablet with software cartridge—and Super Sketch II, as well as four applications software packages for Super Sketch. Super Sketch allows users to create color graphics by moving a stylus control as they would a pencil. It is compatible with the TI-99/4A and the Commodore 64 and is priced at \$59.95. Super Sketch II is physically similar to Super Sketch, but it has different styling and "a color scheme designed to blend with professional environments." It is compatible with Apple II and IBM PC and PCjr computers. Its suggested retail price is \$79.95. The applications packages, for Commodore 64 computers, include Super Music Box



Personal Peripherals, Inc.
930 N. Beltline, Suite 120
Irving, TX. 75061
(214) 790-1440

(\$19.95) for composing and performing music, Business Presentor (\$39.95) for business-related graphics, Master Home Planner (\$49.95) for creating home and commercial floor plans, and Printer Utility (\$29.95) which allows print-out capability for video graphics created with Super Sketch.



Junior's Desk Gets Windows

Integrated Productivity & Adventure Packs for PCjr

The jr. Series (tm), a line of software products for the PCjr, has been released by Oakwood Publishing. The series currently includes jr. DESK, an integrated package with window displays, for personal finances and small business operations; jr. FILE, a home file management system; jr. QUEST, a fantasy

adventure game that teaches players how to use logic to enhance creative problem-solving skills; and jr. WORLD TRIATHLON, an adventure tutorial with an Olympics theme that tests players' skills in typing, spelling, and memory recall. Suggested retail price for jr. DESK is \$99.95, and for the other programs, \$49.95.

Oakwood Publishing
P.O. Box 3934
Gardena, CA 90247
(213) 217-1323



Learning With Robots

Construction-Kit Game Teaches Digital Electronics

A robot construction kit that teaches the basics of digital electronics, and develops logic and hypothesis formulation skills is the basis of Robot Odyssey I by The Learning Company. Players begin by falling into a futuristic underground city inhabited by robots. The object is to escape by

designing the circuitry and chips for robots which will help them get through various levels of civilization. The program, aimed at teenagers and young adults, comes with tutorials to assist players. It will be available for \$49.95 for the Apple II family of computers.

The Learning Company
545 Middlefield Rd., Suite 170
Menlo Park, CA. 94025
(415) 328-5410



That's Entertainment!

Imagic Launches New Software Series

Imagic has launched four new entertainment product lines—Fun with Experts, Educational Simulations, Living Literature, and Time Travelers. The Fun with Experts series kicks off with Crime and Punishment. Players assume the role of judge in sentencing offenders for crimes. The Educational Simulations series that began with Microsurgeon continues with Injured Engine, where the player is provided with the technical information

and tools required to tune a car engine. The first Living Literature product brings the recently published Damiano trilogy into the realm of an interactive graphics adventure game. The Time Travelers series opens with Another Bow, a Sherlock Holmes mystery set in post-Victorian England, and The Time Machine, based on H.G. Wells' science fiction story. All five of these programs will debut this fall at \$34.95.

Imagic
981 University Ave.
Los Gatos, CA. 95030
(408) 399-2200



More Storage for Junior

PCjr Gets Dual-Disk Controller

Legacy Technologies, Ltd. has announced a two drive controller for the IBM PCjr. Legacy's new floppy disk controller provides access to two disks through Junior's operating system.

Legacy Technologies, Ltd.
4817 North 56th St.
Lincoln, NE. 68504
1-800-228-7257

The cabling provided can power one disk drive inside junior, and extend under the cover to control a second drive. The second drive can be house inside a Legacy II or positioned alongside.



For Those With All the Answers

Trivia Games for Home Computers

Trivia Mania, a new game by Professional Software, Inc., brings the current trivia craze to Commodore 64, Apple II family, and IBM PC and PCjr users. The game consists of approximately 3,500 questions on diskette and in printed form, in three levels of difficulty and in seven categories: Science and Technology, Geography, History, Sports, Films and Entertainment, Famous People, and Nature and Animals. Trivia Mania retails for \$39.95. Professional Software is developing a series of add-on

Professional Software, Inc.
51 Fremont St.
Needham, MA. 02194
(617) 444-5224



diskette packages for the game entitled Super Sports (tm), Movie Madness (tm), What's in a Word (tm), and Educational Learning Diskettes (tm).

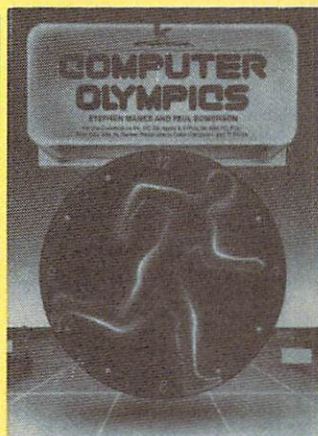


Booking An Olympic Event

Game Activity Book Teaches The BASICS

Released just in time for the Olympic Games, a new book by Stephen Manes and Paul Somerson provides ready-to-key-in listings of Olympic games and sideline activities. Computer Olympics, published by Scholastic, Inc., can be used with TI-99/4A, IBM PC and PCjr, Commodore 64 and VIC-20, and Apple II family computers. Programs include Toss the Javelin, Track and Field Record Book, and Bronze Medal Diver. The paperback book retails for \$4.95.

Scholastic Inc.
730 Broadway
New York, NY. 10003
(212) 505-3546



Cartoons and Kid Shows Go Floppy

Two Big-Name Titles from First Star

First Star Software's licensing agreements will bring MAD Magazine's cartoon strip Spy vs. Spy and the television classroom, Romper Room to the home computer this October. Spy vs. Spy will employ animated graphics, and, like the Romper Room programs, will be released initially for the Commodore

First Star Software
18 East 41st St.
New York, NY. 10017
(212) 532-4666

64 and Apple II family computers. The first program of the ROMPER ROOM Little Learner (tm) Series will be Romper Room's I Love My Alphabet. It will feature animations demonstrating different action words. The games will retail for \$34.95 for the C-64 versions, and \$39.95 for the Apple versions.



Tapes do the Teaching

Hands-On Learning Exercises for PCjr

FlipTrack Learning Systems has published a tutorial on How to Operate the PCjr. The self-paced tutorial is designed to teach hands-on computer operation through two audio cassettes. The first cassette guides users through start-up procedures; keyboard familiarization; BASIC programming; and the PCjr's color, sound, graphics,

FlipTrack Learning Systems
999 Main, Suite 200
Glen Ellyn, IL. 60137
(312) 790-1117

and mathematical capabilities. The second cassette details managing disk storage and files with DOS; using tree-structured directories; copying, renaming, and erasing files; and batch processing. An indexed quick reference guide accompanies the cassettes. How to Operate the PCjr is priced at \$39.95.



A Bigger Image for Apples

Wide-Carriage Printer Makes Debut

A wide carriage model of the Imagewriter dot matrix printer is now available from Apple Computer. The Wide Carriage Imagewriter is suited for producing documents that require wide paper such as spreadsheets, forecasting models, budgets, and data processing reports. It accommodates a range of paper sizes from three to 15 inches wide, and is compatible with Apple II and Apple III computers. Like the standard size model, the Wide Carriage Image-



writer prints in a 7 x 9 dot matrix at a rate of up to 120 characters per second. It also features eight character fonts, and provides variable resolution, pitch, and line spacing. This new Apple printer retails for \$749.

Apple Computer, Inc.
20525 Mariani Ave.
Cupertino, CA. 95014
(408) 996-1010



Make It Easy On Yourself

Word Processing & Utilities for the C-64

Educomp has announced three new low-priced programs for the Commodore 64—a word processor and two utilities. The Quickwriter II has over 60 editing commands, and a printer routine compatible with every printer interface and printer combination.

Power Plus adds over 40 new commands to the C-64, making it easier to send disk commands, write and debug

Educomp
2139 Newcastle Ave.
Cardiff, CA. 92007
(619) 942-3838

programs, and write code in machine and assembly languages. Menu-driven Disk Pac can check all sectors of a diskette for problems without losing the diskette's data. It can also unscratch data files that have been accidentally scratched or erased. Both Quickwriter II and Power Plus are priced at \$19.95, and the Disk Pac is priced at \$14.95.



PC Software in the Public Domain

New Directory Shows Where and How To Find It

A new directory from PC Software Interest Group lists hundreds of public domain and user-supported programs available for the IBM PC and compatible computers. The Directory of Public Domain Software for the IBM Personal Computer is composed of programs written by people who have chosen not to market their

PC Software Interest Group
1556 Halford Ave. Suite 1305
Santa Clara, CA. 95051
(408) 730-9291

software. The directory catalogs what programs are available and where to get them. It lists word processing, communications, data base, DOS and BASIC utilities, games with color graphics, Pascal, C and assembly language programs, and more. The directory retails for \$4.95.



Balancing a Checkbook Made Easy

Rocketman to the Rescue

A new program for the TI-99/4A has been released by Rocketman to assist people who hate to balance checkbooks. Rocketman Jr. (cassette version) and Rocketman Sr. (diskette version) systematically input all the information needed for reconciliation. The program provides a single screen read-out, and compiles, displays, and com-

Rocketman
4104 San Pablo Dam Rd.
El Sobrante, CA. 94893
(415) 222-1626

pares all data. Entries can be corrected until DIFFERENCE=0, indicating balance. It includes graphics and a built-in calculator, which checks the additions and subtractions of all entries in the checkbook register. Rocketman Jr. retails for \$24.95 and Rocketman Sr. retails for \$39.95.



HOME COMPUTERTM

product news

FLASH! LATEST IBM NEWS!

IBM Adds Enhancements to the PCjr

Five new features that extend the power of the PCjr have been added to the machine by IBM. Available in August, the enhancements include a typewriter-style keyboard, an optional 128KB memory expansion attachment, a program to allow all or part of the expanded memory to be transformed into an "electronic diskette" and used as if it were a second disk drive, a speech synthesizer, and an expansion attachment to provide extra power.

The new standard keyboard has 62 individually-contoured, programmable keys, arranged in typewriter layout. It is battery-powered and operates through an infrared optical link. An optional connecting cord is available. Current PCjr owners and those who purchase a PCjr from existing supplies can obtain the new keyboard for no charge.

The IBM PCjr 128KB Memory Expansion Attachment adds 131,072 characters of user memory to a PCjr, and can be used to run thousands of IBM PC programs. It has sixteen 64K X 1 DRAMS, and comes with the Memory Options diskette, which allows DOS to use the expanded mem-

IBM Entry Systems Div.
P.O. Box 1328
Boca Raton, FL 33432
(305) 241-7632



ory. Up to three attachments can be connected to the computer's side expansion port to boost Junior's total memory to 512KB of RAM. Each attachment retails for \$325.

The Power Expansion Attachment provides 20 watts of additional power to support multiple 128KB Memory Expansion, Speech, and Parallel Printer side attachments in any combination, to a maximum of three. It retails for \$150.

The IBM PCjr Speech Attachment is a speech synthesizer that supports speech encoding in compressed mode, and contains 196 words in its ROM. A 3.5mm microphone input jack is provided for recording speech on a diskette. It retails for \$300.

Big Package for Expanded Model

Lotus Announces 1-2-3 Cartridge for the PCjr

In November, Lotus Development Corp. will release a new version of their 1-2-3 integrated software package for the IBM PCjr on a ROM cartridge. It will

have a suggested retail price of \$495. Lotus 1-2-3 combines spreadsheet analysis, database management, and business graphics in one program.

Lotus Development Corp.
161 First Street
Cambridge, MA 02142

Big Blue Unveils Computer Assistant

New IBM Productivity Packs Replace PFS Line-Up

IBM has introduced a new, modular family of software products called the IBM Personal Computer Assistant Series, which can be used with the full line of IBM Personal Computers.

The series includes the IBM Personal Computer Writing Assistant, a word processing program for \$149 that includes the IBM Personal Computer Word Proof spelling verification aid; IBM Personal Computer Filing Assistant for \$149, an enhanced version of the IBM Personal Computer PFS: FILE program that enables users to design filing systems, add or delete items, and quickly search and update the records; IBM Personal Computer Reporting Assistant for \$129, an enhanced version of the IBM Personal Computer PFS:REPORT program that

IBM Entry Systems Div.
P.O. Box 2989
Delray Beach, FL 33444
(305) 241-7614

sorts and organizes files generated with IBM Filing Assistant and displays or prints them in tabular form; and the IBM Personal Computer Graphing Assistant for \$149, which produces up to four line, bar, or pie graphs as a single chart using information from IBM Filing Assistant or IBM Reporting Assistant.

The IBM Personal Computer Planning Assistant is a spreadsheet program that helps professionals with budgeting, planning, forecasting and financial analysis. It will not be available until the first quarter of 1985 and will sell for \$149.

Individuals may upgrade their existing IBM PFS:FILE to IBM Filing Assistant and IBM PFS:REPORT to IBM Reporting Assistant for \$45 each until Nov. 15, 1984.

Junior Goes to School

Educational, Graphics Programs Released by IBM

Several software products were released by IBM in August for the PC and Pcj, including the Earth Science Series, a set of 4 programs that teaches students about the water processes in the physical environment. The series includes The Hydrologic Cycle, Ground Water, Surface Water, and Moisture in the Atmosphere. These programs retail for \$49 each.

PCjr Color Paint is a graphic-design cartridge program that permits users to create full-color artwork only on the PCjr. It is \$99 retail.

Rocky's Boots, an educational game program, teaches the basics of computer logic and electronic circuitry. It is available for \$49.95.

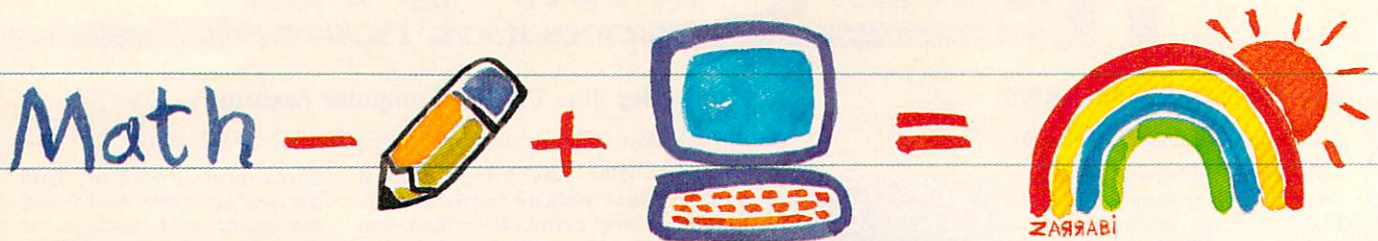
IBM Entry Systems Div.
P.O. Box 1328
Boca Raton, FL 33432
(305) 241-7632

BASIC Primer 2.0 includes lessons specifically written for the PCjr. It is \$60 retail.

IBM Private Tutor 2.0, with enhanced graphics and video disk compatibility, is a self-study system available for \$50.

Teacher's Quiz Designer helps instructors create and administer quizzes. Retail price is \$70.

The "Writing to Read" system will be available to all schools this September. Using the IBM PCjr, workbooks, and cassette tapes, children hear sounds and see pictures that they then learn to read and write. A "Writing to Read" center that can accommodate 120 students per day costs approximately \$10,456 to set up. Purchasing the materials for just one pupil would cost about \$2,298.



by Mark Dewese
and the HCM Staff

ELEMENTARY ADDITION AND SUBTRACTION

The previous two issues of *Home Computer Magazine* contained versions of this educational program for pre-schoolers for VIC-20, C-64, and TI-99/4A computers. Now, for readers with IBM PC, PCjr, or Apple II family computers, we present two additional versions. These programs take advantage of the outstanding color and sound capabilities of these machines to make learning the basics of addition and subtraction an entertaining experience for a small child.

Varying Levels of Difficulty

Add Subtract Program offers simple problems with answers ranging from zero to nine, on three levels of difficulty. You help your child get

started by selecting either addition or subtraction from the first menu. The next menu lets you choose one of the three levels. The easiest level is aimed at children just learning to count. Each problem is accompanied by a graphic representation of the problem—including a graphic answer. The next level includes graphics of the problem, but the answer's graphics do not appear until the child indicates the answer. The hardest level displays only numeric problems.

To enter an answer, the child presses any one of the number keys.

The computer immediately evaluates the child's answer and provides feedback to the child. As with most good educational software, the child is rewarded for selecting the right answer; here the reward is a little tune and colorful graphics. If the answer is not correct, the computer erases the child's incorrect answer and gives the child another chance. Following a correct answer, the screen is erased and a new problem appears. At any point you may change modes by pressing M and returning to the first menu.



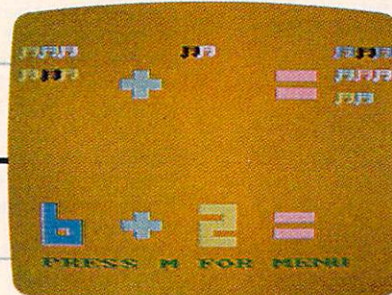
The subroutine in lines 420 and 430 selects problems for both addition or subtraction, and assigns the values depending upon which option was chosen. A random number between 0 and 9 is selected in line 420:

```
420 L = INT (RND (1) * 9) + 1
```

Line 430 then selects a smaller number between 0 and the first number, and determines the final number by subtracting the first from the second.

```
430 S2 = INT (RND (1) * L) + 1:  
S2 = L - S1: RETURN
```

Line 220 calls this routine and assigns the numbers to the left center or right position on the screen depending on the value of OV—which will be 1 if addition is selected, or 2 if subtraction is the selection. This is a good demonstration of how



sends sound through only the television/external speaker; and finally, SOUND OFF:BEEP OFF sends sound to the console alone. The console supports only a single voice; the external speaker channel is multi-voice.

This program uses the default mode (SOUND OFF:BEEP ON) to maintain compatibility between the IBM PC and PCjr. You won't find the BEEP ON :SOUND OFF commands in the program, however, because BASIC on the PC doesn't support them.

to use a minimum of code to achieve two seemingly different tasks.

ADD SUBTRACT PROGRAM (Apple II Family)

Explanation of the Program

Line Nos.	
100-160	Program header.
170-240	Initialization.
250-430	Main program loop.
440-490	Make up problem routine.
500-600	Title and menu screen routines.
610-990	Subroutines to make music, draw objects, etc.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

HCM



The tune at the start of the PC and PCjr versions of *Add Subtract Program* is the first four measures of Bach's "Two-Part Inventions, No. 1." The music is contained in lines 220-230.

Sound on the IBM PC always comes out of the internal speaker in the console, but on the PCjr it can be directed through either the internal speaker or through an external speaker. Sound is enabled in three modes on the PCjr: the default mode, SOUND OFF:BEEP ON, sends sound through both the console and the external speaker; SOUND ON:BEEP OFF

Since this mode doesn't allow multiple voices, the tune is Bach's right hand alone, and plays through the console speaker. You pianists with a PCjr and external speakers might like to try putting the other voice in, and enable the SOUND ON:BEEP OFF.

ADD SUBTRACT PROGRAM (IBM PC and PCjr)

Explanation of the Program

Line Nos.	
100-170	Program header.
180-350	Initialization, menu screens.
360-380	Select numbers.
390-570	Main program loop.
580-600	Get input subroutine.
610-1160	Subroutines to draw numbers and graphics.

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

HCM



This article continues an ongoing tutorial on the *Multiplan* software package. Newcomers to the program may wish to consult back issues of *99'er Home Computer Magazine* for previous articles in this series.

Many people feel they must justify the cost of adding a new peripheral to their home computer system. They think it is not enough to simply want to improve their machine's efficiency or increase their own computer literacy. If you are one of those who is seeking "cost-effectiveness rationalization," you will be interested in finding out how *Multiplan* can help you calculate the cost effectiveness of adding new equipment to your system. As an example, we will determine whether the cost of adding a printer to your system is justified in terms of the time and money it saves. The ideas presented here can be used for any major purchase decision.

There are two types of costs associated with equipment: initial and recurring. The purchase price of the printer is an initial (one-time) cost, while the costs of maintenance and supplies will recur over the years. In addition, recurring costs generally increase each year at some rate, such as the inflation rate.

Even though the equipment will last for several years, we must make our buying decision today. So, we need to express the projected costs and savings associated with the equipment in today's dollars. Thus, the decision to buy should also take into consideration the so-called "opportunity rate." This rate is also called the cost of money, the discount rate, or the interest rate.

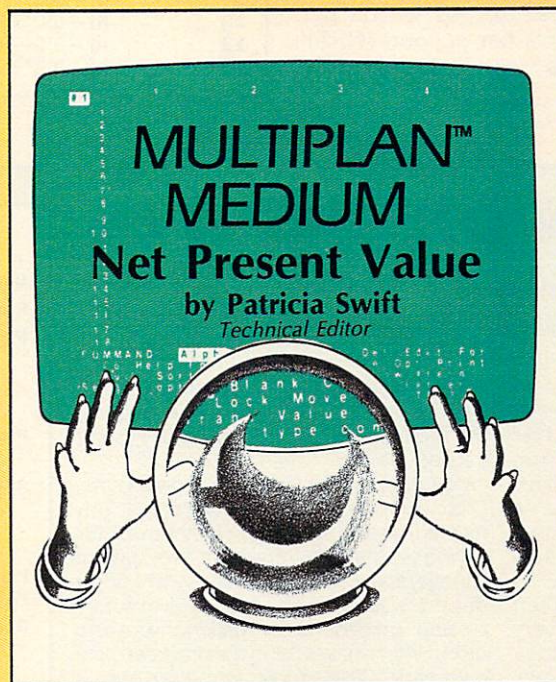
This is where Net Present Value (NPV) comes in. In mathematical terms, the total cost of something which will be purchased over n years with a discount rate of k in effect will be (in today's dollars):

$$\sum_{i=1}^n \frac{\text{cost during year } i}{(1+k)^i}$$

Multiplan's NPV function performs this calculation. We supply it with a list of costs for each year and the discount rate, and *Multiplan* returns the value in today's dollars. Thus, we don't have to be accounting wizards to make good buying decisions. Figure 4 shows the Net Present Value model to which the following discussion refers.

Factors in Equipment Cost

The initial costs for adding a printer are the cost of the equipment and the cost of the software to use the printer. Our hypothetical user already has *Multiplan*, which probably means that he or she already has whatever else the system requires to run *Multiplan* (e.g., memory expansion). Therefore, the only hardware needed is probably the printer, a cable, and an RS232



Interface Adapter. *Multiplan* can output to the printer, so no new software must be added. Since these one-time costs all occur at the beginning of the first year of use, they are already in today's dollars. So no fancy calculations based on inflation rates are needed at this point.

Two recurring costs of owning a printer are maintenance and supplies. I have set the first year's maintenance at 10% of the cost of the printer itself because my experience with printers shows that this is a reasonable figure. Some users may purchase a maintenance agreement with their printers, which would cause this figure to vary. The cost of supplies depends on exactly how the printer will be used. You should include paper and ribbons in this figure. If you are

analyzing the purchase of a daisy wheel printer, be sure to include the cost of the wheels at this point, since they wear out over time and may need to be replaced.

This model assumes that the printer's life is three years. The recurring costs in the second and third years of life have had a projected inflation rate applied to them. The model assumes a fixed inflation rate, but you can easily get fancier if you like. The figure for each year is based on the year before, so you can use two different inflation rates to arrive at year 2 and year 3.

Once the costs have been filled in, you'll want to calculate the NPV of each recurring item. I have based this calculation on the interest rate given at the top of the model. Because this rate can vary, I have set up the model so this can be changed easily. The NPV of each recurring cost is figured on this rate and on the three years of cost that are in the same row. Thus, the figures in the NPV column of the model represent costs in today's dollars.

Some users may want to include more costs than those shown. For example, if you are adding memory expansion as well as a printer, then you should probably come up with a figure for electricity usage. This would be a recurring cost, and the amounts projected for the second and third years might be based on a rate different from the inflation rate. If you are adding a large printer, you might also want to buy a word processor to make maximum use of your new printer.

Time and Accuracy Savings

Now for the other side of the analysis, the savings you expect to gain from adding a printer. If you will be using the printer to produce something which you are already producing by hand or on a typewriter, then these savings are fairly easy to measure. I have used two types of savings: time and increased accuracy.

Because the value of time saved is a fairly complicated calculation, I have used a supporting worksheet to calculate it. Figure 3 shows this supporting worksheet. If you currently type the materials that will be printed in the future, then you can use this supporting sheet

to put a value on your time saved by using a printer. You must first estimate the volume of typing you usually do. In the example, this is expressed in weeks: 4 documents of 300 words per week. The average word used is 7 characters long. You must also specify your typing speed and the speed of the printer you are considering; in the example these are 40 words per minute (WPM) and 150 characters per second (CPS), respectively. To figure the hours per week currently spent typing, use the formula:

$$\frac{\text{words/document} \times \text{documents}}{\text{speed (wpm)} \div 60 \text{ min./hr.}}$$

Figure 1—Formulae for Supporting Worksheet

1	2	3
11	R[-4]C*R[-3]C/R[-7]C/60	
12	R[-5]C*R[-4]C/R[-7]C*R[-6]C/60/60	
13	R[-2]C - R[-1]C	
15	R[-2]C*52*R[-6]C	

MULTIPLAN ON THE C-64

A review
by Patricia Swift



When asked to try out *Multiplan* on the Commodore 64, I agreed eagerly even though I had never used that computer before. I have run *Multiplan* on several other microcomputers and have always found the program to be about the same as far as the human interface goes. *Multiplan* on the C-64 is no exception.

You need at least one disk drive to run this program on the C-64. Not being very familiar with this computer, I had more trouble correctly hooking up the console, monitor, and disk drive than I did using *Multiplan*. In other words, HesWare has done a good job of implementing *Multiplan* on the C-64. The manual is first-rate. It's the standard Microsoft *Multiplan* manual with tutorial and reference sections, but edited for the C-64 with information about special keys and disk handling. The "Getting Started" and "Operating Instructions" sections are extremely helpful and should definitely be read before you try to use the program.

Slow Disk Access

The start-up sequence for *Multiplan* is clumsy, although seasoned C-64 users would probably think nothing of it. You have to type LOAD "MP", 8 and then RUN to get going. After that, you have to wait for over two minutes until you can use the program. This long load time might scare new users (it worried me quite a bit), but you just have to be very patient. This slow disk performance is a characteristic of the C-64 and should not be blamed on *Multiplan*—the disk drive is connected to the console via a serial interface. Serial interfaces transmit data one bit at a time, which explains why a large program like *Multiplan* takes a long time to load into memory. Most other computers I have tried (including the TI-99/4A) use a parallel interface for the disk; parallel interfaces transmit a byte (8 bits) at a time.

Tortoise Speed

Once the program is loaded, it runs very quickly, except when the disk is being accessed. Unfortunately, that seems to happen more often than with other versions of *Multiplan*. Many of the commands require a disk read. You'd expect the HELP and TRANSFER commands to access the disk, but BLANK, FORMAT,

WINDOW, and others do it too. Although each disk access takes only a few seconds, the cumulative effect can really slow you down.

Multiplan for the C-64 comes on a write-protected disk that cannot be copied. This means that you must use a separate disk for saving worksheets—making it necessary to change disks if you are running with a single disk drive (as I was). The manual gives a step-by-step procedure for making working disks, which makes using the program quite a bit easier. This procedure copies onto a separate disk the two files that *Multiplan* needs at runtime, and even formats the working disk for you. You still have to load *Multiplan* from the original disk, but then you can switch to a working disk and leave it inserted while you work. A word of warning: This procedure was explained on a separate page that was stuck into the disk holder at the back of the manual. It's easy to miss, but definitely worth looking for.

HesWare will sell you a backup copy of the *Multiplan* disk for \$10. This sounds like a good way to protect your software investment, especially since floppy disks get quite warm (even hot) after being in the Commodore's disk drive for a while.

While running the program, I had the most trouble with the [SHIFT] key. Seasoned Commodore users already know that the [SHIFT LOCK] key gives you uppercase everything, not just capital letters. Even the [RETURN] key works differently with the [SHIFT LOCK] down. This setup really had me confused for a while—I recommend that you leave the [SHIFT LOCK] up (disengaged) when using *Multiplan*.

The display shows 40 columns with a solid border around the whole worksheet. This resulted in fairly small characters on my 10" monitor, which made it hard to read the screen. However, the border is a good idea because it prevents a badly-aligned screen from chopping off characters. Thus, if you use a TV set, you won't suffer with characters that are fuzzy around the edges.

The Key Advantage

The special keys on the C-64 are used to make *Multiplan* easier to operate. For example, the [RUN/STOP] key means cancel (it's sort of a panic button). It is extremely handy to have this often-used command as a single key, although you can still use the traditional [CONTROL] C for cancel.

Name: Microsoft/Multiplan
Program Type: Electronic Worksheet
Distributor: HesWare
150 North Hill Drive
Brisbane, CA 94005
(415) 468-4111
Price: \$99.95

System Requirements: Disk Drive

	Poor	Fair	Good	Excellent
Performance:	████████████████████			
Ease of Use:	████████████████████			
Documentation:	████████████████████			

My other favorites are [F1] for tab and [F3] for delete. A small overlay is provided for the function keys so you don't have to memorize their meanings. There are still a few key sequences which must be learned or looked-up (for example, scroll down is [CONTROL] R (DOWN ARROW)).

When printing with *Multiplan* on the C-64, you can skip the usual preliminary steps of opening channels between peripherals to access your printer. Once you boot up the system and access a file, you can get printouts as easy as typing P. The P command automatically puts you in print mode, and displays the following:

PRINT: Printer File Margins Options

These are the four printer subcommands available with *Multiplan*. The first subcommand, Printer, allows you to print an entire worksheet, under the limits of preset margins. The second subcommand, File, allows you to store printed output on disk rather than send it directly to the printer. This provides the option of adding to or changing data before getting a printout. The third subcommand, Margins, lets you set margins, specify the number of characters per line, and set page lengths. The last subcommand, Options, lets you print specific areas within your worksheet.

The major differences in running *Multiplan* on the Commodore 64—as you can see—all relate to the machine itself. All of *Multiplan*'s commands and functions are there, and the syntax is the same as on other machines. This means that Commodore 64 users can use the *Multiplan* models described in my *Multiplan* Medium series, and elsewhere. Just take some time to become familiar with the Commodore's special keys, and don't forget to make yourself a working disk.

HCM

Figure 2

Building the Model

1. First build the supporting worksheet. Set the Format Width for column 1 to a width of 25 characters. Then fill in the labels in column 1 (see Figure 3).
2. Fill in the speeds and their accompanying labels. Note that the format of the cell containing the hourly rate (R9C2) should be changed to \$ via the **FORMAT CELLS** command.
3. Now fill in the calculations (R11C2:R15C2) and their accompanying labels in column 3.
4. Use the **NAME** command to assign the name **TIMEVAL** to the result in R15C2.
5. Store the supporting sheet on disk via the **TRANSFER RENAME** command. Save it on a file named **TIMEWK**.
6. Now get ready to construct the main model. Use the **TRANSFER CLEAR** command to clear the screen.
7. Set the Format Width for column 1 to a width of 26 characters. Then fill in the labels in column 1 (see Figure 4).
8. Change the default format for the worksheet to Fixed with 2 decimal

places via the **FORMAT DEFAULT CELLS** command.

9. Change the format of the two cells for rates (R4C2:R5C2) to percentage (%) via the **FORMAT CELLS** command. Then fill in the projected inflation rate and interest rate in those cells. Name those cells **INFL** and **INTEREST**, respectively.

10. Fill in the headings in row 7.

11. Fill in the costs of the printer, cable, and RS232 card in column 2. **SUM** these into R12C2. Then set the NPV in R12C5 to this same value by using = and picking up the total just calculated.

12. Fill in the cost of software at R14C2 and put it into R14C5 in the same way.

13. Put the formula for the first year's maintenance in R16C2. If you'll have a maintenance contract, you may want to use an actual figure here instead.

14. Fill in the formula for R16C3, then Copy it 1 cell to the right. Fill in the NPV formula at R16C5.

15. Fill in the first year's cost of supplies at R18C2. Then copy the formula

for years 2 and 3 from R16C3 to R18C3:R18C4. Copy the NPV formula from R16C5 to R18C5.

16. Use the external **COPY** command to put the time value from the supporting worksheet into R22C2. Copy from sheet **TIMEWK**; name it **TIMEVAL**. Be sure to set the Link option to Yes so that if the supporting sheet is changed, then the main sheet will also be changed.

17. Fill in the value of increased accuracy at R23C2. Then **SUM** the two savings values into R24C2. Compute the savings for the second and third years by copying the formulas from cell R16C3 to cells R24C3:R24C4. Copy the NPV formula from R16C5 to R24C5.

18. Compute the overall NPV of adding the printer by entering the formula shown into R26C5.

19. Store the main worksheet on disk via the **TRANSFER RENAME** command. Save it on a file named NPV, or any other name of your choice.

The calculation for the hours which the printer would take to accomplish the same task is more complicated only because printer speeds are usually expressed in characters per second:

$$\frac{\text{words/doc.} \times \text{documents} \times \text{avg. word length}}{\text{speed (cps)} \div 3600 \text{ sec/hr.}}$$

The number of hours saved per week is just the difference between these two figures. (This printer example assumes the documents are already stored in a word processor.) Finally, you must assign a value to your time. Then the value of the time saved per year by using a printer will be:

$$\text{time saved/wk.} \times 52 \text{ wks./yr.} \times \text{your hourly value}$$

This supporting worksheet can supply the bottom-line time value to the main worksheet automatically. To do this, you must construct the supporting sheet

Figure 3—Value of Time Saved

1	2	3
1 Value of Time Saved		
2 Supporting Worksheet		
3		
4 Typing Speed	40	WPM
5 Printer Speed	150	CPS
6 Average Word Length	7	CHARS
7 Average Document Length	300	WORDS
8 Avg. Documents Per Week	4	
9 Value of Your Time	\$10.00	PER HR
10		
11 Current Time Spent Typing	0.5	HRS/WK
12 Printer Time	0.0155556	HRS/WK
13 Time Saved	0.4844444	HRS/WK
14		
15 Value of Time Saved	\$251.91	Per Year

first, name the cell containing the final result, and save the supporting sheet on disk. Then when you construct the main (NPV) model, you can ask *Multiplan* to use this "external" value by specifying the filename of the worksheet and the name of the cell. By specifying at this time that the worksheets are to be linked, you can cause changes in the supporting sheet to be automatically reflected on the main worksheet.

Let's return to our discussion of the main model. The other savings shown is in increased accuracy. This is more difficult to measure, as it can arise from many factors. For example, if your results are used for billings, then mistakes can cost you money. You may already have an idea of how much mistakes like this have cost you in the past.

You may be able to come up with other savings for your particular situation. Just remember that these will generally be recurring savings, and that you can use supporting sheets to work out the amounts.

In the model, the total savings are extended to the second and third years in the same ways as the costs, and the NPV is figured on the three yearly totals.

To decide whether buying a printer would be cost effective, you must see whether the NPV of the savings is more than the NPV of the costs. The final cell of the main model is calculated as savings minus all the costs. The sample model comes out to a positive figure here, meaning that this particular printer is cost effective for this situation.

To decide whether buying a printer would be cost effective, you must see whether the NPV of the savings is more than the NPV of the costs. The final cell of the main model is calculated as savings minus all the costs. The sample model comes out to a positive figure here, meaning that this particular printer is cost effective for this situation.

Building and Using the Model

The steps for building this model are shown in Figure 2. Once it has been built and saved, you'll want to use it to evaluate several different situations by varying some of the rates and costs. As you do this, you'll

Figure 4—Net Present Value Model

	1	2	3	4	5
1	Net Present Value Model				
2	For Adding a Printer				
3					
4	Projected Inflation Rate	6.00%			
5	Interest Rate	8.50%			
6					
7		Year 1	Year 2	Year 3	NPV
8	Cost of Printer:				
9	Printer	400.00			
10	Cable	35.00			
11	RS232 Interface Card	100.00			
12	Initial Printer Cost	535.00			535.00
13					
14	Cost of Software	0.00			0.00
15					
16	Maintenance	40.00	42.40	44.94	108.07
17					
18	Supplies	50.00	53.00	56.18	135.09
19					
20					
21	Savings:				
22	Value of Time Saved	251.91			
23	Increased Accuracy	150.00			
24	Total Savings	401.91	426.03	451.59	1085.87
25					
26	Overall NPV to Add Printer				307.71

probably want to see the final result each time. This result appears in row 26 in the sample model, but you can use *Multiplan*'s windowing facilities to keep the result visible at all times.

To do this, put the cell pointer on the rightmost cell on the result line. Then move the cell pointer up one cell. Type in **WINDOW** and press **[ENTER]** or **[RETURN]** three more times. This will give you a second window, just one row high, at the bottom of the screen. Notice that you should not choose the **Linked** option, since you don't want the new window to move around horizontally when the larger window does. Now move your cell pointer down one row to show the result. After this, go to the main window and

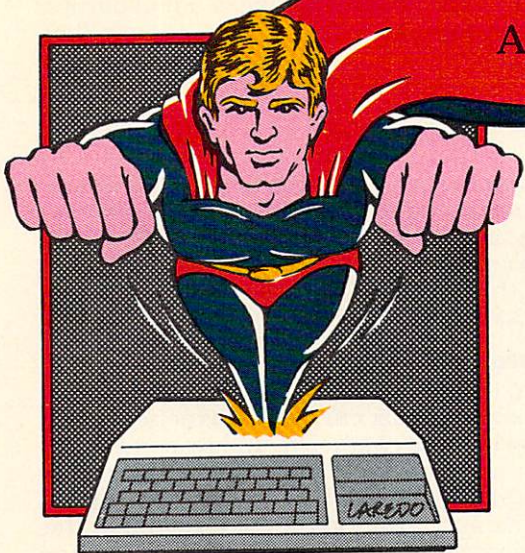
experiment with the model. For example, change the interest rate and watch the effect on the overall NPV.

The techniques presented here can be used to analyze the costs of buying all sorts of equipment. Even when the purchase involves no savings, it is still useful to find out how much the projected purchase will cost you in today's dollars. The model illustrates that the initial purchase price is not the only cost you should consider; maintenance and other recurring costs should also be included in a thoughtful analysis. And *Multiplan* provides the structure for translating all of this information into an accurate prediction of cost effectiveness.

HCM

Figure 5—Formulae For NPV Model

	1	2	3	4	5
1					
2					
3					
4					
5					
6					
7					
8					
9					
10					
11					
12		SUM(R[-3]C:R[-1]C)			RC[-3]
13					RC[-3]
14					
15					
16		R[-7]C/10	RC[-1]*(1+INFL)	RC[-1]*(1+INFL)	NPV(INTEREST,RC[-3]:RC[-1])
17					
18			RC[-1]*(1+INFL)	RC[-1]*(1+INFL)	NPV(INTEREST,RC[-3]:RC[-1])
19					
20					
21					
22		[TIMEWK TIMEVAL]			
23					
24		SUM(R[-2]C:R[-1]C)	RC[-1]*(1+INFL)	RC[-1]*(1+INFL)	NPV(INTEREST,RC[-3]:RC[-1])
25					
26					R[-2]C-R[-14] C-R[-12]C-R[-10]C- R[-8]C



SUPER LANGUAGE

A Home Computer Assembly Language Series

Have No Fear:

Assembly Language Won't Byte:

Part IV

by **Peter Lottrup**
and the HCM Staff

Programmers use Assembly Language to create video games and other extremely fast routines. Key into a deeper level of understanding with this super tutorial.

In this fourth and final segment of our tutorial on Assembly Language for the TI-99/4A, we will look at the two remaining Assembly Language directives available with the Mini Memory: *EQU* and *BSS*. Then we will put together everything you've learned so far and write a program.

EQU and BSS Directives

The last two directives are *EQU* and *BSS*. You can think of *EQU* as an *EQU*al sign in BASIC — equating a label with any quantity two bytes in length. For example, if you wish to use a label for Video Multiple-Byte Write (*VMBW*), you enter: `VW EQU >6028`

This directive takes up no actual program space—it merely adds another symbol to the *SYMBOL* table. See Listing 1 for a short program that demonstrates how this is used.

Because we used the *EQU* directive to equate *VW* with the address of the *VMBW* utility, we could use the *VW* in line `>7D0C` (of Listing 1) in place of the address.

A new instruction is introduced in Listing 1: *LIMI* (Load Interrupt Mask Immediate). *LIMI 2* and *LIMI 0* are used in a loop at the end of the program (lines `>7D10` through `>7D18`) so that the program can be halted using (*FCIN*) (*QUIT*). This gives you a convenient way to end your program without shutting off the machine.

The *BSS* directive is used when you wish to reserve a section of memory for text, numbers, or variables. With the *BSS* directive you can set aside an area called a *buffer*. Use *AORG* to go to the desired location for the buffer area, assign the start with a label (if you wish), then enter *BSS*, a space, and the number of bytes you want to set aside.

Programming Tips

As you write longer and longer programs using the Line-by-Line Assembler, you will tend to use more and more labels to keep track of addresses and buffer areas. Each label takes up four bytes in the *SYMBOL* table. This table starts at memory location `>7CD8`, and each label you

add pushes the end of this table toward `>7FFF`. If you start your program at `>7D00` and use more than seven labels, the *SYMBOL* table will write over the beginning of your program.

The second programming tip concerns short jumps in a program. Due to the limited space reserved for the symbol table, it is often better to use *\$* to specify jumps of just a few addresses. When used in an assembly language instruction, *\$* stands for the current location counter. If you wish to jump back eight addresses from the current instruction, just type: `JMP $-8`.

To figure any jump, just subtract the current location counter from the location you wish to jump to. Remember that the addresses are hexadecimal (HEX or base 16), but the numbers you enter in the Assembler are decimal (base 10) unless you specify otherwise. For example, if you wish to jump from address `>7D58` to address `>7D74`, subtract:

```
>7D74 (Address to jump to)
- >7D58 (Address of jump instruction)
-----
>1C (length of jump in HEX)
```

You would enter 28 (decimal) because 10 HEX = 16 decimal, C HEX = 12 decimal and 16 + 12 = 28.

Once Upon a KSCAN

One of the trickiest (but most powerful) utilities available to you with the Mini Memory is the key-scan (*KSCAN*) utility located at address `>6020`. Key-scan is your program's link to the keyboard. It allows you to do all the things a BASIC programmer does with *CALL KEY* and *INPUT* statements.

To use *KSCAN* you need to understand the functions of three memory locations:

- 1) `>8374` is like the *key-unit variable* in *CALL KEY*. A 0 here means *KSCAN* will scan the whole keyboard.
- 2) `>8375` is similar to the *return-variable* in *CALL KEY*. The ASCII code of the last key pressed is here.
- 3) `>837C` is used like the *status-variable* in *CALL KEY*.

Listing 1

```

7D00 xxxx      AORG >7D00
7D00 xxxx      EQU >6028
7D00 0200      LI 0,392
7D02 0188
7D04 0201      LI 1,ST
7D06 7D1A      LI 2,12
7D08 0202
7D0A 000C      BLWP @VW
7D0C 0420
7D0E 6028
7D10 0300      NN      LIM1 2
7D12 0002      LIM1 0
7D14 0300
7D16 0000
7D18 10FB      JMP NN
7D1A 4845      ST      TEXT 'HELLO THERE!'
7D1C 4C4C
7D1E 4F20
7D20 5448
7D22 4552
7D24 4521
7C26 xxxxx      END

```

To do a KSCAN of the entire keyboard, simply load a 0 into >8374; then branch and link to KSCAN. If all you had to do was check the character at address >8375, it would be easy. But first you must check to see if a key was pressed at all. To do this, you must check bit number 2 of location >837C. The Compare Ones Corresponding (COC) instruction is used with a register containing a mask constant. The bit we wish to test is the only place in the mask containing a 1. In our case we need to mask off all but bit 2 with zeros, so our mask constant would be >2000.

Here's how to set up a mask. In the TMS9900, the bits are numbered from 0 (far-left bit) to 15 (far-right bit) as illustrated below:

```

mask: >2000
0 1 0 0 0 0 0 0 | 0 0 0 0 0 0 0 0
0 1 2 3 4 5 6 7 | 8 9 10 11 12 13 14 15
bit position

```

When you use the COC instruction, all bits with zeros in the mask are ignored, but any bit containing a 1 (bit 2 above) is compared to the specified memory location or register. The status flags are set according to the

comparison of those bits with ones. If we place our mask in register 6, move the byte to be compared to register 1, and then do a COC instruction, we can find out if a key has been pressed. Here is a section of code that would do this:

```

LI 6,>2000
MOVB @>837C,1
COC 6,1

```

If the comparison shows that the two are not equal, the next instruction could jump (JNE) back to redo the KSCAN. If they are equal (i.e., both of the registers have ones in the second bit position), then the program could proceed to get the ASCII value of the key pressed from memory location >8375.

The Program

The program in Listing 2 displays a greeting on the screen and prompts the user to type in his or her name. It then accepts up to 12 characters and responds with a greeting that uses the name. The program is meant to be run using Mini Memory's Run option. We will refer to the actual memory locations of the program in the Mini Memory as we explain exactly what the program is doing.

Before we start the program itself, we enter the text we want displayed and assign a label to each section. First the initial greeting, 'HI! WHAT IS YOUR NAME?' is typed in using a TEXT directive at >7D00 and is given the label T1. To display the blanks that prompt for the user's name, we type ('.....') (twelve underline characters) at >7D16 and label this T2. Finally, we assign the label T3 to 'HELLO,' (our last message), starting at location >7D22. This brings us to the beginning of the program at >7D2A. Note this address; you will need to enter it in the REF/DEF table after keying in the program.

Because Mini Memory's Run option clears the screen automatically, we can begin by writing our first greeting. The screen is divided into 768 character locations (24 rows by 32 columns). We select location 100 (row 4, column 5) for our starting place. In lines >7D2A through >7D38, we load this location in register 0; the address of our text (T1) in register 1; and the length of the message (22 characters) in register 2. Then we branch and link to Video Multiple-Byte Write (VMBW).

Our input routine will give the user the option to Erase mistakes with [FCTN][3]. We place the label AG (for AGain) at line >7D3A so we can branch back there if Erase is used. Lines >7D3A through >7D48 place the starting address of the underlines in register 0; T2 in register 1; and 12 (the number of underlines) in register 2. Then we branch and link to VMBW.

At line >7D4A we begin our key-scan routine. We clear location >8374 to tell the KSCAN routine to scan the whole keyboard, and clear register 4 to keep track of the number of characters that have been input. Then in line >7D50 we load register 5 with the address of our buffer area (>7E00) where we will store the name input and load register 6 with the mask for checking the status of the keyboard.

Next we place the label LP at >7D58, which is the beginning of our key-scan loop. Here we branch and link to >6020 (KSCAN), and at >7D5C we move the status byte into register 1. Then the COC 6,1 instruction checks to see if bit 2 is set (i.e., if a key has been pressed). If not, we branch back to LP. If the bit is set, we move the character from >8375 into register 1. Lines >7D6C through >7D82 do a series of tests. First, we see if input is complete by checking for the [ENTER] key (ASCII 13). If the input is complete, we jump to the END to display the final message. If the ASCII value is not

Mini-Memory Run Option

To use this option, the REF/DEF table located at the high end of the Mini Memory's RAM area must be altered. This process was covered in detail in Part III of this series (HCM, Vol. 4, No. 1), but here's a quick list of what you need to do:

- 1) Use the AORG statement to go to >701C. Here, use the DATA directive to update this location with >7E0C. This is the new First Free Address in memory because it is one location beyond our buffer area.
- 2) When you have entered >7E0C in this location, change the Last Free Address in memory at location >701E. Enter >7FE8 here to make room for our addition to the REF/DEF table.
- 3) Next use the AORG directive to go to address >7FE8. Here, use the TEXT directive to identify our new program by name. Choose any name you want, as long as it is six characters long including spaces. For example, you could enter:

```
7FE8 0000 TEXT 'HELLO '
```

With the cursor at >7FEE, use the DATA directive to enter the starting address (7D2A) of the program.

- 5) Now you type the END directive and press [ENTER] twice. You will be returned to the Mini Memory main menu.

- 6) To run your program, just choose the Run option, and enter HELLO as the name of your program.

Listing 2

```

7D00 xxxx AORG >7D00
7D00 4849 T1 TEXT 'HI! WHAT IS YOUR NAME?'
7D02 2120
7D04 5748
7D06 4154
7D08 2049
7D0A 5320
7D0C 594F
7D0E 5552
7D10 204E
7D12 414D
7D14 453F
7D16 5F5F T2 TEXT '-----'
7D18 5F5F
7D1A 5F5F
7D1C 5F5F
7D1E 5F5F
7D20 5F5F
7D22 4845 T3 TEXT 'HELLO, '
7D24 4C4C
7D26 4F2C
7D28 2000
7D2A 0200 LI 0,100
7D2C 0064
7D2E 0201 LI 1,T1
7D30 7D00
7D32 0202 LI 2,22
7D34 0016
7D36 0420 BLWP @>6028
7D38 6028
7D3A 0200 AG LI 0,228
7D3C 00E4
7D3E 0201 LI 1,T2
7D40 7D16
7D42 0202 LI 2,12
7D44 000C
7D46 0420 BLWP @>6028
7D48 6028
7D4A 04E0 CLR @>8374
7D4C 8374
7D4E 04C4 CLR 4
7D50 0205 LI 5,>7E00
7D52 7E00
7D54 0206 LI 6,>2000
7D56 2000
7D58 0420 LP BLWP @>6028
7D5A 6020
7D5C D060 MOV B @>837C,1
7D5E 837C
7D60 2046 COC 6,1
7D62 16FA JNE LP
7D64 04E0 CLR @>837C
7D66 837C
7D68 C060 MOV @>8375,1
7D6A 8375
7D6C 0281 CI 1,13
7D6E 000D
7D70 1318 JEQ ED
7D72 0281 CI 1,7
7D74 0007
7D76 13E1 JEQ AG
7D78 0281 CI 1,32
7D7A 0020
7D7C 11ED JLT LP
7D7E 0281 CI 1,90
7D80 005A
7D82 15EA JGT LP
7D84 06C1 SWPB 1
7D86 DD41 MOV B 1, *5+
7D88 0420 BLWP @>6024
7D8A 6024
7D8C 0580 INC 0
7D8E 0584 INC 4
7D90 0284 CI 4,12
7D92 000C
7D94 11E1 JLT LP
7D96 0600 DEC 0
7D98 0204 LI 4,12
7D9A 000C
7D9C 0205 LI 5,>7E0B
7D9E 7E0B
7DA0 10DB JMP LP
7DA2 0284 ED CI 4,0
7DA4 0000
7DA6 13D8 JEQ LP
7DA8 0200 LI 0,356
7DAA 0164
7DAC 0201 LI 1,T3
7DAE 7D22
7DB0 0202 LI 2,7
7DB2 0007
7DB4 0420 BLWP @>6028
7DB6 6028
7DB8 0200 LI 0,363
7DBA 016B
7DBC 0201 LI 1,>7E00
7DBE 7E00
7DC0 C084 MOV 4,2
7DC2 0420 BLWP @>6028
7DC4 6028
7DC6 0300 LIM 2
7DC8 0002
7DCA 0300 LIM 0
7DCC 0000
7DCE 10FB JMP $-8
7DE0 END

```

13, we go on to see if it is ASCII 7 (the code for [FCTN][3], which is the Erase option). If it is 7, we jump back to AG, where the underlines will be displayed and the KSCAN can begin anew. Finally, in lines >7D78 through >7D82 we check to see if the character has an ASCII value of at least 32 (a blank) and no more than 90 (capital Z). If it is outside of this range, we jump back to LP.

Putting the Moves on

As each letter is accepted, lines >7D84 through >7D8A save it in the buffer area beginning at >7E00, and print it on the screen at the underline characters. The SWPB 1 in line >7D84 moves the ASCII value of the character to the leftmost (most significant) byte in register 1; then line >7D86 moves this byte to the buffer that has its address in register 5. After each character is moved to the address pointed to by register 5, the register is automatically incremented to point at the next byte of the buffer.

After the character is placed in the buffer area, lines >7D88 and >7D8A immediately branch to Video Single-Byte Write. This echoes the keypress by putting the most recent character on the screen. Register 0 contains the starting address of the character's screen position, and register 1 contains its ASCII code.

Lines >7D8C through >7D94 INCRement register 0 (the address of the screen) and register 4 (the length of the name input), then check the status of the input. If the maximum of 12 characters has not been reached and the [ENTER] key has not been pressed, the program simply jumps back to LP to see what the next input will be. If 12 characters have been input and the [ENTER] key has not been pressed, the computer will go to lines >7D96 through >7DA0. Lines >7D98 through >7D9E prevent the user from entering too many characters. Input will stop at the 12th one, and the program will accept any new input as the 12th character. If [ENTER] has been pressed, the program will jump to the ED label (see line >7D70). If the user chose the Erase option, the program would restart the input (line >7D76).

Now we come to the final section of the program, which begins at the ED label (line >7DA2) and prints the word HELLO followed by the name accepted from the keyboard. This section is reached only if the [ENTER] key was pressed and detected (line >7D6C). Lines >7DA2 through >7DA6 make sure that at least one character has been accepted by comparing register 4 to zero. If register 4 equals zero, no characters have been entered, so the computer will branch back to LP and scan the keyboard for input. If at least one character has been accepted, then lines >7DA8 through >7DB4 display the beginning of the message, starting at screen position 356 (row 11, column 5). Lines >7DB8 through >7DC4 display the user's name that we saved in the buffer at >7E00. Because we have the number of characters in the name stored in register 4, we merely transfer this quantity to VMBW.

Lines >7DC6 through >7DCE form the loop introduced in Listing 1 that allows the processor to be interrupted by the [QUIT] function. The computer will stay in this loop until [FCTN] [QUIT] is pressed. **HCM**

Part I of "Have No Fear" ran in Vol. 2, No. 12 of 99'er HCM; Part II ran in Vol. 2 No. 13 of 99'er HCM; and Part III ran in Vol. 4 No. 1 of HCM. Beginning Assembly Language programmers may also want to consult chapter 5 of The Best of 99'er from Emerald Valley Books.



The RS-232 Interface: Your Link to the Periphery

In the world of computer interfacing, a "standard" isn't always standard. Knowing what your cable "sees" at each end is the key.

by Patricia Swift

If you use a microcomputer, sooner or later you will come across the "RS-232 standard interface." This article explains what the RS-232 interface is and how it is far from standard in practice. It also gives a practical example of how to design a cable to connect an RS-232 device to a computer.

What Is An Interface?

Computers use an interface to communicate with external devices (or "peripherals") If you want to attach a printer to a computer, you must have an interface and a cable in addition to a printer and computer, as shown in Figure 1. The interface is attached to the computer, and is usually a circuit board which is installed inside the main box or peripheral expansion box. The visible part of the RS-232 interface is the connector. Between this and a similar connector on the printer runs a cable with end connectors mated to those on both the printer and the computer. The cable's wires must also be arranged so that the interface and the printer "understand" each other.

How Is An Interface Used?

The RS-232 interface can be used to attach printers, modems, and other peripheral equipment to the computer. This "serial" interface (where data is sent one bit at a time over one wire) is available for most microcomputers, and many peripherals are RS-232 compatible. Unfortunately, there are many variations of the "RS-232 standard"—you can't just hook up your RS-232 printer to your RS-232 interface using a standard cable (with the possible exception of the new "smart cables") and expect it to work. The reasons why will be detailed later.

Another common way to attach a printer to a computer is with a "parallel" interface (where data is usually sent eight bits at a time over several wires). Although there is no single standard for parallel interfaces, there seem to be fewer variations than with the RS-232 interface. When we are unable to make an RS-232 connection work, we can usually use the parallel interface with very little effort. But, because many peripherals do not have a parallel interface, this option may not be open to you.

What Does An RS-232 Connector Look Like?

The most common RS-232 connector is known as the D-type with 25 pins (see Figure 2), although variations

abound: 7-pin, 9-pin, D-, and edge connectors are just a few. Connectors can be male or female; male connectors with pins are usually part of the cable, and socketed female connectors are usually found on the printer and interface. Figure 2 illustrates how the pins or sockets in the connector are numbered. Some RS-232 connectors have tiny numbers stamped near some or all of the pins or sockets.

The EIA RS-232 Standard

The Electronics Institute of America (EIA) has established a standard for the RS-232 interface that determines the signal that belongs on each pin, and the voltage levels of the signals themselves. Figure 3 summarizes this standard.

Common Variations

If every RS-232 interface and peripheral used the EIA standard, there would be no need for articles like this. Many printers, however, use a subset of the standard, and need only some of the signals given in Figure 3. Some devices vary slightly from the standard, using different pins for some signals. Sometimes the signal levels used are in different voltage ranges. For example, 0 to 5V is not defined; but this problem is beyond the scope of this article.

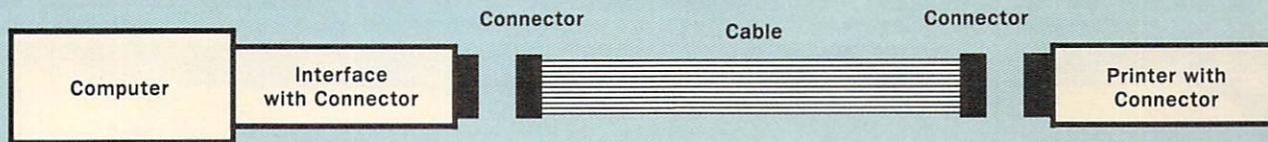
To understand what your printer and computer require to communicate, you must look at the RS-232 charts supplied by the printer and the computer interface manufacturers. We'll do that next, with a specific example, to give you some ideas of what to look for.

Example

We will look at attaching an Okidata Microline 82A printer to a Texas Instruments 99/4A computer via a TI RS-232 interface card. Both use 25-pin D-connectors like the one in Figure 2. The question is: How is the cable constructed? The cable will contain several wires, each wire connecting a pin from the interface at the computer end to a pin at the printer end.

The Okidata manual has a chart of serial interface signals and shows several possible cable arrangements for connecting the 82A to various computers (Okidata calls them "controllers"). For this example, we'll use the simplest arrangement listed. In Figure 5 (showing the TI to 82A cable wiring), the information in the column headed OKIDATA is straight out of the printer manual. The directions for the signals are also shown

Figure 1: ATTACHING A PRINTER TO A MICROCOMPUTER



in that manual. The diagram shows that only four pins need to be connected between the printer and "controller"—two grounds, one line for data into the printer, and one supervisory send data (or SSD) line for the Okidata to signal when the printer is busy or ready to receive data. These are pins 1, 7, 3, and 11 respectively. Pins 6 and 20 also need to be "jumped" (wired together) on the printer end of the cable. The pin numbers on the computer side are not given in the Okidata manual, so we'll need to refer to the TI RS-232 interface manual for the rest of the story.

"The phrase 'RS-232 standard interface' is deceptive because so many manufacturers deviate from the standard."

The 99/4A RS-232 manual has a chart (shown in Figure 5) that describes its RS-232 connector. We will use the chart to fill in the pin numbers in the left-hand part of Figure 5. Okidata and Texas Instruments do not use the same mnemonics for similar functions—this situation is typical in the field. But we can locate the proper pins by deduction. Pins 1 and 7 are grounds; no problem there. To locate the pin for data out, we look for an output pin for data. The only candidate on the chart is pin 3. TI calls it TX while Okidata calls it TD. Finally, we have to locate an input pin for control, to be connected to the Okidata's SSD pin 11. The only input control line on the interface is pin 20, called "Data Terminal Ready."

Figure 4 shows the cable you'd need to connect the two devices. With this simple cable, the serial interface will work at speeds up to 1200 baud.

HCM

Figure 2: PICTURE OF RS-232 25-PIN D-CONNECTOR Looking at the Printer

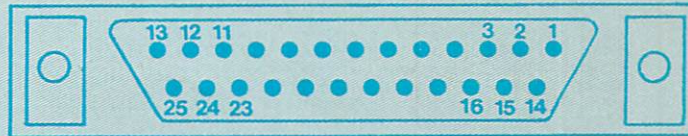


Figure 3: EIA RS-232 STANDARD, REVISION C

PIN	SIGNAL	SOURCE
1	PROTECTIVE GROUND	NONE
2	TRANSMITTED DATA	PRINTER
3	RECEIVED DATA	COMPUTER
4	REQUEST TO SEND	PRINTER
5	CLEAR TO SEND	COMPUTER
6	DATA SET READY	COMPUTER
7	SIGNAL GROUND	NONE
8	CARRIER DETECT	COMPUTER
11	REVERSE CHANNEL	PRINTER
20	DATA TERMINAL READY	PRINTER

SIGNAL LEVELS:	-25 TO -3V	-3 TO +3V	+3 TO +25V
DATA SIGNAL	MARKING	NOT DEFINED	SPACE
TIMING/CONTROL	OFF	NOT DEFINED	ON

Figure 4: 99/4A RS-232 PIN DEFINITIONS

PIN	DIRECTION	SIGNAL
1		GROUND
2	INPUT	DATA IN (RD)
3	OUTPUT	DATA OUT (TX)
5	OUTPUT	CLEAR TO SEND (CTS)
6	OUTPUT	DATA SET READY (DSR)
7		GROUND
8	OUTPUT	DATA CARRIER DETECT (DCD)
20	INPUT	DATA TERMINAL READY (DTR)

Figure 5: THE 99/4A TO OKIDATA 82A PRINTER CABLE

COMPUTER			OKIDATA		
SIGNAL		PIN	PIN	SIGNAL	
GROUND	PG	1	1	PG	GROUND
DATA OUT	TD/TX	3	3	RD	RECEIVED DATA
GROUND	SG	7	7	SG	GROUND
DATA TERM READY	DTR	20	11	SSD	SUPERVISORY SEND DATA
			6	DSR	DATA SET READY
			20	DTR	DATA TERM READY

jumper pins 6 and 20



Saving Applesoft Numeric Arrays To Disk With DOS 3.3

```

1000 REM *****
1100 REM * SAVE ARRAY DEMO *
1200 REM *****
1300 REM BY M.D. BROWNSWORTH
1400 REM HOME COMPUTER MAGAZINE
1500 REM VERSION 4.4.1
1600 REM APPLE // FAMILY APPLESOFT
1700 REM DEMO PROGRAM--SAVES AND RELOA
DS A FLOATING-POINT ARRAY
1800 CLEAR : HOME : DS = CHRS (4)
1900 SIZE = 10 : DIM A(SIZE)
2000 REM LOAD A( ) WITH DUMMY VALUES
2100 FOR I = 0 TO 10
2200 A(I) = I : PRINT A(I)
2300 NEXT I
2400 PRINT
2500 REM SET VARIABLES EQUAL TO ARRAY
START AND LENGTH
2600 A(0) = A(0) : REM REFERENCING ARRAY
SETS POINTERS 131 & 132
2700 POKE 60, PEEK (131) : POKE 61, PEEK
(132) : START = PEEK (60) + PEEK (
61) * 256 : REM START OF A( ) DATA
ELEMENTS
2800 LENGTH = (SIZE + 1) * 5 + 16 : REM L
LENGTH OF A( ) IN BYTES--NUMBER OF E
LEMENTS, INCLUDING A(0), MULTIPLIED
BY FLOATING-POINT ELEMENT LENGTH (
5) PLUS OFFSET
2900 REM SAVE A( ) TO DISK AS A BINAR
Y FILE, GIVING IT A NAME
3000 PRINT DS : "BSAVE ARRAYNAME,A" : START
: "L" : LENGTH
3100 REM CLEAR THE ARRAY WITH ZERO'S
3200 FOR I = 0 TO 10
3300 A(I) = 0 : PRINT A(I)
3400 NEXT I
3500 PRINT
3600 A(0) = A(0) : REM REFERENCE THE ARR
AY AGAIN
3700 REM RELOAD THE DATA BACK INTO A(
)
3800 PRINT DS : "BLOAD ARRAYNAME,A" : START
3900 REM PRINT THE VALUES FROM THE ARR
AY AGAIN
4000 FOR I = 0 TO 10
4100 PRINT A(I)
4200 NEXT I
4300 END

```

Under Apple's ProDOS, the STORE and RESTORE commands make saving arrays to disk far easier than under DOS 3.3. (These commands should not be confused with Applesoft cassette commands STORE and RECALL.) Prior to ProDOS, the only means of saving arrays to disk was through sequential and random-access text files. It's disadvantageous to use text files to save arrays because an extra routine must be written to move the array data to the text file. Here's a trick that saves arrays directly to disk from DOS 3.3.

Applesoft allocates array space dynamically—i.e., DIMensioning an array with a DIM statement in the program reserves space for that array in memory. The variable type is important because floating point array elements require 5 bytes, 3 bytes more than an integer variable's 2 bytes per element. Applesoft stores arrays in the memory area immediately following the simple variable storage located after the program itself. Thus, the array is relocated in memory as the simple variable space changes size.

Applesoft uses pointers—pairs of locations on page zero—to keep track of where in memory the arrays reside. The pointers at locations 107 and 108 (\$6B and \$6C hex) are the respective low and high bytes of the starting address of the entire array area. Locations 109 and 110 (\$6D and \$6E) point to the end of the array space. Another pair of pointers, 131 and 132 (\$81 and \$82), contain the address of

the last-used variable. This last set of pointers is the key to saving a particular array.

To save the entire array space, set a variable equal to the start of the array area by PEEKing the appropriate pointers: START = PEEK (107) + PEEK (108) * 256. Next, set a variable for the top of the array space: TP = PEEK (109) + PEEK (110) * 256. Then derive the length of the array area by subtraction: LENGTH = TP - START + 1. Finally, save the array space as a binary file:

PRINT DS;"BSAVEALLARRAYS,A";START;"L";LENGTH.

The accompanying program demonstrates the process of saving and reloading a single floating point array named A(). To reference the targeted array, set the pointers at locations 131 and 132 with this statement: A(0) = A(0). The values at these locations must be placed in another pair of locations, 60 and 61 (\$3C and \$3D). Next, line 270 sets the start of the A() array data elements: START = (PEEK (60) + PEEK (61) * 256). Note that SIZE is a variable which holds the size at which A() was initially DIMensioned. To calculate the length of a floating-point array, use this formula: LENGTH = (SIZE + 1) * 5 + 16. To save an integer array, besides changing A() to A%(), change the element byte length in line 280 from 5 to 2: LENGTH = (SIZE + 1) * 2 + 16. Now to save the array:

PRINT DS;"BSAVE NAMEOFARRAY,A";START;"L";LENGTH

To retrieve arrays saved with this procedure, they must be BLOADED by either the same program that saved them in the first place, or by another program exactly the same in length. Otherwise, due to Applesoft's dynamic array locating, the array could be loaded into the wrong place. To get around this restriction, alter the end-of-program pointers in the programs to make the arrays load in the same memory space every time.

—Michael D. Brownsworth



Lower Case Letters for TI-99/4A

Would you like to have true lower-case letters (rather than shortened capitals) available with TI BASIC? You can—simply by turning the shortened character set that the TI-99/4A uses into an alternate set by redefining characters 97 through 122 using the CALL CHAR command.

```
100 CALL CHAR(97,'0000007008384874')
110 PRINT "a"
120 GOTO 120
```

The above example shows you how to change the letter A from shortened uppercase to true lowercase using TI BASIC. Unfortunately, this solution uses some memory, and program initialization time is increased. If you have a Mini-Memory module, however, these problems can be solved by installing the following assembly language routine:

```
AORG >7D00
AL DATA >0000,>0070,>0838,>4874
DATA >0040,>4078,>4444,>4478
DATA >0000,>0038,>4440,>4438
DATA >0004,>043C,>4444,>443C
DATA >0000,>0038,>447C,>403C
DATA >0018,>2420,>7020,>2020
DATA >0000,>0438,>4438,>047C
DATA >0040,>4078,>4444,>4444
DATA >0010,>0030,>1010,>1038
DATA >0008,>0018,>0808,>4830
DATA >0040,>4048,>5070,>4844
DATA >0030,>1010,>1010,>1038
DATA >0000,>0078,>5454,>5454
DATA >0000,>0058,>2424,>2424
DATA >0000,>0038,>4444,>4438
DATA <0000,>0078,>4478,>4040
DATA >0000,>0038,>4454,>4834
DATA >0000,>0058,>6440,>4040
DATA >0000,>003C,>4038,>0478
DATA >0010,>3810,>1010,>1408
DATA >0000,>0048,>4848,>4824
DATA >0000,>0044,>4428,>2810
DATA >0000,>0044,>5454,>5424
DATA >0000,>0044,>2810,>2844
DATA >0000,>0044,>2418,>1060
DATA >0000,>007C,>0810,>207C
LWPI >70B8 SET UP WORK SPACE REGISTERS
MOV R11,R10 SAVE LINK TO BASIC
LI R0,>0608 ADDRESS OF ASCII 97 IN VDP RAM
LI R1,AL ADDRESS OF CHARACTER CODES
LI R2,208 208 (26*8) BYTES TO WRITE
BLWP @,>6028 MULTIPLE BYTE WRITE
B *R10 RETURN TO BASIC
END
```

Next, enter the following to name the routine LOWCAS, and add it and the entry point (>7DD0) to the REF/DEF table:

```
AORG >7FE8
TEXT 'LOWCAS'
DATA >7DD0
```

Then, you can call it from a BASIC program using a CALL LINK ('LOWCAS') statement. By keeping this routine permanently loaded in a Mini-Memory module, it will be available for you any time from TI BASIC. It only needs to be run once per program.

—HCM Staff



Double Your DOS Master



The MS-DOS 2.10 master disk is a single-sided disk that comes formatted 8 sectors per track (to make it compatible with older IBM machines). The newer DOS format, supported by the latest PCs and PCjr, is double-sided with 9 sectors per track. This chart shows the amount of storage available when using various MS-DOS 2.10 formats:

Format		Capacity in Bytes
sides	sectors/track	
1	8	160,256
1	9	179,712
2	8	322,560
2	9	362,496

When you make a back-up of your DOS 2.1 master using DISKCOPY, the result is a disk with the smaller single-sided format—a real waste of disk space (if your machine supports the new format). But there is an alternative: Instead of using DISKCOPY, format the disk with the /s option using the following command (with your DOS master in drive A):

```
A> format b: /s
```

Place the new disk in the drive (drive b: if you have a two-drive system) when the following appears on the screen:

**Insert new diskette for drive B:
and strike any key when ready**

Formatting then begins. When the process is complete, the following message will appear on screen:

```
Formatting . . . Format complete
System transferred
362496 bytes total disk space
40960 bytes used by system
321536 bytes available on disk
```

Format another (Y/N)?

Answer this question by entering N. Now, to copy the rest of the DOS disk onto this double-sided, 9-sector disk, make sure the DOS master is in drive A and type the following command:

```
A> copy a:*. * b:
```

If you have a two-drive system, the rest of the disk will be copied to your new disk. If you have only one drive, you will be prompted to switch disks when necessary. When the process is complete, the directory of your new disk will reveal that you still have 204,800 bytes free instead of the meager 28,672 you would have had if you used DISKCOPY. This extra space is enough to similarly copy the entire DOS supplemental disk and still have 91,136 bytes free for any other files. With high-quality disks priced at about \$5 each, this method represents a considerable saving of both disk space and money, not to mention convenience.

—Roger Wood

TECH NOTES

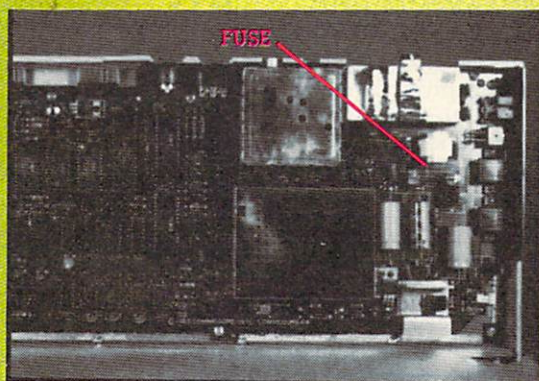
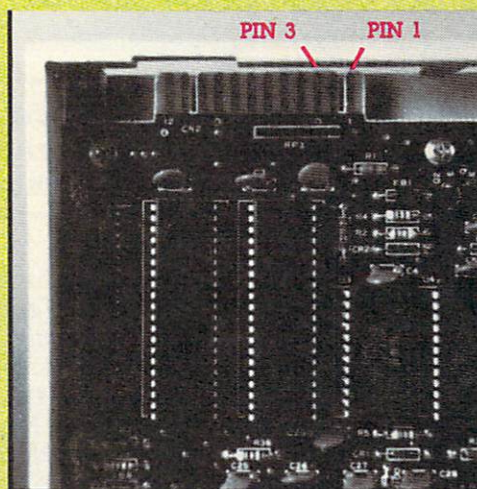


Installing a Reset Switch



There are times when advanced Commodore programmers wish for a way to reset their computer without erasing everything in RAM. Each time a program hangs up during debugging—which can happen when you are doing some complex programming or moving basic pointers around in memory—you must turn off the power to reset the machine. But, there is an easy way to install a reset switch in your C-64 or VIC-20: you can purchase a simple switch (single pole-single throw, momentary contact push-button type), at any electronics parts store, and install it yourself with the instructions below.

Turn off the power to your Commodore and then disconnect all cables and peripherals. You will need a small Phillips screwdriver to remove the three screws holding down the cover. As the computer is facing you, find the far left hand port (see photo at right). The particular pins of interest are pins 1 and 3. Simply solder one wire to pin 1 and the other wire to pin 3. (One thing to remember when soldering: You must be sure to solder the wires on the pins far enough back so that they don't interfere with the insertion of the cartridges.) You could run the wires out the back port and just leave your switch hanging loose. It is much better, however, to securely mount the switch on the side of your machine. To do so, drill a hole into the top left-hand side of your computer cover and mount the switch there.



That's all there is to it. Put everything back together and connect all the power cords and peripherals, and your Commodore 64 or VIC-20 is better than new.

WARNING: If, while experimenting with the connector pins, your computer dies, don't panic. Chances are you shorted the wrong pins together and blew the fuse inside the machine (see photo at left). Just replace that fuse and be more careful next time.

—Steve Nelson

[Home Computer Magazine, the publisher, and the author shall not be held liable for unsuccessful project completion. Modification of your C-64 or VIC-20 will probably void any remaining warranty. Proceed at your own risk.]

ONE FOR THE MONEY, TWO FOR THE SLOW:

ADDING A SECOND DRIVE TO THE PCjr

by David G. Brader
HCM Staff

Much software designed for the IBM PC would run faster and more conveniently on the PCjr if the smaller machine had a second disk drive. Now you can add your own . . .

[Caveat: What follows is a hardware modification procedure for a PCjr with 128K of memory and the IBM single-disk system. The successful implementation of this procedure will result in a PCjr system capable of accessing two disk drives. This project assumes that you have the requisite technical skills and knowledge to complete it. We have done our best to include all the necessary data and procedural steps to guide a knowledgeable technician. Home Computer Magazine, its publisher, and the author assume no liability for unsuccessful project completion or damage to any of your equipment. Modification of the IBM PCjr will probably void any remaining warranty. These instructions are offered as is, and readers should proceed at their own risk—Ed.]

The Drive For Success

Tired of swapping diskettes in and out of your single-drive PCjr? What if you were told, "For less than \$300 (depending on how much you pay for the drive) and a few hours work you can have a second disk drive on Junior" Well, now you can.

When the PCjr was first announced as a single-drive system, we said, "Sure, bet all you have to do is stick an extension cable out the back of the box and daisy-chain a second drive to it." We were wrong. IBM very carefully de-engineered their disk-drive controller design making it impossible to simply change cables and add a second drive. Now here was a challenge!

Our answer to this challenge? Modification of the disk-drive controller board, requiring only the addition of two ICs (integrated circuits), the opening of two circuit paths on the board, the addition of ten wires to the board, and the construction of a new flat cable.

Initial tests of the hardware uncovered a roadblock—the IBM PCjr BIOS program (Basic Input/Output System stored in ROM memory) was the culprit. When Junior is reset or powered up, BIOS sets what are called

"equipment status software flags" in RAM memory which always report that only one disk drive (device name A:) is connected.

I therefore had to design a software fix to fool the system. With some special batch files installed on the PC-DOS 2.1 disk, I retested the system. DIR B: was entered at the keyboard, the red light on the second drive came on, and a moment later the directory listing of the data disk appeared on the PCjr screen—it worked!

Three To Get Ready

Before attempting this modification, you need to accomplish three tasks: (1) Read this entire article and study the illustrations carefully, (2) purchase the parts needed for the modification, and (3) gather the necessary tools. Use the parts list in Figure 1 and the tool list in Figure 2 as checklists to make sure you have everything you need. When shopping for a second disk drive, make sure the drive is an IBM-compatible, double-sided, double-density model, either full-height or the newer half-height size, and that it comes with a case and power supply. Don't order a cable with the drive because a special one must be built for this application.

Voiding The Warranty

It is important that you have good lighting for this work. Disconnect all cables from the rear of the PCjr including the power cord. Using a wide-blade screwdriver, pry up the top cover at the rear of the unit. Slide the cover back and remove it. Locate the disk-drive controller board with the wide flat cable connecting it to the disk drive. Disconnect the cable at the disk drive and carefully remove the controller board. Remove the flat cable from the controller board and lay it aside. Place the controller board, component side up, on a clean surface.

Referring to the board diagram in Figure 3, use a small, sharp pair of diagonal cutters to cut pin 1 of integrated circuit ZM21 free from the printed circuit board (snip it as close to the board as possible). Carefully bend the lead up and out from the board. We will be soldering a wire to it later.

One other integrated circuit (IC) pin also requires this procedure, but the task will be more difficult because pin 13 of ZM4 is partially hidden by a capacitor mounted on the board. Using the soldering iron, melt the joint at the left lead of the capacitor while gently pulling up on the lead with the long-nosed pliers until the capacitor lead comes free from the board. Carefully bend the capacitor away from the area of pin 13 on IC ZM4. Using the small cutters, snip the pin free from the board. This is easier said than done—use care not to damage adjacent pins. Bend the pin up and out from the board (so that a wire can be soldered to it). Move the capacitor back in place and resolder its lead in the hole from which it was removed.

Congratulations, 10% Of The Job Is Done!

Now the two new ICs must be prepared for mounting on the printed circuit board. Because there is insufficient room to place these ICs on the main board area, they must be "piggy-backed" onto other ICs that are already mounted on the board. Bend the "parasite" IC's pins straight out, leave them alone, or cut them off, depending on the use of each one (as described below and shown in the accompanying photos). The pins are counted counterclockwise as viewed from the top of the IC. Pin 1 is always located to the left of the dimple that is in one end of an IC.

The first one of our ICs is piggy-backed to the existing IC at location ZM1 (see Figure 3) which is of the same type as the new IC (74LS175). Investigation of the design shows that both ICs use or need to be connected to the same CLEAR, CLOCK, and D0 inputs (pins 9, 1, and 4 respectively). They also share the +5 volt power and ground pins (pins 16 and 8). By bending all the other pins (numbers 2, 3, 5, 6, 7, 10, 11, 12, 13, 14, and 15) up and straight out from the IC body, the IC is ready for mounting (actually, pins 10 through 15 can be clipped off because they are not used). The prepared IC is shown in the accompanying photo. To actually mount this parasite IC, it must be carefully aligned and seated on the host IC (ZM1), and all shared pins joined with solder (pin 1 to pin 1, pin 4 to pin 4, etc.). Be careful not to mount it backwards.

Figure 1. TOOL LIST
For PCjr Disk Drive Controller Modification

DESCRIPTION	USE
Small long-nosed pliers	Bending leads on ICs
Very small diagonal-side cutters	Cutting IC leads
"No nick" wire strippers	Removing wire insulation
Low-wattage soldering iron	Soldering connections
Fine-wire rosin-core solder	Soldering connections
Spool of #28 tinned-copper wire with insulation	additional circuitry
Wooden-jaw vise	Making flat cable
Screwdriver set	Various uses

On the other parasite IC (74LS10) to be mounted there are only two shared pins. The +5 volt power and ground pins (pins 14 and 7) are the ones used to piggy-back this IC. All other pins are bent up and out from the IC body. The unused pins, 1, 2, 12, and 13, may be clipped off. The host IC for this parasite is in location ZM9, as shown in Figure 3 (the host IC is a type 74LS08). Carefully align and seat the new IC on the ZM9 host and solder pin 14 to pin 14 and pin 7 to pin 7.

Time To Get Wired, Folks!

There are ten wires that must now be connected to the printed circuit board. Using a low-wattage soldering iron, attach the wires according to the chart in Figure

4. Refer to the partial schematic in Figure 5 showing the affected portion of the disk controller logic, the pictorial illustration in Figure 3 for parts location, and the photograph of the wired board. Double-check each solder connection that you make for correct location and a solid joint. In each case, inspect with a magnifying glass to see if any of the solder flow caused a short to an adjacent connection.

Figure 2.

PARTS LIST

QTY	DESCRIPTION	MANUFACTURER
1	SN74LS175 Integrated Circuit	Texas Instruments
1	SN74LS10 Integrated Circuit	Texas Instruments
2	RF16-2852-0 Ribbon Cable Receptacle	Texas Instruments
1	SFM34-2841-0 Ribbon Cable Socket	Kel-Am Inc.
4	Feet of GEX28-34 Ribbon Cable	Kel-Am Inc.
1	IBM compatible dual-sided double-density Floppy disk drive with power supply and case. We used an MPI model 52 with a case and supply from a broken drive.)	MPI

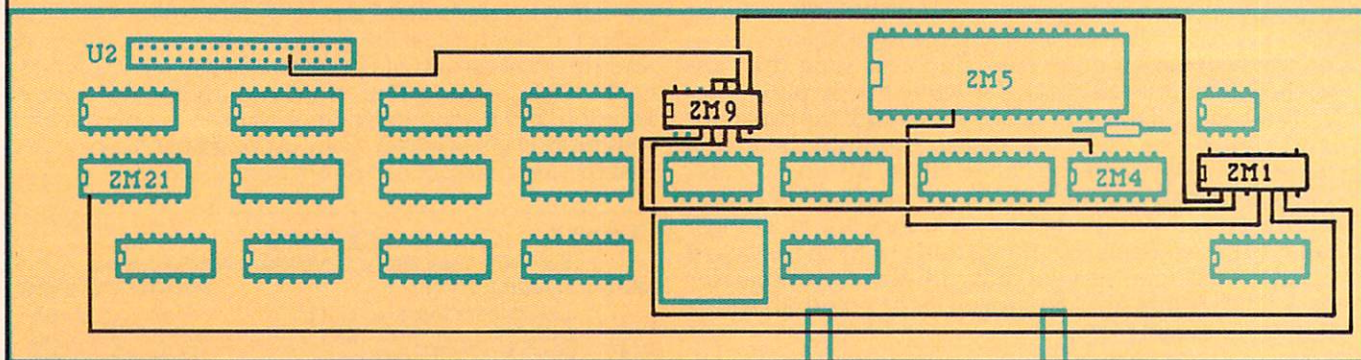
The wire that connects ZM9 (piggyback) pin 8 to connector U2 pin 10 must be installed with additional precautions. First, identify the correct U2 connector pin by holding the board in front of you and counting 5 over from the right side on the bottom row as shown in Figure 3. The wire must be wrapped tightly at the very bottom of the pin and soldered with a minimum of solder. If the length of the pin is obstructed by the wire being placed too high or excess build-up of solder, the flat ribbon cable connector that mates with U2 will not seat properly, causing possible intermittent operation of the disk system. After the wire is installed, try pressing the old ribbon cable connector onto U2 to make sure it fits tightly over all pins. After checking the connector fit, carefully remove the flat cable from U2 and set it aside. The disk-drive controller board is now ready for installation in the PCjr. Place it aside for the moment.

Making The New Flat Cable

A new cable must be made that replaces the old flat cable and that extends out the side of the PCjr. This requires the installation of three specially-matched, ribbon cable connectors at the right points along the length of a flat ribbon cable. These connectors are usually installed using a special arbor press tool and custom dies, but it is possible to install them yourself. Slide the ribbon cable through the gap formed by the main body of the connector and its plastic backing plate until the connector is located at the right spot on the cable. Using your hands, press the plastic backing plate into the main connector body as hard as you can to keep it from moving during the final step. Using a wood-working vise (don't use a vise with metal jaws), clamp the new connector assembly in such a fashion as to apply further pressure to seat the plastic backing plate into the connector's main body. Tighten the vise slowly while watching the gap between the flat cable and the connector body. Stop tightening when the cable and connector appear to have formed an airtight seal. *Warning: over-tightening the vise may damage the connector-cable assembly.*

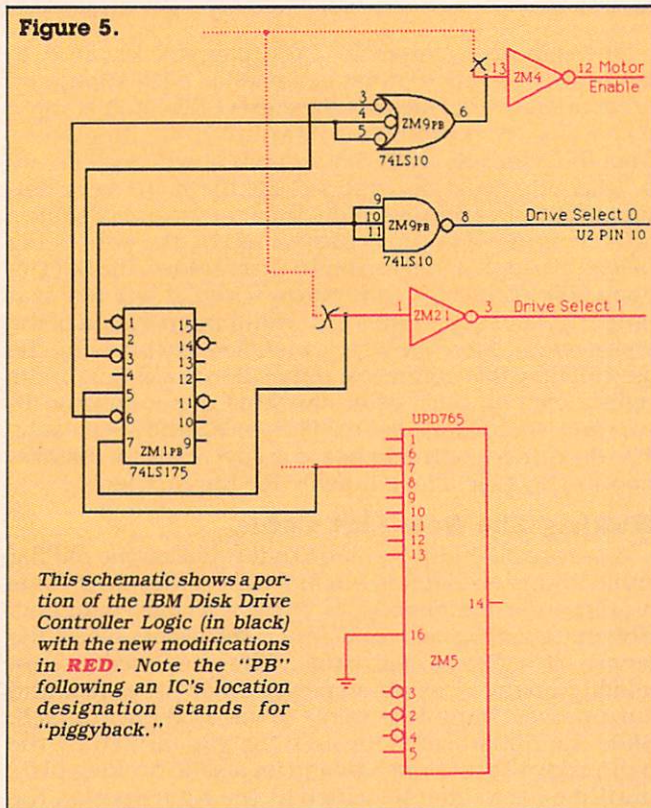
Starting with one end of the ribbon cable, install the connector that will mate with the controller board match the distance between the connectors on the old IBM flat cable. Install this connector on the same side of the cable as the old IBM cable. The last connector, which will mate with the new external disk drive, is mounted on the other end of the cable (facing the same side as the other disk drive connector).

Figure 3.

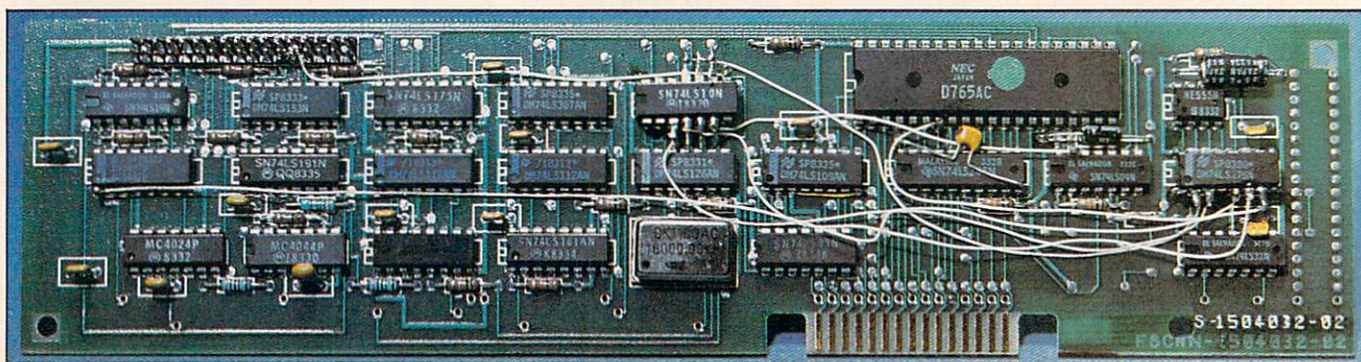


ABOVE: This diagram shows the layout of the major components on the IBM PCjr Disk-Drive Controller board. The two new ICs and new wiring are shown in black. Note the locations of ZM4 pin 13 and ZM21 pin 1; both need to be cut free from the circuit board prior to the addition of the new wires (see text).

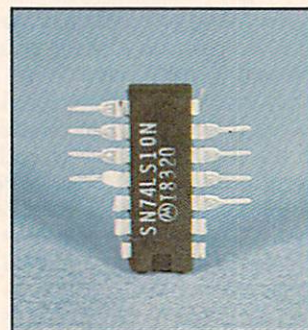
Figure 5.



BELOW: This photo shows the IBM PCjr Disk-Drive Controller board after the completion of modifications. Note the new wiring in white and the "piggyback" ICs at locations ZM1 (74LS175) and ZM9 (74LS10).



RIGHT: The prepared "piggyback" IC for location ZM9 (74LS10) is shown after its pins have been formed (or removed) per the instructions in the text.



RIGHT: The prepared "piggyback" IC for location ZM1 (74LS175) is shown here after its pins have been formed (or removed) per the instructions in the text.

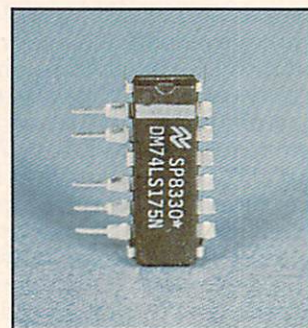
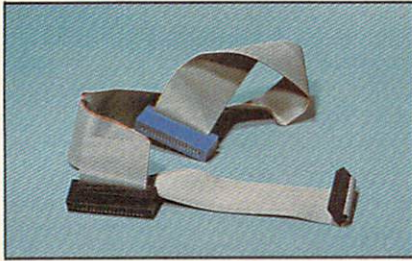


Figure 4.

WIRE CONNECTION CHART FOR CONTROLLER BOARD FROM LOCATION TO LOCATION

IC	PIN#	IC	PIN#
ZM1(piggyback)	2	ZM9(piggyback)	9
ZM9(piggyback)	9	ZM9(piggyback)	10
ZM9(piggyback)	10	ZM9(piggyback)	11
ZM1(piggyback)	3	ZM9(piggyback)	3
ZM1(piggyback)	5	ZM5	7
ZM1(piggyback)	6	ZM9(piggyback)	4
ZM9(piggyback)	4	ZM9(piggyback)	5
ZM1(piggyback)	7	ZM21	1
ZM9(piggyback)	6	ZM4	13
ZM9(piggyback)	8	U2(connector)	10

The completed project with the modified controller board and new cable installed prior to "buttoning up."



Shown above is the new cable that connects the modified controller board to the two disk drives (the original drive inside the PCjr and the additional drive on the outside). The portion between the controller board connector and the internal drive has been sized and formed to match the original IBM cable.

Notice how the old IBM cable was folded, and duplicate folds in the new cable. [*Home Computer Magazine* has a kit available for this project. It includes the two integrated circuits and the new finished flat cable. The cost of the kit is \$49.95.—Ed.]

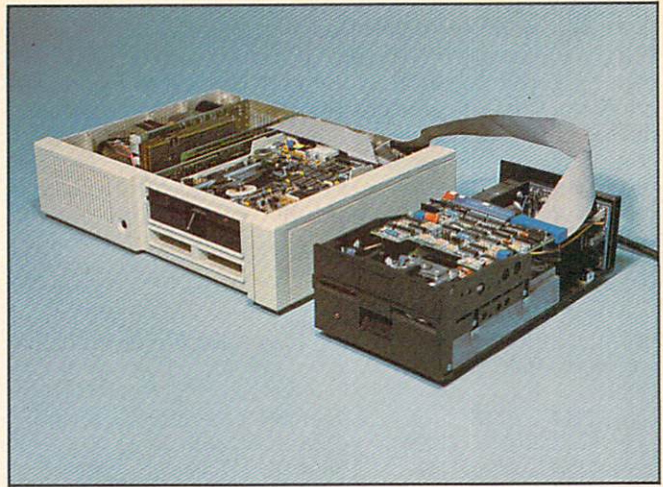
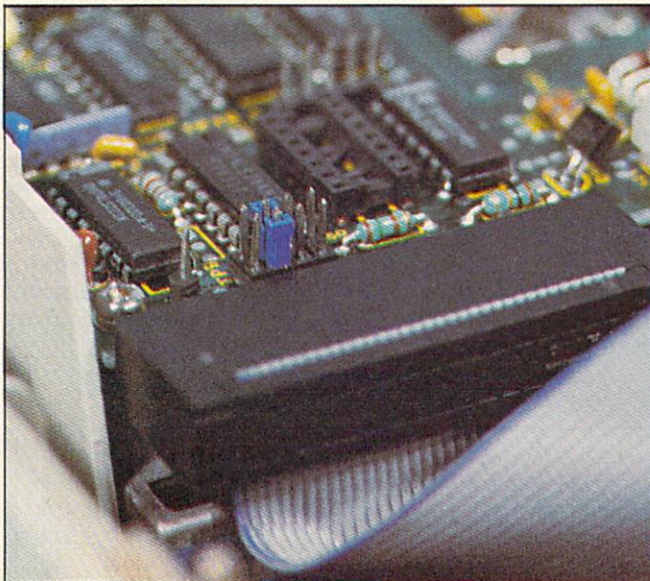
Cable Installation

Install the modified controller board in the PCjr, connect the new cable to the controller board, and connect the internal disk drive. Fold the free end of the cable so that it passes over the right side where the slot is molded into the case lip. Crease the cable so it will pass up and over any attachment when the PCjr lid is replaced.

Disk Drive Address Selection

You've got one last item to take care of before replacing the PCjr lid—the internal disk-drive selector jumper. Notice that there are four sets of pins on the disk-drive circuit board next to the edge-type connector with the new flat cable. A plastic and metal jumper device is mounted on the second set of pins from the left. Pull this jumper off that set and press into the far left set of pins. This will ensure that the system recognizes this disk drive as "device name A:" which

The IBM internal disk drive's selection jumper is shown in its factory installed location on the left. It must be moved to the new location as shown on the right (see text).

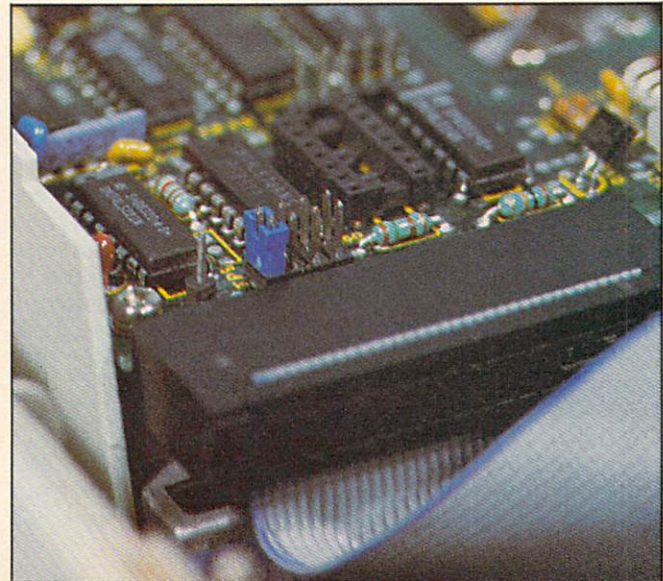


is the default drive at start-up time. Replace the PCjr lid, being careful to guide the cable out the slotted cover lip.

Place your new external disk drive to the right of the PCjr and remove its cover. The instructions that follow are general in nature because we have no way of knowing the precise configuration of the new drive that you purchased.

"Initial tests of the hardware uncovered a roadblock—the IBM PCjr BIOS . . ."

The new disk drive must be set up as "device name B:" for the Disk Operating System. Locate the drive-select jumper area on the drive's circuit board. Usually, it will have letters printed on the circuit board next to it such as DS0, DS1, DS2, and DS3 (sometimes they are numbered 1, 2, 3, and 4). Make sure that all of these jumper options are "open" except the second from the lowest, which should be jumpered. This will select this drive as the second one ("B:"). Now locate the "load resistor" IC pack socket near the drive-select jumper area. If the socket is not empty, remove the resistor pack and discard it. Locate the edge-type connector of the new drive. Press the last connector of the new flat cable



(coming out of the PCjr) onto this edge-type connector. Plug in the power on the new drive and turn on its power switch. If the red light of the new drive comes on and stays on, the flat cable connector is upside down. Turn off the power of the second drive, reverse the flat cable connector at the new drive end, and try again. The red light should not come on. Finally, replace the external disk drive's cover while guiding the new cable out the unit's cable slot.

Fooling PCjr BIOS

Any diskette that you use to "boot" from will need to have at least the following files on it if it is to utilize the new disk drive:

FILE NAME	ORIGIN
COMMAND.COM	IBM DOS DISK
DISKCOPY.COM	IBM DOS DISK
DEBUG.COM	IBM DOS DISK
AUTOEXEC.BAT	SOURCE GIVEN BELOW
BOOT.BAT	SOURCE GIVEN BELOW
MODBOOT.BAT	SOURCE GIVEN BELOW
SWITCH.BAT	SOURCE GIVEN BELOW

Make certain that the diskette has been formatted with the /S option so that the IBM PC-DOS 2.10 and the COMMAND.COM file may be installed before installing the rest of the files. Then copy the DEBUG.COM and DISKCOPY.COM files from the master IBM PC-DOS 2.10 diskette onto the new boot disk. You can use a text processor, such as the IBM EDLIN that comes on the master PC-DOS 2.10 disk, to create the four BATCH files described here.

"... BOOT will indirectly cause changes in the information that BIOS stored in RAM. . ."

The AUTOEXEC.BAT file must contain the following lines exactly as they appear here, in the order that they appear:

```
IF EXIST switch.bat GOTO first
GOTO last
:first
RENAME switch.bat off.bat
BOOT
:last
RENAME off.bat switch.bat
DATE
TIME
```

AUTOEXEC.BAT is automatically executed each time the PCjr system is powered up or reset. As it executes, it first checks to see if the file SWITCH.BAT exists on the disk in drive A:. If it does (and it should) the program logic will GOTO the line labeled :last. Here the program renames the SWITCH.BAT file to OFF.BAT and executes the file called BOOT.BAT. BOOT will indirectly cause changes in the information that BIOS stored in RAM and will affect a restart of the system. This means that AUTOEXEC.BAT is executed again from the start. This time the SWITCH.BAT file doesn't exist, so the program logic will GOTO the label :last. At this point, the OFF.BAT file is renamed to SWITCH.BAT, and the DATE and TIME prompts appear.

The BOOT.BAT file simply starts up the IBM DEBUG.COM utility with directions to take debug commands from the MODBOOT.BAT file instead of the keyboard:

```
DEBUG<MODBOOT.BAT
```

The MODBOOT.BAT file causes four things to happen: It (1) assembles a short assembly language routine starting at location hexadecimal 9080 in memory; (2) executes this same routine causing the bit in the system's Equipment Status byte (which signals that a second disk drive is on line) to be set; (3) loads the "boot track" from the diskette into memory starting at Hexadecimal 7C00; and (4) executes this boot. MODBOOT.BAT looks like this:

```
AO 0:980
XOR AX,AX
MOV DS,AX
OR BY (410),40
NOP
[insert a blank line here to stop assembly.-Author]
G=0:9080 9089
L 0:7C00 0 0 1
G=0:7C00
```

The last file to be created allows AUTOEXEC to detect where it is in its sequence. The SWITCH.BAT file is only a dummy which must exist on the disk, but may contain anything at all. We simply entered a comment line describing its function as follows:

```
REM This is a dummy file used in booting.
```

Once these four files are on your new boot disk, label that disk as the PCjr Dual Disk Drive System Disk.

Now, Go Man Go!

This is it. Make sure the new Dual Disk Drive System Disk is in the A: drive (the one in the PCjr case). Place any other good IBM formatted diskette in the new external drive. Turn on the power to all system units (the PCjr itself should always be last), and watch the screen. After about 45 seconds of clicks, flashes, and text scrolling on the screen, you should see the prompt for the date. Enter the date and the time.

The big test is to now try accessing the second drive. Enter DIR B: and press the [ENTER] key. You should see the second disk-drive light come on, and the directory of the diskette in that drive should appear on the screen. If it doesn't, recheck each of the new files that you placed on the boot disk for errors. Also, have your modification work checked by someone else for errors. Once you are successful, place a blank diskette into the new drive (B:), enter DISKCOPY A: B: and press [ENTER]. Press any key to start the process. Watch how fast the system now copies a disk! Isn't that worth all this work? Label the new disk as the working backup to the Dual Disk Drive System Disk and copy any other files that you usually use (such as the EasyWriter II system files).

OK, that's it. Have fun exploring the wonderful world of IBM PC software!

Note: Listed above are the manufacturers of the specific products that were used in our project. There are other manufacturers of similar products that will work as well. Sources for these items are major electronics component distributors and mail order houses. Sorry, you are on your own when locating the disk drive.

Home Computer Magazine can supply a professionally made cable and the two ICs in a kit. To order, send \$49.95 to PCjr Disk Kit, Home Computer Magazine, 1500 Valley River Drive Suite 250, Eugene, OR 97401.



TM

LOGO SPREADSHEET

If you thought LOGO was limited to moving a turtle around the screen, this spreadsheet application will change your mind.

by Rich Haller
and the HCM Staff

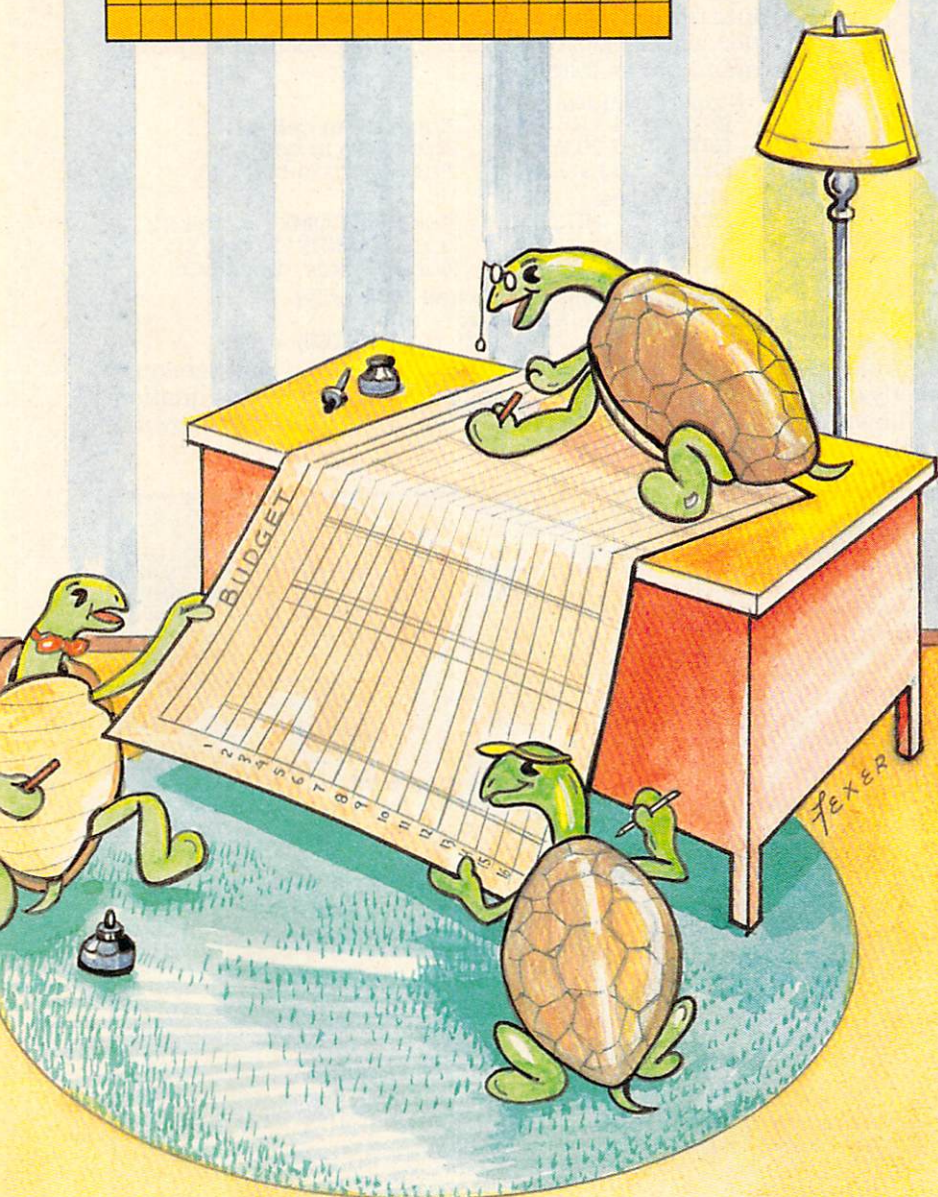
Although LOGO is perhaps best known for its graphics capability, it has other powerful features that make it a good candidate for use in writing a spreadsheet program. For example, LOGO variables can be numbers, strings of characters, or lists of values. What's more, variables are not restricted to one kind of value as they are in BASIC, where a variable must be either numeric or string. Because any cell in the spreadsheet can contain a number, a label, or a formula, the LOGO variable is ideal for representing the spreadsheet cells.

The TI Exception

The TI-99/4A home computer can't erase a variable from memory once it has been created, as the other machine implementations of LOGO can. Thus, it is impossible to store information in the spreadsheet by using variables to represent the cells. TI LOGO would never be able to distinguish between a value entered from an old spreadsheet or a new one. For this reason, a different algorithm was devised which places the values for the spreadsheet into a list. Each cell will be a list of two items in a greater list. The first item for a cell is the cell name, which will be used for searches. The second item in the list is the value for that cell. It could be a number, a string label, or another list (formula).

LOGO Runs Away

One of the most valuable aspects of writing a spreadsheet program in LOGO is the RUN command. This



command allows you to save a formula as a variable, and then RUN the contents of the variable. The output will be the result of the formula. Because of this feature, you will be able to enter a formula as the value of a cell, and have the result of the formula displayed on the spreadsheet. The following are samples of how procedures of a program execute equations from a list. This line can be used with the IBM LOGO, Apple LOGO II, and C-64 LOGO:

```
RUN FPUT "OUTPUT :CELL
```

This line can be used with the TI LOGO:

```
RUN (SE [ MAKE FIRST [ Y ] ] :EXPR )
```

Screen And Cell Size

The first step in writing a program like *LOGO Spreadsheet* is to develop a clear picture of how the end result should look from the user's point of view. Because of the difficulty in extending the spreadsheet beyond the borders of the screen, we decided to limit the workspace to five columns and eighteen rows, which should be adequate for simple operations. However, the cell size does vary slightly between systems:

C-64 LOGO	Seven (7) characters, or integer numbers.
Apple LOGO II	Six (6) characters, or integer numbers.
IBM LOGO	Six (6) characters, or integer numbers.
TI LOGO	Five (5) characters, or integer numbers.

The length of a formula is limited only by your system's maximum length for a string. But, a formula's result should never be a value that will exceed the space provided for it in a cell. Depending on your system, the result may be truncated, or an error message may appear in the cell.

Data Entry

Data entry takes two forms, depending on the system you are using. Both TI and C-64 LOGOs allow you to display graphics in the top two-thirds of the screen, and text in the bottom one-third. This allows the bottom lines to scroll without affecting the top of the screen where the spreadsheet is displayed. The TI and C-64 LOGOs also allow you to enter commands directly under the LOGO editor. The IBM and Apple LOGOs however, do not allow this to happen quite so easily, so the entire

screen is left in text mode. All data entry is done interactively with a procedure called **GETINPUT**.

To start the program, enter **SPREADSHEET**. You will then be asked if you want an old or a new spreadsheet. If you select **OLD**, the values assigned to the variables the last time you saved the file will be used. If you select **NEW**, the spreadsheet will be displayed with empty cells. At any time you can clear the contents of the spreadsheet by entering the command **NEW**. This erases all values stored in the spreadsheet—if you haven't saved them to disk before using the **NEW** command, they will be lost. On the IBM and Apple you can bypass the **SPREADSHEET** procedure when you want to use old data and simply enter **OLD**.

Spreadsheet Commands

A number of commands can be used to interact with the spreadsheet. You can enter numeric values (numbers), labels (words), or equations (lists) into any cell of the spreadsheet. You can then calculate one equation, or the whole spreadsheet. A detailed description of each of these commands is given below.

—ENTER :cell-name :cell-value

With this one command you can enter either numbers, labels, or equations into a cell, and the value will be displayed at the same time. If the value you enter into the cell is an equation, then the result of the equation will be displayed. Some examples of the **ENTER** command are as follows:

Numeric Values:

ENTER "A1 110	Place 110 in cell A1.
ENTER "A2 2036	Place 2036 in cell A2.
ENTER "D13 98375	Place 98375 in cell D13.

Label values:

ENTER "B1 "INCOME"	Place "INCOME" in cell B1.
ENTER "E16 "TAXES"	Place "TAXES" in cell E16.
ENTER "C8 "Totals"	Place "Totals" in cell C8.

Equations: (IBM, Apple, and C-64 only)

ENTER "A12 [:A1 + :D13 - :A2]
ENTER "E14 [(:A1 - :A2) * (:D13 / 2000) + 1]

It is possible to use an equation within another equation. For example, if cell A12 contained the equation shown above, you could then use A12 in the following equation:



LOGO Times is an information resource for users who want to create their own personal languages—languages that will easily allow them to communicate with the computer in a

totally new audiovisual realm of applied imagination, exploration, and self-discovery. The articles on these pages concern the use of the LOGO language, but readers do not need any additional software or equipment (or even a computer) to understand and learn from the material presented here.

If readers want to actually experience a LOGO environment, they will need a computer, the requisite software and/or cartridges, and any additional hardware required for a particular implementation. A disk drive is required for some LOGO implementations, but in other cases, a user's work may be saved on cassette tape, or copied into a notebook (for later re-keyboarding).

The varieties of LOGO we'll consider include—but are not limited to—Terrapin LOGO for the Apple II, II+ or IIe and the Commodore 64, TI LOGO for the TI-99/4A, and LOGO Computer Systems LOGO for the IBM PC and PCjr.

- **Apple:** Terrapin LOGO requires an Apple II, II+ or IIe with 64K of RAM, one disk drive with controller, and a blank, initialized disk.
- **Commodore 64:** Terrapin LOGO requires a Commodore 64 with a VIC-1541 Disk Drive and a blank, initialized disk.
- **TI-99/4A:** TI LOGO requires the TI LOGO or TI LOGO II cartridge and a compatible 32K memory expansion unit. A cassette recorder may be used for storage, but a compatible disk system is recommended for convenience.
- **IBM PC or PCjr:** LOGO Computer Systems LOGO requires the PC or PCjr with 128K bytes of RAM, one disk drive, and a blank, initialized disk.

In each issue, one or more of the articles may refer to or build upon the topics discussed in a previous article. It is therefore recommended that for maximum benefit and understanding, new readers obtain the appropriate back issues of *Home Computer Magazine* containing *LOGO Times* articles.

LOGO Listings

As you enter LOGO statements, the last thing you do at the end of every statement is to press **ENTER** on the TI and IBM (the key with the \leftarrow symbol), or **RETURN** on the Commodore 64 and Apple. This signals

the system to begin a new line. In our typeset listings, single LOGO statements may carry over from one line to the next without ending. The end of a LOGO statement is marked with a curved arrow (\curvearrowright) to indicate that you press **ENTER** or **RETURN** at that point.

Notice

LOGO Times is actively soliciting articles. Manuscripts should be typed double-spaced, and accompanied by a cassette tape or disk if containing any lengthy procedures or graphics.

Send all materials to:

LOGO Times Editorial Dept.
Home Computer Magazine
1500 Valley River Dr., Suite 250
Eugene, OR 97401

All mail directed to the Letters-to-the-Editor column (*Letters on LOGO*) will be published in accordance with the conditions set forth on *Home Computer Magazine's* Masthead page.

Our Contributing Editors

Henry Gorman, Jr.	Roger B. Kirchner
William M. Goodman	Rich Haller

• LOGO Times is a trademark of Emerald Valley Publishing Co. •

“Use caution though when creating equations within equations, for if a cell appears within its own equation, or within an equation it uses, you will have problems with recursion.”

ENTER [(RUN :A12) + :A2]

In this example, the result of the equation in cell A12 is added to the value in cell A2. Use caution though when creating equations within equations, for if a cell appears within its own equation, or within an equation it uses, you will have problems with recursion. Recursion occurs when one procedure keeps calling itself in an infinite loop. This usually results in an OUT OF SPACE error.

—Equations: (TI-99/4A)

TI LOGO handles equations a little differently because of the method it uses to store the cell values. This method of building equations may seem to have its drawbacks, but as you will see later, it also has many advantages. Here are some typical equations:

```
ENTER 'A12 [ (VAL [ A1 ]) + (VAL [ D13 ]) - (VAL [ A2 ]) ]  
ENTER 'C13 [ ((VAL [ A1 ]) - (VAL [ A2 ])) * ((VAL [ D13 ]  
/ 2000) + 1) ]
```

The VAL function returns the numeric value of the cell, whether it is a number or an equation. This is where the advantage comes in. In this version of *LOGO Spreadsheet* it makes no difference whether the cell is a number or an equation in extracting the cell's value. In the above example, A1 could have been either a number or an equation, and the equation using A1 would be the same.

UPDATE :cell-name

If you have a full spreadsheet, it could become quite time-consuming to recalculate every formula when only a few changes are needed. This command lets you pick and choose the formulas you want recalculated. For a cell to be recalculated, it should have a formula in it. To enter the cell desired, use the UPDATE command like this:

UPDATE "C12

This command lets you update only one cell at a time. In this example, the cell at column C row 12 will be updated.

RECALC

This command is useful if you wish to recalculate the entire spreadsheet. If any values have been entered or changed, then RECALC will display them.

OLD (IBM, and Apple only)

This command is available only with IBM LOGO and Apple LOGO II. It is necessary because in special cases, a bad command entry may cause a LOGO error message to be displayed. If this occurs, the program is no longer interactive (you are at the TOPLEVEL of LOGO). It's also possible that the error message will cause the screen to scroll, or even display the message right in the middle of the spreadsheet. If this happens, you can simply enter OLD and the spreadsheet will automatically be redisplayed. In addition, the RECALC procedure is automatically called at this time. If the error was caused by a bad cell formula, you should correct the problem before entering the OLD command. To correct a formula, you can use either the ENTER command as provided, or the MAKE primitive to reassign a value to a variable (the variable being the cell with the problem). If this isn't

satisfactory, you can erase a variable name by entering ERN and the variable name.

Complex Formulas

(Apple, IBM, and C-64 only)

Creating formulas can become a very complex task, even in simple spreadsheets. We added two procedures to help make the task easier with Apple LOGO II, IBM LOGO, and C-64 LOGO, and one procedure for the TI LOGO version.

CSUM :col-letter :beg-row-number :end-row-number

This command can be used to total up columns. You can specify the column you wish to total, the beginning row, and the ending row. A typical example might look like this:

ENTER "A18 [CSUM "A 1 9]

In this example the values in column A, rows 1 through 9 are totaled up and placed in cell A18. The cell receiving the total from the CSUM procedure does not have to be in the same column as the one on which the CSUM procedure is working. The same formula above could have been assigned to cell 'D7 just as easily. Notice that the column name is preceded by a quote. This is a requirement whenever LOGO works with words that are not numbers. The next two values are numbers, and do not require the single quote. There should also be a space between the column letter and the beginning row, and between the beginning and ending rows.

RSUM :row-number :beg-col-letter :end-col-letter

This procedure works the same as the CSUM procedure except that it provides row totals. A typical example looks like this:

ENTER "E18 [RSUM 1 "A "E]

The two procedures above can be mixed to create some very complex formulas. You also can directly insert normal cells with values, and integer numbers:

ENTER "E18 [(CSUM "A 1 18) + (RSUM 4 "A "C) * .E17 + 25]

Complex Formulas

(TI-99/4A only)

The TI LOGO version of *LOGO Spreadsheet* uses one procedure to replace the two procedures used in the other versions. The procedure ADDUP lets you obtain a total of all the cells in a rectangular area of the spreadsheet. You simply supply the upper-left cell and the lower-right cell in a list, and the sum of all cells between will result. An example would be:

ENTER "E18 [ADDUP [A1 C3]]

In this example, the value from cells A1 through A3, B1 through B3, and C1 through C3 were all added together and placed in cell 'E18. Notice that the colon (:) is not required on the cell names inside the formula. This is because they are not variable names. This procedure can also be used in complex formulas of the type:

ENTER "E18 [(ADDUP [A1 C3]) + (VAL [D4]) * 4]

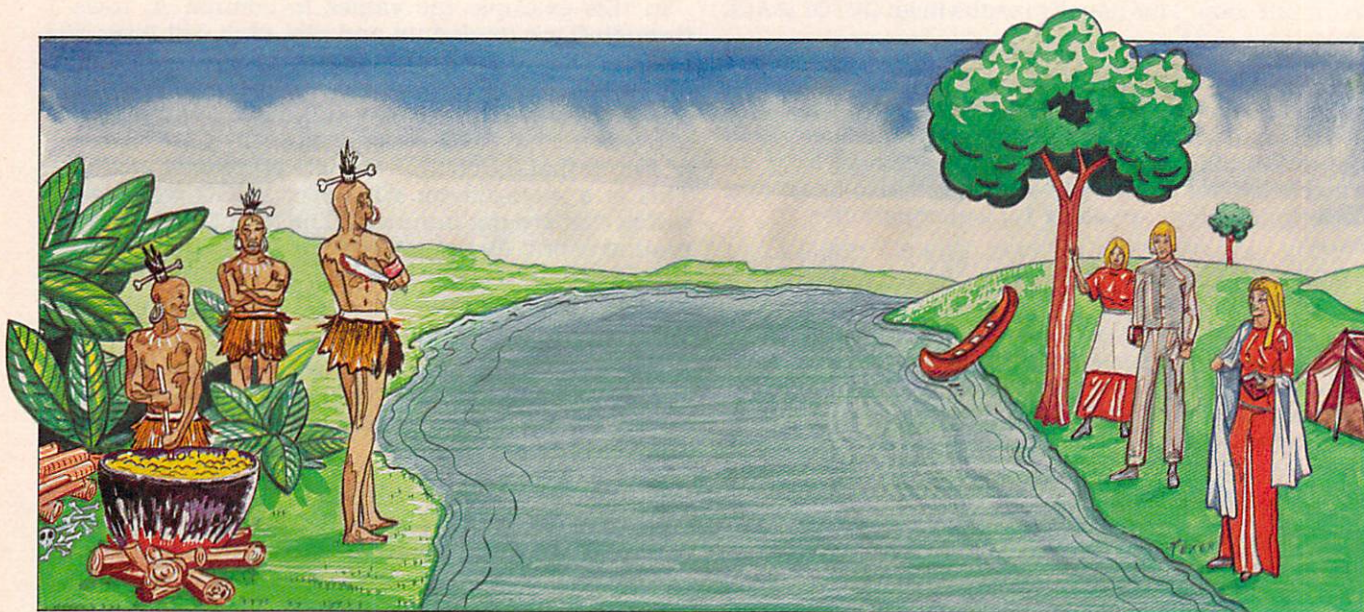
Expanding The Program

You may easily expand on any of the procedures presented here, or even add your own. The beautiful thing about LOGO is that you can create your own language. Make your commands in the form of procedures, to assist with shortcuts in your own personalized spreadsheet. You may also want to define a procedure which initializes some of the cells to a predesigned set of formulas and labels. The limit to this program's expansion and capabilities is only your imagination.

HCM



MISSIONARY IMPOSSIBLE



*Crossing the river?
Beware the cannibals—they'd love
to have you drop in for lunch!
An old puzzle finds a new form in LOGO.*

by Roger Kirchner
and the HCM Staff

One of the central problems in artificial intelligence research is the difficulty of devising efficient algorithms for "intelligent" search. By this we mean the ability to assess a problem, test the consequences of specific actions, and search for a legal solution. A good algorithm would not only find a *legal* solution, but would also search for the *best* solution. A classical example is the old "Missionaries and Cannibals" puzzle:

Three missionaries and three cannibals come to a river. There a boat transports people back and forth across the river, but it can hold a maximum of only two people at a time. The problem is to get everyone across safely. Everyone is "safe" as long as the cannibals never outnumber the missionaries on either shore. There can be cannibals on a shore with no missionaries.

In order to consider a problem like this, it is necessary to represent each possible state of the game, determine the

rules which apply in each state, and devise procedures to control the order in which rules are applied in transforming a start state into a goal state.

Designing The Solution

Several logical problems must be solved before this program can be written:

1. Decide how to represent the state of the puzzle—who is where, and where is the boat?
2. For a given state, determine what the legal boat loads are.
3. Make it possible for a user to play with the puzzle. The user must be able to enter the number of missionaries and cannibals to go in the boat. Play should continue until the puzzle is solved, and the program recognizes a solution.
4. Provide an automatic puzzle solution. For each state, the program should be able to calculate a list of possible moves and then try them out according to the rules. One of four things can happen at each stage:

- We arrive at a certain state for a second time. If this happens then we should back up and try again.
- We arrive at our goal state, at which time we will have solved the puzzle.
- We have tried all possibilities without success, in which case there would be no solution.
- We arrive at a new state. In this case we might be on a possible solution track. We compute the list of boat loads possible from this state, and try the first one in the list.

Representing A State

Many possibilities exist for representing the state of the puzzle. Obviously, we could represent a state with a list. But exactly how? The following method was chosen.

```
[ S [ m1 c1 ] [ m2 c2 ] ]
```

Here, **S** stands for the side of the river where the boat is positioned. The symbols **m1** and **c1** represent the number of missionaries and cannibals on the same side as the boat. The number of missionaries and cannibals on the opposite side of the river from the boat are represented by **m2** and **c2**. For example, `[L [3 3] [0 0]]` represents the start position. The boat is on the left shore (**L**), and there are three missionaries and three cannibals on that shore. No one is on the opposite shore (the right side).

The boat loads can also be represented by a list of two values: `[m c]`. The number of missionaries in the boat is **m**, while the number of cannibals is **c**.

"A good algorithm would not only find a legal solution, but would also search for the best solution."

Playing With The Puzzle

To play the puzzle, type **MISSCAN** and press `[RETURN]` or `[ENTER]`. You will be asked if you would like to solve the puzzle yourself. If you want to solve the puzzle, respond by entering **Y** for yes.

The **PLAY** procedure will then initialize the game. **"D** (for Description) is initialized to the start position. **"D** will contain the current state of the game. **PLAY** then calls **PLAY1**, which does all of the work—including prompting you for the number of missionaries and cannibals to send across in the boat.

The variable **"BTLD** contains the contents of the boat. The **MOVE** procedure is then called using **"D** and **"BTLD**. The **MOVE** procedure will output the new state if the move is legal (safe), or it will output `[]` if the move is not safe. **PLAY1** will be recursive until you enter a legal boat load. Once a legal boatload has been entered, the program will call the **ACTION** procedure to actually move from one shore to the other.

Automatically Searching For A Solution

After entering **MISSCAN** to start the puzzle, you will be asked if you would like to solve the puzzle yourself. If you want the computer to solve it for you select **N**.

The key to this portion of the program is the **BACKTRACK1** procedure. This is the procedure which will find the solution, and then remember the shortest path from the starting position to the current position.

Its purpose is to search for legal paths from one position in the puzzle to another. The procedure is recursive, calling itself until it locates the proper path to the end of the puzzle.

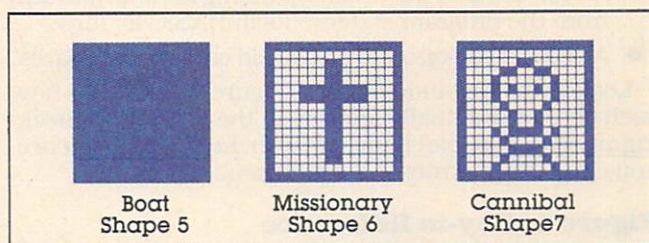
During the **BACKTRACK1** procedure, the results of the search are displayed on the screen. The Commodore, IBM, and Apple versions display only the contents of the boat during the search. The TI version actually goes through the whole sequence of moving the boat back and forth while it searches for a solution.

After a solution is found, control is given to **DO**, which displays each step in the solution by moving the missionaries and cannibals back and forth in the boat. After seeing the solution displayed, you can view it again or return to the main menu.

Graphics

—TI LOGO II

TI LOGO II is a fantastic LOGO for demonstrating the missionaries and cannibals puzzle because of its sprites. Sprites are ideal for depicting both the people and the boat. We merely put six sprites on each side of the screen and assign each sprite to carry one of three shapes: a missionary, a cannibal, or a blank for neither. The boat is made up of four sprites—two sprites on top of two others. We then assign the two sprites on top to carry one of the three shapes. The bottom two sprites make up the boat itself. The shapes for the cannibals and missionaries are shown here. They can be entered using the **MAKESHape** command.



—Commodore LOGO

Commodore LOGO uses sprites to place the missionaries, cannibals, and boat on the screen. Three sprites are used for the missionaries, three for the cannibals, and two for the boat. To move around the screen, the sprites' **X** and **Y** coordinates are changed. Three shapes are used, and must be loaded from the LOGO utility disk, which comes with the LOGO language disk. After loading the program and before running it, insert the utility disk. Type **MISSCAN** and press `[RETURN]`. The LOGO procedure **READSHAPES** will **BLOAD** the shapes into memory for you.

—IBM LOGO and APPLE LOGO II

The lack of sprites in both IBM and Apple LOGOs makes it necessary to use normal text characters for the missionaries, cannibals, and the boat. The missionaries are represented by a capital **M**, and the cannibals are a capital **C**. The boat is constructed to look like so:

```
{ _ }
```

To move the boat, it's necessary to locate the cursor with the **SETCURSOR** primitive, and then print the boat repeatedly with either a leading or trailing space, depending on whether it's traveling left or right across the screen.

HCM

For the Key-in listing see HCM PROGRAM LISTINGS Contents on page 93.

TI LOGO Versus TI LOGO II

The TI version of this procedure was written to work with **TI LOGO II** not **TI LOGO**. There aren't enough **NODES** available to enter all of the procedures with **TI LOGO**. If you do not have **TI LOGO II** you can at least play the game by entering the following procedures:





TI LOGO Procedures

TO SAVE? :P	TO DISPLAY :N :S :T
TO ADD :P :Q	TO SWITCH :SIDE
TO SUB :P :Q	TO PLAY
TO LEGALS :LOADS :D	TO ACTION :POS
TO INITVALS	TO MOVE :LOAD :SIDE :P :Q
TO MISSCAN	TO PLAY1
TO SPLIT :N :S1 :S2	TO NONO
TO MAP :X :XYS	TO SHOUT
TO UNLOADBOAT	TO INIT
TO MOVEBOAT :S	TO SETBOAT :S
TO LOADBOAT :LOAD	TO SHOW :D

HOME COMPUTER™

PROGRAM LISTINGS

CONTENTS

	 Page No.	 Page No.	 Page No.	 Page No.
LOGO SPREADSHEET	94	94	95	96
MISSIONARY IMPOSSIBLE	97	98	100	101
BOOLEAN BRAIN		104		106
ELEMENTARY ADD AND SUBTRACT	102		103	
MARKET MADNESS	108	110	113	116
STADIUM JUMPING	117	119	121	123
TAX DEDUCTION FILER	125	126	128	129

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.



Please include your subscriber number found at the top of your address label.

	Subscriber Price	Non-Subscriber Price
10 Diskettes 5¼" certified single-sided, double density with reinforced hub rings. Bulk-packaged 10 to a set with separate white envelopes and identification labels.	\$19.95 plus \$ 1.75 shipping	\$29.95 plus \$ 1.75 shipping
12 Cassette Tapes C-20 digital computer cassettes (nominally 10 minutes per side) with 5 screw housing for data integrity.	\$14.95 plus \$ 1.75 shipping	\$24.95 plus \$ 1.75 shipping

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.

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 _____

MUST BE IN US FUNDS DRAWN ON US BANK

☐ Check or Money Order Enclosed

Bill my ☐ VISA ☐ MASTERCARD

Account Number _____

Signature _____ Exp. Date _____

Home Computer Magazine
P.O. Box 5537 • Eugene, OR 97405

For information on ordering TOLL FREE see bind-in card located at the middle of this magazine.

Please clip or photocopy coupon

PROGRAM LISTING

LOGO SPREADSHEET (LOGO)

APPLE II Family

```

TO MAKE ROWS COL ROW + 1 CHAR C
MAKE "ROW CELL [DISPLAY
OL NAME CELL] [DISPLAY
IF THING END
END
TO LOCAL SUM COL MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END

```

```

TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END

```

```

TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END
TO LOCAL SUM ROW MAKE BEG END
LOCAL "N MAKE "N SUM 0
LOCAL "I MAKE "I BEG -
REPEAT 1 COL MAKE "I (WOR
D + :CELL SUM
:CELL PUT
END

```

HCM

LOGO SPREADSHEET (LOGO)

COMMODORE 64

```

TO SPREADSHEET
TO OUTLINE
END
TO MARGINS MAKE "I 64
LOCAL "TOP MAKE "TOP
LOCAL "3BL MAKE "3BL
OPTION "STAMP CHAR 2 1
PU HT SET HEADING "I 90
REPEAT 5 (MAKE "I TOP
MAKE CHAR 19 TO PP SET
SET STAMP 119 TO PP SET
SET STAMP 108 TO PP SET
THEADING "I 180
MAKE REPEAT 9 (MAKE "I
FD STAMP 417 (WORD SETX -144
REPEAT 9 (MAKE "I
TX -144
OPTION "STAMP CHAR 2 0
END
TO FORMAT :ITEM
IF NUMBER? :ITEM THEN
JUSTIFY :ITEM ELSE CENT
ER END

```

```

TO JUSTIFY :ITEM
LOCAL "I MAKE "I 6 - C
COUNT :ITEM
REPEAT :I [STAMP " ' ' ]
STAMP :ITEM
END
TO DISPLAY XCO :CELL
LOCAL "YCO 56
MAKE "YCO 120 - :ROW *
10 417
PU SETX XCO SETY YCO
STAMP CHAR SET SHIF
IF WORD? :CELL FORMAT FORMU
LA BUCKET :XCO :YCO
END
TO UPDATE R LOCAL C
MAKE "C FIRST :CELL
MAKE "R BUT FIRST :CELL
IF ( ANY OF NOT MEMBER?
: C ) ( ABCDE PR : < :R >
18 OF ) RANGE : PR : CELL ELSE
SHOW : CELL : R : C
END
TO OUTLINE
TO GRID
END

```

```

TO DUMP PROCES : EDIT ION
SAVE WORDS PROEC OUTL
(ION SPREADSHEET RECALC
INE MARGINS GRID SHOW STAM
ENTER UPDATE SHOW DISPLAY B
PUCKET FORMATS JUSTIFY CE
VALUE FORMULA RSUM CSUM
JEND INITEX1 DUMPPROCS
END
TO GRID
HT SETX 72
REPEAT 5 FD SETY 112 SET
H 180 FD 5613 SETX 127 S
90 SETX 108 FD 1279 PU
ETH 180 FD 10 417
END
TO ENTER R :CELL :VALUE
MAKE "C FIRST :CELL
MAKE "R FIRST :CELL
IF ( ANY OF NOT MEMBER?
: C ) ( ABCDE PR : < :R >
18 OF ) RANGE : PR : CELL ELSE
MAKE :CELL :VALUE SHOW
:CELL :R :C
END

```

Continued

[illegible][illegible][illegible]

HCM

LOGO SPREADSHEET (LOGO)

IBM PC & PCjr

[illegible][illegible][illegible]

Continued

Continued

[illegible]

```

TO LOCAL ROWS : CELL : COL : R : C :
MAKE "ROW : ROW + 1 :
MAKE "CELL WORD CHAR : C
OL : ROW :
IF NAMEP : CELL [DISPLAY
THING : CELL] [DISPLAY
" ]
END :

TO SHOWV : CELL : R : C :
IF NOT NAMEP : CELL [STO
PL
MAKE "ROW : R :
MAKE "COL (ASCII : C) -
64 :
DISPLAY THING : CELL :
END :

```

HCM

TI-99/4A

TI-99/4A

TO	BLANKS	:	ITEM	COL	:	IT
BLANKS	1	:	* (
END	ROW	:	ITEM			

TOP	COL	:	CELL	FIRST	:	CEL
OP	(C	HARNUM			
L)	-	64			
END)					

[illegible]

TO	COL	HEAD	:	KEN	STOP				
IF	HE	:	:	1	:				
CO	AD	:	5	*	K	+	2	0	
LT	+	:							

[illegible]

TO	IF	ROW	HEAD	:	THEN	STOP
IF	ROW	HEAD	:	:	1	
END	NUM	:	N	2	:	N

[illegible]

```

END
TO RECALC1 :SS
1 IF :SS = [ ] THEN FIRST STOP
TEST WORD? LAST FIRST :

```

[illegible][illegible]

IFT	SHOW	FIRST	:SS	STOP
MAKE	"SS	BF	:SS	
GEND	1			

TO	ENTER	CELL	:	CELL	:	ITEM	
TEST	PRINT	? [CELL]	OUT		
IF	STOP				OF		
RANGE							

E	N	D	S	-	-	-	-	-	-	-	-	-	-	-	-
T	O	S	I	S	:	W	:	C	R	S	:W:	C	:	R	:
T	E	S	T	P	=	F	I	R	S	T	:W:	C	:	R	:
I	F	S	T	O	P	E	C	H	A	R	N	U	M	:	R

[illegible][illegible][illegible][illegible][illegible][illegible]

MAKE	17	OKR	BOTH	:R	>	0	:
MAKE	17	OKC	BOTH	:C	>	0	:
MAKE	6	OKR	BOTH	:OKC	>	0	:

TEST	ST	GREATER	CHARNUM	(
LAST	OP	CELL	64	
LIFT	(OP		
OP	(CHARNUM	(LAST
CELL)	-	48	+
				10
				*

TO	U	P	D	A	T	E	:	C	E	L	L	:	S	H	E	E	T	:
U	P	D	A	T	E	1	:	C	E	L	L	:	S	H	E	E	T	:
E	N	D	:															

Continued

Continued

[illegible]

APPLE II Family

[illegible]

97

PROGRAM LISTING

MISSIONARY IMPOSSIBLE? (LOGO) *Continued*

IBM PC & PCjr

```
TO MAKE BOAT WORD : BOAT C
MAKE LOCAL REPEAT 1 TYPE
END

MOVE BOAT WORD : BOAT C
MAKE LOCAL REPEAT 1 TYPE
END

ELEFT BOAT WORD : BOAT C
MAKE LOCAL REPEAT 1 TYPE
END

LEFT BOAT WORD : BOAT C
MAKE LOCAL REPEAT 1 TYPE
END

WORD : BOAT C
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO MAKE BOAT WORD CHAR 32
MAKE LOCAL REPEAT 1 TYPE
END

MOVE BOAT WORD CHAR 32
MAKE LOCAL REPEAT 1 TYPE
END

ELEFT BOAT WORD CHAR 32
MAKE LOCAL REPEAT 1 TYPE
END

LEFT BOAT WORD CHAR 32
MAKE LOCAL REPEAT 1 TYPE
END

WORD CHAR 32
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO SET CURSOR 255
MAKE LOCAL REPEAT 1 TYPE
END

NONCURSOR 255
MAKE LOCAL REPEAT 1 TYPE
END

CURSOR 255
MAKE LOCAL REPEAT 1 TYPE
END

SET CURSOR 255
MAKE LOCAL REPEAT 1 TYPE
END
```

HCM

MISSIONARY IMPOSSIBLE? (LOGO)

TI-99/4A

```
TO MISSCAN
MAKE LOCAL REPEAT 1 TYPE
END

MISSCAN
MAKE LOCAL REPEAT 1 TYPE
END

CAN
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO INVALSY 97
MAKE LOCAL REPEAT 1 TYPE
END

INVALSY 97
MAKE LOCAL REPEAT 1 TYPE
END

VALSY 97
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO BACKTRACK 1 : DATALIST
MAKE LOCAL REPEAT 1 TYPE
END

BACKTRACK 1 : DATALIST
MAKE LOCAL REPEAT 1 TYPE
END

TRACK 1 : DATALIST
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO TELL IN ALL CARRY 8 SC : B
MAKE LOCAL REPEAT 1 TYPE
END

TELL IN ALL CARRY 8 SC : B
MAKE LOCAL REPEAT 1 TYPE
END

IN ALL CARRY 8 SC : B
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO HOW FIRST : D
MAKE LOCAL REPEAT 1 TYPE
END

HOW FIRST : D
MAKE LOCAL REPEAT 1 TYPE
END

FIRST : D
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO LEGALS : LOADS : D
MAKE LOCAL REPEAT 1 TYPE
END

LEGAL : LOADS : D
MAKE LOCAL REPEAT 1 TYPE
END

LOADS : D
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO ACTION : POS
MAKE LOCAL REPEAT 1 TYPE
END

ACTION : POS
MAKE LOCAL REPEAT 1 TYPE
END

POS
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO PLAY1
MAKE LOCAL REPEAT 1 TYPE
END

PLAY1
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO SPLIT : N : S1 : S2
MAKE LOCAL REPEAT 1 TYPE
END

SPLIT : N : S1 : S2
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO LOAD BOAT : LOAD : CODE
MAKE LOCAL REPEAT 1 TYPE
END

LOAD BOAT : LOAD : CODE
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO SAFE P1 : P2
MAKE LOCAL REPEAT 1 TYPE
END

SAFE P1 : P2
MAKE LOCAL REPEAT 1 TYPE
END
```

```
TO MOVE : LOAD : SIDE : P
MAKE LOCAL REPEAT 1 TYPE
END

MOVE : LOAD : SIDE : P
MAKE LOCAL REPEAT 1 TYPE
END
```

Continued

PROGRAM LISTING


```

TO TEST SET BOAT = "A" : S : DIST : BX + :
IFF MAKE "A" : S : SXY : BX + 1
TELL BY 1 : A : BY : SXY : BX + 1
A TELL 2 : A : BY : SXY : BX + 1
6 END

TO IF ADD = P : Q THEN OUTPUT
IF INPUT : SE SUM BF : P BF :
OUT FIRST : Q ADD BF : P BF :
Q END

TO IF SUB = P : Q THEN OUTPUT
IF INPUT : SE DIFFERENCE BF :
OUT FIRST : Q SUB BF :
P BF :
END

TO APPR RULE : DATA : LOADS : D
ATA PUT LEGALS :
END

TO TERM : IS : D : GOAL
OUT PUT :
END

TO MAPS : X : Y : THEN OUTP
IF MAKE [X] : Y : SE FIRST FIRST : X
UT MAKE [X] : Y : SE FIRST FIRST : X
YS : XYS FIRST BF FIRST : X
IF SE LAST = X : Y : THEN OUTP
LAST FIRST : X : Y :
OUT PUT MAP : X : Y :
END

TO MEMBER? : ITEM : LIST OUT
IF LIST = [ ] THEN
PUT "FALSE" : FIRST : LIST
IF ITEM = FIRST : LIST
THEN PUT MEMBER? : ITEM BF
OUT PUT MEMBER? : ITEM BF
END

TO CLEANUP : RDATALIST [ ] :
MAKE "PATH [ ] :
MAKE "PATH [ ] :
END

TO SWITCH : SIDE : THEN OUTP
IF SIDE = "L" THEN
UT ELSE OUTPUT "L"
END

TO UNLOAD BOAT :
TELL [1 2] CARRY 8 :
END

DO LIST [ ] THEN STO
PACT ON FIRST : LIST :
DO BF : LIST :
END

MOVE BOAT : S :
TELL : BOAT : R :
TEST : S :
IFF SH 90 :
REPEAT : DIST [FD 1] :
END

PF FIRSTS : LISTS :
IF LISTS = [ ] THEN ST
OPRINT FIRST FIRST : LIST
S PFIRSTS BF : LISTS :
END

TO DEADEND? : D :
OUT PUT "FALSE" :
END

TO UNLOAD BOAT :
TELL [1 2] CARRY 8 :
END

```

HCM

BASIC ADDITION AND SUBTRACTION

APPLE II Family

```

100 REM *** ADD SUBTRACT PROGRAM ***
110 REM *****
120 REM *****
130 REM *****
140 REM BY MARK DEWESE
150 REM AND THE HCM STAFF
160 REM HOME COMPUTER MAGAZINE
170 REM VERSION 4.4.1
180 REM APPLE FAMILY II APPLESOFT
190 GOSUB 610
200 GOSUB 470
210 GOSUB 520 : GOSUB 570
220 HOME : VTAB 22 : HTAB 2 : PRINT "PRES
230 S < M > : TO RETURN TO MENU"
240 GOSUB 720 : GOSUB 410
250 IF OV = 1 THEN N1 = S1 : N2 = S2 : N3 =
260 L : GOTO 240
270 N1 = L : N2 = S2 : N3 = S1
280 IF OV = 3 THEN 270
290 SY = 8 : GOSUB 440 : SX = 3 : NM = N1 : G
300 OSUB 670 : COLOR = 0 : ON OV GOSUB 730
310 750 : SX = 17 : NM = N2 : GOSUB 440 : G
320 SUB 670 : COLOR = 0 : GOSUB 770
330 IF OV = 1 THEN SX = 31 : NM = N3 : GOS
340 UB 440 : GOSUB 670
350 REM DISPLAY NUMBER PROBLEM
360 SY = 24 : SX = 3 : NM = N1 : GOSUB 440 :
370 GOSUB 460 : COLOR = 0 : ON OV GOSUB 73
380 750 : SX = 17 : NM = N2 : GOSUB 440 : G
390 OSUB 460 : COLOR = 0 : GOSUB 770
400 POKE 16368,0 : GET NS : IF (NS < "
410 "OR NS > "9") AND NS < "M" THE
420 N = 290
430 IF NS = "M" THEN VAL 190
440 GOSUB 440 : NM = VAL (NS) : GOSUB 440
450 IF SX = 31 : GOSUB 460
460 IF OV = 2 THEN SX = 31 : NM = VAL (N
470 S) : GOSUB 440 : GOSUB 670
480 IF VAL (NS) = N3 THEN 360
490 D = 4 : P = P(2) : GOSUB 700 : D = 30 : P
500 = P(8) : GOSUB 700 : IF OV = 2 THEN S
510 X = 31 : NM = VAL (NS) : COLOR = 15 : G
520 OSUB 670 : NM = VAL (NS) : GOSUB 460
530 GOTO 290
540 IF OV = 2 THEN SX = 31 : NM = VAL (N
550 S) : GOSUB 440 : GOSUB 670
560 GOSUB 440 : GOSUB 460 : FOR I = 1 TO
570 7 : STEP 2 : D = 4 : P = P(1) : GOSUB 700 :
580 GOSUB 460 : GOSUB 460 : IF I < 7 THE
590 N = 2 : P = P(I + 1) : GOSUB 700 : NE
600 XT DE = 1 TO 300 : NEXT DE
610 GOTO 210
620 FOR DE = 1 TO 1000 : NEXT DE : GOTO 2
630 10
640 REM MAKE UP PROBLEM
650 L = INT ( RND (1) * 9 ) + 1
660 S2 = INT ( RND (1) * 9 ) + 1 : S1 = L
670 S2 = S2 : RETURN
680 REM SELECT RANDOM COLOR
690 COLOR = INT ( RND (1) * 14 ) + 1 : RE
700 TURN
710 ON NM + 1 GOSUB 790,810,830,850,870
720 890,910,930,950,970 : RETURN
730 REM
740 TITLE SCREEN

```

Continued


```

810 REM DISPLAY 1
820 VLIN 24,30 AT SX + 2: RETURN
830 REM DISPLAY 2
840 HLIN 24,30 AT SX + 2: RETURN
850 REM DISPLAY 3
860 HLIN 24,30 AT SX + 2: RETURN
870 REM DISPLAY 4
880 VLIN 24,30 AT SX + 2: RETURN
890 REM DISPLAY 5
900 HLIN 24,30 AT SX + 2: RETURN

```

```

910 REM DISPLAY 6
920 VLIN 24,30 AT SX + 1, SX + 4 AT 27: RETURN
930 REM DISPLAY 7
940 HLIN 24,30 AT SX + 1, SX + 4 AT 27: RETURN
950 REM DISPLAY 8
960 HLIN 24,30 AT SX + 1, SX + 4 AT 27: RETURN
970 REM DISPLAY 9
980 HLIN 24,30 AT SX + 1, SX + 4 AT 27: RETURN
990 DATA 143,124,105,85,74,52,24,8

```

HCM

BASIC ADDITION AND SUBTRACTION

IBM PC & PCjr

```

100 ** ADD SUBTRACT PROGRAM **
110 ** BY MARK DEWESE AND THE HCM STAFF **
120 ** HOME COMPUTER MAGAZINE **
130 ** VERSION 4.4.1 **
140 ** IBM PCjr WITH CASSETTE & CARTRIDGE **
150 ** BASIC **
160 ** IBM PC WITH CASSETTE BASIC & BASIC **
170 ** RANDOMIZE TIMER:DEFINT A-Z:SCREEN 0 **
180 ** WIDTH 40:COLOR 7,1 **
190 ** KEY OFF:CLS:LOCATE 11,13,0:PRINT "BASIC ADDITION" **
200 ** LOCATE 13,12:PRINT "AND SUBTRACTION" **
210 ** LOCATE 24,13:PRINT "PRESS ANY KEY": **
220 ** LOCATE 25,14:PRINT "TO CONTINUE": **
230 ** Z$="T2400314CDEFDECL2GO4CO3L16CBCBCL" **
240 ** FL4FL2G" **
250 ** MS=CHR$(219):MINUS$=M$+M$+M$ **
260 ** IF INKEY$="" THEN 250 **
270 ** CLS:X=0:LOCATE 9,14:PRINT "ENTER:" **
280 ** LOCATE 11,13:PRINT "1 FOR ADDITION": **
290 ** LOCATE 13,13:PRINT "2 FOR SUBTRACTION" **
300 ** GOSUB 580:IF A<>49 AND A<>50 THEN **
310 ** OV=A-48 **
320 ** CLS:LOCATE 6,12:PRINT "ENTER:" **
330 ** LOCATE 9,5:PRINT "1 GRAPHICS WITH GRAPHIC ANSWERS" **
340 ** LOCATE 12,5:PRINT "2 GRAPHICS WITHOUT GRAPHIC ANSWERS" **
350 ** LOCATE 15,5:PRINT "3 NUMBERS ONLY" **
360 ** GOSUB 580:IF A<49 OR A>51 THEN 340 **
370 ** GV=A-48 **
380 ** CLS:L=(RND*9) **
390 ** S2=(RND*L):S1=L-S2 **
400 ** IF OV=1 THEN LN=S1:RN=S2:N=L ELSE L **
410 ** N=L:RN=S2:N=S1 **
420 ** PN=LN:ROW=12:COL=5:GOSUB 1110:IF GV **
430 ** >2 THEN 410 **
440 ** NC=LN:AD=5:X1=8:GOSUB 1120 **
450 ** IF OV=2 THEN GOTO 440 ELSE GOSUB 11 **
460 ** 60:ROW=13:GOSUB 430:IF GV<3 THEN GO **
470 ** SUB 1160:ROW=6:GOSUB 430 **
480 ** GOTO 450 **
490 ** LOCATE ROW,14:PRINT M$:LOCATE ROW+1 **
500 ** ,13:PRINT MINUS$:LOCATE ROW+2,14:PR **
510 ** INT M$:RETURN **
520 ** GOSUB 1160:LOCATE 14,14:PRINT MINUS **
530 ** $:IF GV<3 THEN GOSUB 1160:LOCATE 6, **
540 ** 14:PRINT MINUS$ **
550 ** PN=RN:ROW=12:COL=19:GOSUB 1110:IF G **
560 ** V>2 THEN 470 **
570 ** NC=RN:AD=5:X1=22:GOSUB 1120 **
580 ** GOSUB 1160:LOCATE 13,26:PRINT MINUS **
590 ** $:LOCATE 15,26:PRINT MINUS$:IF GV<3 **
600 ** THEN GOSUB 1160:LOCATE 7,26:PRINT **
610 ** MINUS$:LOCATE 9,26:PRINT MINUS$ **
620 ** IF GV=1 THEN AD=5:X1=37:NC=N:GOSUB **
630 ** 1130 **
640 ** LOCATE 23,12:COLOR 7,1:PRINT "PRESS **
650 ** M$ FOR MENU":GOSUB 580:IF A=77 OR A **
660 ** =109 THEN 260 ELSE IF A<48 OR A>57 **
670 ** THEN 490 **
680 ** GOSUB 560:IF SN<>N THEN 540 **
690 ** IF LEN(R$)>9 OR R$="" THEN R$="O2C" **
700 ** :R=71 **
710 ** RS=RS+CHR$(R):PLAY "MB T240:XRS$":R **
720 ** =R-1:Z$=" " **
730 ** FOR I=1 TO 2400:NEXT X=0:GOTO 360 **
740 ** Z$="O2C":PLAY "MB T240:XZ$":R$=" " **
750 ** FOR J=0 TO 4:LOCATE 12+J,33:PRINT " **
760 ** :NEXT J:IF GV=1 THEN 480 ELSE " **
770 ** FOR J=0 TO 4:LOCATE 6+J,33:PRINT " **
780 ** :NEXT J:GOTO 480 **

```

```

560 SN=A-48:ROW=12:COL=33:PN=SN:GOSUB 1
570 110:IF GV<>2 THEN RETURN
580 AD=5:NC=SN:X1=37:GOSUB 1130:RETURN
590 AS=INKEY$:IF AS<>" " THEN 580
600 AS=INKEY$:IF AS=" " THEN 590
610 A=ASC(AS):RETURN
620 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '0
630 LOCATE ROW+1,COL:PRINT M$+" "+M$
640 LOCATE ROW+2,COL:PRINT M$+" "+M$
650 LOCATE ROW+3,COL:PRINT M$+" "+M$
660 LOCATE ROW+4,COL:PRINT M$+M$+M$+M$:
670 RETURN
680 LOCATE ROW,COL:PRINT " "+M$ '1
690 LOCATE ROW+1,COL:PRINT " "+M$
700 LOCATE ROW+2,COL:PRINT " "+M$
710 LOCATE ROW+3,COL:PRINT " "+M$
720 LOCATE ROW+4,COL:PRINT " "+M$:RETU
730 RN
740 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '2
750 LOCATE ROW+1,COL:PRINT " "+M$
760 LOCATE ROW+2,COL:PRINT " "+M$
770 LOCATE ROW+3,COL:PRINT " "+M$
780 LOCATE ROW+4,COL:PRINT " "+M$:RETU
790 RN
800 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '3
810 LOCATE ROW+1,COL:PRINT " "+M$
820 LOCATE ROW+2,COL:PRINT " "+M$
830 LOCATE ROW+3,COL:PRINT " "+M$
840 LOCATE ROW+4,COL:PRINT " "+M$:RETU
850 RN
860 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '5
870 LOCATE ROW+1,COL:PRINT M$
880 LOCATE ROW+2,COL:PRINT M$+M$+M$+M$
890 LOCATE ROW+3,COL:PRINT " "+M$
900 LOCATE ROW+4,COL:PRINT M$+M$+M$+M$:
910 RETURN
920 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '6
930 LOCATE ROW+1,COL:PRINT M$
940 LOCATE ROW+2,COL:PRINT M$+M$+M$+M$
950 LOCATE ROW+3,COL:PRINT M$+" "+M$
960 LOCATE ROW+4,COL:PRINT M$+M$+M$+M$:
970 RETURN
980 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '7
990 LOCATE ROW+1,COL+3:PRINT M$
1000 LOCATE ROW+2,COL+3:PRINT M$
1010 LOCATE ROW+3,COL+3:PRINT M$
1020 LOCATE ROW+4,COL+3:PRINT M$:RETURN
1030 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '8
1040 LOCATE ROW+1,COL:PRINT M$+" "+M$
1050 LOCATE ROW+2,COL:PRINT M$+M$+M$+M$
1060 LOCATE ROW+3,COL:PRINT M$+" "+M$
1070 LOCATE ROW+4,COL:PRINT M$+M$+M$+M$:
1080 RETURN
1090 LOCATE ROW,COL:PRINT M$+M$+M$+M$ '9
1100 LOCATE ROW+1,COL:PRINT M$+" "+M$
1110 LOCATE ROW+2,COL:PRINT M$+M$+M$+M$
1120 LOCATE ROW+3,COL:PRINT " "+M$
1130 LOCATE ROW+4,COL:PRINT " "+M$:RET
1140 URN
1150 GOSUB 1160:ON PN+1 GOTO 610,660,710
1160 760,810,860,910,960,1010,1060
1170 CH=END*2+1:CH=(CH=1)*-1+(CH=2)*-2+(
1180 CH=3)*-14
1190 IF NC=0 THEN RETURN
1200 FOR I=0 TO NC-1:IF I//3=INT(I//3)
1210 THEN AD=AD+1:X1=X1-3
1220 GOSUB 1160:LOCATE AD,X1:PRINT CHR$(
1230 CH):X1=X1+1:NEXT:RETURN
1240 CL=INT(RND*14)+2:COLOR CL,1:RETURN

```

HCM


```

100 REM *****
110 REM * BOOLEAN BRAIN *
120 REM *****
130 REM BY WILLIAM K. BALTHROP
140 REM HOME COMPUTER MAGAZINE
150 REM VERSION 4.4.1
160 REM C-64 BASIC
170 REM
180 POKE 53280,0:DIM RMS(10,2),GT(9,3),
  G(9)
190 FORZ=1TO10:READRMS(Z,1),RMS(Z,2):NE
  XT
200 TS="40CRSRRIGHT"
210 DEF FN RV(C)=VAL(MID$(RMS(R,2),C,1)
  )
220 R=1:DI=1:SC=0:DR=0:TS=0
230 POKE53281,6:PRINT"SHIFT CLR"CTRL
  GRN"CTRL RVSON"CMDCR"CTRL RVSON"
  F
240 PRINT"CTRL RVSON"SHIFT"CMDCR"CTRL RV
  SOFF
250 PRINT"CTRL RVSON"SHIFT"CMDCR"CTRL R
  VSOFF
260 PRINT"CTRL RVSON"SHIFT"CMDCR"CTRL
  RVSOFF
270 PRINT"CTRL RVSON"SHIFT"CMDCR"CTRL
  RVSOFF
280 PRINT"CTRL RVSON"SHIFT"CMDCR"CTR
  L RVSOFF
290 PRINT"CTRL RVSON"SHIFT"CMDCR"CT
  RL RVSOFF
300 PRINT"CTRL RVSON"SHIFT"CMDCR"CT
  RL RVSOFF
310 PRINT"CTRL RVSON"SHIFT"CTRL CYN
  GRN
320 PRINT"CTRL RVSON"CTRL CYN
  GRN
330 PRINT"CTRL RVSON"CTRL CYN
  GRN
340 PRINT"CTRL RVSON"CTRL CYN
  GRN
350 PRINT"CTRL RVSON"CTRL CYN
  GRN
360 PRINT"CTRL RVSON"CTRL CYN
  GRN
370 PRINT"CTRL RVSON"CTRL CYN
  GRN
380 PRINT"CTRL RVSON"CTRL CYN
  GRN
390 PRINT"CTRL RVSON"CTRL CYN
  GRN
400 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
410 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
420 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
430 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
440 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
450 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
460 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
470 PRINT"CTRL RVSON"CTRL RVSO
  FF"SHIFT"CMDCR"CTRL RVSON"
480 POKE 2023,95:POKE 56295,5
490 IFR=0 THEN 1980
500 ON DI GOSUB 610, 640, 680, 720
510 PRINT"HOME"23CRSRDOWN"CTRL RVSO
  FF"2CRSRRIGHT"
520 PRINT"HOME"23CRSRDOWN"CTRL WHT
  "CTRL RVSOFF":LEFT$(TS,(40-LEN(RMS
  (R,1)))/2):RMS(R,1):
530 GETAS:IFAS=" "THEN 530
540 IFAS="E" THEN DI=1:GOSUB 920::GOT
  O 590
550 IFAS="W" THEN DI=2:GOSUB 920::GOT
  O 590
560 IFAS="N" THEN DI=3:GOSUB 920::GOT
  O 590
570 IFAS="S" THEN DI=4:GOSUB 920::GOT
  O 590
580 GOTO 530

```

```

590 IFD=1THENGOTO 500
600 GOTO 230
610 ON FN RV(5)+1 GOSUB 760, 770::PRINT
  "HOME"10CRSRDOWN"24CRSRRIGHT"CT
  RL RVSON"CTRL CYN"
620 ON FN RV(7)+1 GOSUB 810, 820
630 ON FN RV(8)+1 GOSUB 870, 880::RETU
  RN
640 ON FN RV(6)+1 GOSUB 760, 770
650 PRINT"HOME"10CRSRDOWN"24CRSRRIG
  HT"CTRL RVSON"CTRL CYN"
660 ON FN RV(8)+1 GOSUB 810, 820
670 ON FN RV(7)+1 GOSUB 870, 880::RETU
  RN
680 ON FN RV(7)+1 GOSUB 760, 770
690 PRINT"HOME"10CRSRDOWN"24CRSRRIG
  HT"CTRL RVSON"CTRL CYN"
700 ON FN RV(6)+1 GOSUB 810, 820
710 ON FN RV(5)+1 GOSUB 870, 880::RETU
  RN
720 ON FN RV(8)+1 GOSUB 760, 770
730 PRINT"HOME"10CRSRDOWN"24CRSRRIG
  HT"CTRL RVSON"CTRL CYN"
740 ON FN RV(5)+1 GOSUB 810, 820
750 ON FN RV(6)+1 GOSUB 870, 880::RETU
  RN
760 PRINT"CTRL RED"::GOTO 780
770 PRINT"CTRL BLK"
780 PRINT"HOME"10CRSRDOWN"17CRSRRIG
  HT"
790 PRINT"CTRL RVSON"CRSRDOWN"6
  SHIFT CRSRLEFT"CRSRDOWN"6
  SHIFT CRSRRIGHT"CM
  DR+"CRSRDOWN"8SHIFT CRSRLEFT"
  CRSRDOWN"6SHIFT CRSRLEFT"
800 PRINT"CRSRDOWN"6SHIFT CRSRLE
  FT"CRSRDOWN"6SHIFT CRSRLEFT"
  "":RETURN
810 PRINT"CTRL RED"::GOTO 830
820 PRINT"CTRL BLK"
830 PRINT"HOME"10CRSRDOWN"CRSRRIGHT
  "
840 PRINT"CTRL RVSON"CRSRDOWN"4S
  HIFT CRSRLEFT"CRSRDOWN"4SHIFT
  CRSRLEFT"CRSRRRIGHT"CMDCR+"C
  RSRDOWN"6SHIFT CRSRLEFT"CRSR
  DOWN"4SHIFT CRSRLEFT"CRSRDOWN"
  4SHIFT CRSRLEFT"CRSRDOWN"4SH
  IFT CRSRLEFT"
850 PRINT"CRSRDOWN"4SHIFT CRSRLE
  FT"CRSRDOWN"4SHIFT CRSRLEFT"
  CRSRDOWN"4SHIFT CRSRLEFT"CTRL
  RVSOFF"SHIFT"CRSRDOWN"4SHIFT C
  RSRLEFT"CTRL RVSON"CTRL RVSOFF
  SHIFT"CRSRDOWN"3SHIFT CRSRLEFT
  CTRL RVSON"CTRL RVSOFF"SHIFT
  CRSRDOWN"2SHIFT CRSRLEFT"SHIFT
  "
860 RETURN
870 PRINT"CTRL RED"::GOTO 890
880 PRINT"CTRL BLK"
890 PRINT"HOME"10CRSRDOWN"35CRSRRIG
  HT"CTRL RVSON"
900 PRINT"CRSRDOWN"4SHIFT CRSRLE
  FT"CRSRDOWN"4SHIFT CRSRLEFT"
  CRSRDOWN"4SHIFT CRSRLEFT"CRSR
  DOWN"4SHIFT CRSRLEFT"CRSRDOWN"
  4SHIFT CRSRLEFT"CRSRDOWN"4S
  HIFT CRSRLEFT"
910 PRINT"CRSRDOWN"4SHIFT CRSRLE
  FT"CRSRDOWN"4SHIFT CRSRLEFT"
  CRSRDOWN"4SHIFT CRSRLEFT"CTRL
  RVSOFF"CMDCR"CTRL RVSON"CRS
  RDOWN"3SHIFT CRSRLEFT"CTRL RVSOFF
  CMDCR"CTRL RVSON"CRSRDOWN"2
  SHIFT CRSRLEFT"CTRL RVSOFF"CMDCR
  CTRL RVSON"CRSRDOWN"SHIFT CRSR
  LEFT"CTRL RVSOFF"CMDCR"::RETURN
920 D=1:TR=FN RV(DI):IFR=TRTHEN 2240
930 IF FN RV(DI+4)=0THEN 950
940 R=TR:RETURN
950 POKE 53280,0:POKE 53281,0:D=0:GOSUB
  2350
960 PRINT"SHIFT CLR"CTRL RED"CRSRDOW
  N"12CMDCR P"2CRSRDOWN"3SHIFT CRSR
  LEFT"22CMDCR Y"3CRSRDOWN"3SHIFT C
  RSRLEFT"32CMDCR P"2CRSRDOWN"3SHIF
  T CRSRLEFT"42CMDCR Y"3CRSRDOWN"3S
  HIFT CRSRLEFT"52CMDCR P"2CRSRDOWN"
  3SHIFT CRSRLEFT"62CMDCR Y"3CRSRDO
  WN"3SHIFT CRSRLEFT"72CMDCR P"2CRS
  RDOWN"3SHIFT CRSRLEFT"82CMDCR Y"
  PRINT"3CRSRDOWN"3SHIFT CRSRLEFT"9
  2CMDCR P"2CRSRDOWN"3SHIFT CRSRLEF
  T"02CMDCR Y"
980 FORZ=1TO9:GT(Z,1)=INT(RND(0)*2)+1:G
  T(Z,2)=0:GT(Z,3)=0:NEXT
990 IFGT(1,1)=1THENGOSUB 1800::GOTO 101
  0
1000 GOSUB 1620
1010 IFGT(2,1)=1THENGOSUB 1820::GOTO 103
  0
1020 GOSUB 1640
1030 IFGT(3,1)=1THENGOSUB 1840::GOTO 105
  0
1040 GOSUB 1660
1050 IFGT(4,1)=1THENGOSUB 1860::GOTO 107
  0

```

Continued


```

1060 GOSUB 1780
1070 IF GT(5,1)=1 THEN GOSUB 1880:GOTO 109
0
1080 GOSUB 1700
1090 IF GT(6,1)=1 THEN GOSUB 1900:GOTO 111
0
1100 GOSUB 1720
1110 IF GT(7,1)=1 THEN GOSUB 1920:GOTO 113
0
1120 GOSUB 1740
1130 IF GT(8,1)=1 THEN GOSUB 1940:GOTO 115
0
1140 GOSUB 1760
1150 IF GT(9,1)=1 THEN GOSUB 1960:GOTO 117
0
1160 GOSUB 1780
1170 GETAS:IFAS=" " THEN 1170:IFAS<"0" OR A
S>"9" THEN GOTO 1170
1180 GOSUB 2370
1190 SC=SC+1:ON VAL(AS)+1 GOTO 1290,1200,12
10,1220,1230,1240,1250,1260,1270,12
80
1200 PRINT "HOME":CTRL GRN:CRSRDOWN:1:2
CMRDR P":GT(1,2)=1:GOTO 1300
1210 PRINT "HOME":CTRL GRN:CRSRDOWN:12:
2CMRDR Y":GT(1,3)=1:GOTO 1300
1220 PRINT "HOME":CTRL GRN:CRSRDOWN:13:
2CMRDR P":GT(2,2)=1:GOTO 1330
1230 PRINT "HOME":CTRL GRN:CRSRDOWN:14:
2CMRDR Y":GT(2,3)=1:GOTO 1330
1240 PRINT "HOME":CTRL GRN:CRSRDOWN:15:
2CMRDR P":GT(3,2)=1:GOTO 1360
1250 PRINT "HOME":CTRL GRN:CRSRDOWN:16:
2CMRDR Y":GT(3,3)=1:GOTO 1360
1260 PRINT "HOME":CTRL GRN:CRSRDOWN:17:
2CMRDR P":GT(4,2)=1:GOTO 1390
1270 PRINT "HOME":CTRL GRN:CRSRDOWN:18:
2CMRDR Y":GT(4,3)=1:GOTO 1390
1280 PRINT "HOME":CTRL GRN:CRSRDOWN:19:
2CMRDR P":GT(5,2)=1:GOTO 1420
1290 PRINT "HOME":CTRL GRN:CRSRDOWN:20:
2CMRDR Y":GT(5,3)=1:GOTO 1420
1300 IF(GT(1,2)=0 OR GT(1,3)=0) AND GT(1,1)=
2 THEN GOTO 1170
1310 IF GT(1,1)=1 THEN GOSUB 1800:GT(6,2)=
1:GOTO 1450
1320 GOSUB 1620:GT(6,2)=1:GOTO 1450
1330 IF(GT(2,2)=0 OR GT(2,3)=0) AND GT(2,1)=
2 THEN GOTO 1170
1340 IF GT(2,1)=1 THEN GOSUB 1820:GT(6,3)=
1:GOTO 1450
1350 GOSUB 1640:GT(6,3)=1:GOTO 1450
1360 IF(GT(3,2)=0 OR GT(3,3)=0) AND GT(3,1)=
2 THEN GOTO 1170
1370 IF GT(3,1)=1 THEN GOSUB 1840:GT(7,2)=
1:GOTO 1480
1380 GOSUB 1660:GT(7,2)=1:GOTO 1480
1390 IF(GT(4,2)=0 OR GT(4,3)=0) AND GT(4,1)=
2 THEN GOTO 1170
1400 IF GT(4,1)=1 THEN GOSUB 1860:GT(7,3)=
1:GOTO 1480
1410 GOSUB 1680:GT(7,3)=1:GOTO 1480
1420 IF(GT(5,2)=0 OR GT(5,3)=0) AND GT(5,1)=
2 THEN GOTO 1170
1430 IF GT(5,1)=1 THEN GOSUB 1880:GT(8,3)=
1:GOTO 1510
1440 GOSUB 1700:GT(8,3)=1:GOTO 1510
1450 IF(GT(6,2)=0 OR GT(6,3)=0) AND GT(6,1)=
2 THEN GOTO 1170
1460 IF GT(6,1)=1 THEN GOSUB 1900:GT(9,2)=
1:GOTO 1540
1470 GOSUB 1720:GT(9,2)=1:GOTO 1540
1480 IF(GT(7,2)=0 OR GT(7,3)=0) AND GT(7,1)=
2 THEN GOTO 1170
1490 IF GT(7,1)=1 THEN GOSUB 1920:GT(8,2)=
1:GOTO 1510
1500 GOSUB 1740:GT(8,2)=1:GOTO 1510
1510 IF(GT(8,2)=0 OR GT(8,3)=0) AND GT(8,1)=
2 THEN GOTO 1170
1520 IF GT(8,1)=1 THEN GOSUB 1940:GT(9,3)=
1:GOTO 1540
1530 GOSUB 1760:GT(9,3)=1:GOTO 1540
1540 IF(GT(9,2)=0 OR GT(9,3)=0) AND GT(9,1)=
2 THEN GOTO 1170
1550 IF GT(9,1)=1 THEN GOSUB 1960:GOTO 157
0
1560 GOSUB 1780
1570 DR=DR+1:RMS(R,2)=LEFT$(RMS(R,2),3+D
I)+1:RIGHT$(RMS(R,2),4-DI)
1580 R=FN RV(DI):IF DI=1 OR DI=3 THEN AD=DI+1
:GOTO 1600
1590 AD=DI-1
1600 RMS(R,2)=LEFT$(RMS(R,2),3+AD)+1:R
IGHT$(RMS(R,2),4-AD):GOSUB 2390
1610 FOR ID=1 TO 2000:NEXT:RETURN
1620 PRINT "HOME":CRSRDOWN:3:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:4:
SHIFT CRSRLEFT:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:3CMRDR P:CRSRDOWN:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:SH
IFT L:CMRDR P"
1630 RETURN
1640 PRINT "HOME":6:CRSRDOWN:3:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:4:
SHIFT CRSRLEFT:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:3CMRDR P:CMRDR H:SH
IFT CRSRUP:SHIFT CRSRLEFT:SHIFT O
:CMRDR Y"
1650 RETURN

```

```

1660 PRINT "HOME":1:CRSRDOWN:3:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:3CMRDR P:CRSRDOWN:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:SH
IFT L:CMRDR P"
1670 RETURN
1680 PRINT "HOME":16:CRSRDOWN:3:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:3CMRDR P:CMRDR H:SH
IFT CRSRUP:SHIFT CRSRLEFT:SHIFT
O:CMRDR Y"
1690 RETURN
1700 PRINT "HOME":21:CRSRDOWN:3:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:10CMRDR P":
1710 PRINT "CMRDR H:SHIFT CRSRUP:SHIFT
CRSRLEFT:CMRDR H:SHIFT CRSRUP:SHI
FT CRSRLEFT:CMRDR H:SHIFT CRSRUP:SHI
FT CRSRLEFT:SHIFT O:2CMRDR Y":
1720 RETURN
1720 PRINT "HOME":4:CRSRDOWN:12:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:10CMRDR P:CRSRDOWN:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:
SHIFT L:2CMRDR P"
1730 RETURN
1740 PRINT "HOME":14:CRSRDOWN:12:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CMRDR:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:2CMRDR P:CRSRDOWN:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:
SHIFT L:CMRDR P"
1750 RETURN
1760 PRINT "HOME":17:CRSRDOWN:20:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CRSRDOWN:4:SHI
FT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:2CMRDR P:CRSRDOWN:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:
SHIFT L:CMRDR P"
1770 PRINT "SHIFT CRSRUP:2CMRDR P:CMRDR
H:SHIFT CRSRUP:SHIFT CRSRLEFT:CM
DR H:SHIFT CRSRUP:SHIFT CRSRLEFT:
CMRDR H:SHIFT CRSRUP:SHIFT CRSRLE
FT:CMRDR H:SHIFT CRSRUP:SHIFT CR
SRLEFT:CMRDR H:SHIFT CRSRUP:SHI
FT CRSRLEFT:CMRDR H:SHIFT CRSRUP:SHI
FT CRSRLEFT:SHIFT O:2CMRDR Y":
1780 RETURN
1780 PRINT "HOME":9:CRSRDOWN:29:CRSRRIGHT:
CTRL RVSON:CMRDR:CRSRDOWN:
4:SHIFT CRSRLEFT:CTRL RVSON:SHIFT:
SHIFT CRSRUP:2CMRDR P:CRSRDOWN:
CMRDR H:CRSRDOWN:SHIFT CRSRLEFT:
SHIFT L:CMRDR P"
1790 PRINT "SHIFT CRSRUP:2SHIFT C:2SHI
FT CRSRUP:CTRL RVSON:CRSRDOWN:
3:SHIFT CRSRLEFT:CRSRDOWN:3:SHI
FT CRSRLEFT:CRSRDOWN:3:SHIFT CR
SRLEFT:CRSRDOWN:3:SHIFT CRSRLEF
T":
1800 RETURN
1800 PRINT "HOME":CRSRDOWN:3:CRSRRIGHT:
CMRDR:CTRL RVSON:CMRDR:CRSRD
OWN:3:SHIFT CRSRLEFT:CRSRDOWN:
4:SHIFT CRSRLEFT:SHIFT:CTRL RV
SON:SHIFT:SHIFT CRSRUP:3CMRDR
P:CRSRDOWN:CMRDR H:CRSRDOWN:SHI
FT CRSRLEFT:SHIFT L:CMRDR P"
1810 RETURN
1820 PRINT "HOME":6:CRSRDOWN:3:CRSRRIGHT:
CMRDR:CTRL RVSON:CMRDR:CRSR
DOWN:3:SHIFT CRSRLEFT:CRSRDOWN:
4:SHIFT CRSRLEFT:SHIFT:CTRL RV
SON:SHIFT:SHIFT CRSRUP:3CMRDR
P:CMRDR H:SHIFT CRSRUP:SHIFT CR
SRLEFT:SHIFT O:CMRDR Y"
1830 RETURN
1840 PRINT "HOME":11:CRSRDOWN:3:CRSRRIGHT:
CMRDR:CTRL RVSON:CMRDR:CRSR
DOWN:3:SHIFT CRSRLEFT:CRSRDOWN:
4:SHIFT CRSRLEFT:SHIFT:CTRL RV
SON:SHIFT:SHIFT CRSRUP:3CMRDR
P:CMRDR H:CRSRDOWN:SHI
FT CRSRLEFT:SHIFT L:CMRDR P"
1850 RETURN
1860 PRINT "HOME":16:CRSRDOWN:3:CRSRRIGHT:
CMRDR:CTRL RVSON:CMRDR:CRSR
DOWN:3:SHIFT CRSRLEFT:CRSRDOWN:
4:SHIFT CRSRLEFT:SHIFT:CTRL RV
SON:SHIFT:SHIFT CRSRUP:3CMRDR
P:CMRDR H:SHIFT CRSRUP:SHIFT CR
SRLEFT:SHIFT O:CMRDR Y"
1870 RETURN
1880 PRINT "HOME":21:CRSRDOWN:3:CRSRRIGHT:
CMRDR:CTRL RVSON:CMRDR:CRSR
DOWN:4:SHIFT CRSRLEFT:CTRL RVSON:
CTRL RVSON:CRSRDOWN:4:SHIFT
CRSRLEFT:SHIFT:CTRL RVSON:SHI
FT:SHIFT CRSRUP:11CMRDR P":

```

Continued

THE BOOLEAN BRAIN *Continued*

COMMODORE 64

```

1890 PRINT "CMDR H SHIFT CRSRUP SHIFT
CRSRLEFT CMDR H SHIFT CRSRUP SHI
FT CRSRLEFT CMDR H SHIFT CRSRUP
SHIFT CRSRLEFT SHIFT O CMDR Y":R
RETURN
1900 PRINT "HOME 4 CRSRDOWN 12 CRSRRIGH
T CMDR . CTRL RVSON CMDR . CRS
RDOWN 3 SHIFT CRSRLEFT CRSRDOWN
4 SHIFT CRSRLEFT SHIFT 1 CTRL
RVSON SHIFT 1 SHIFT CRSRUP 10 CM
DR P CRSRDOWN CMDR H CRSRDOWN SH
IFT CRSRLEFT CMDR H CRSRDOWN SHI
FT CRSRLEFT SHIFT L 2 CMDR P"
RETURN
1910 PRINT "HOME 14 CRSRDOWN 12 CRSRRIGH
T CMDR . CTRL RVSON CMDR . CRS
RDOWN 3 SHIFT CRSRLEFT CRSRDOWN
4 SHIFT CRSRLEFT SHIFT 1 CTRL
RVSON SHIFT 1 SHIFT CRSRUP 2 CM
DR P CRSRDOWN CMDR H CRSRDOWN SH
IFT CRSRLEFT SHIFT L 2 CMDR P"
RETURN
1920 PRINT "HOME 14 CRSRDOWN 12 CRSRRIGH
T CMDR . CTRL RVSON CMDR . CRS
RDOWN 3 SHIFT CRSRLEFT CRSRDOWN
4 SHIFT CRSRLEFT SHIFT 1 CTRL
RVSON SHIFT 1 SHIFT CRSRUP 2 CM
DR P CRSRDOWN CMDR H CRSRDOWN SH
IFT CRSRLEFT SHIFT L 2 CMDR P"
RETURN
1930 PRINT "HOME 17 CRSRDOWN 20 CRSRRIGH
T CMDR . CTRL RVSON CMDR . CRS
RDOWN 3 SHIFT CRSRLEFT CRSRDOWN
4 SHIFT CRSRLEFT SHIFT 1 CTRL
RVSON SHIFT 1 SHIFT CRSRUP 2 CM
DR P CRSRDOWN CMDR H CRSRDOWN SH
IFT CRSRLEFT SHIFT L 2 CMDR P"
RETURN
1940 PRINT "HOME 17 CRSRDOWN 20 CRSRRIGH
T CMDR . CTRL RVSON CMDR . CRS
RDOWN 3 SHIFT CRSRLEFT CRSRDOWN
4 SHIFT CRSRLEFT SHIFT 1 CTRL
RVSON SHIFT 1 SHIFT CRSRUP 2 CM
DR P CRSRDOWN CMDR H CRSRDOWN SH
IFT CRSRLEFT SHIFT L 2 CMDR P"
RETURN
1950 PRINT "SHIFT CRSRUP 2 CMDR P CMDR
H SHIFT CRSRUP SHIFT CRSRLEFT CM
DR H SHIFT CRSRUP SHIFT CRSRLEFT
CMDR H SHIFT CRSRUP SHIFT CRSRLE
FT CMDR H SHIFT CRSRUP SHIFT CRS
RLEFT CMDR H SHIFT CRSRUP SHIFT
CRSRLEFT CMDR H SHIFT CRSRUP SHI
FT CRSRLEFT CMDR H SHIFT CRSRUP
SHIFT CRSRLEFT SHIFT O 2 CMDR Y":
RETURN
1960 PRINT "HOME 9 CRSRDOWN 29 CRSRRIGH
T CMDR . CTRL RVSON CMDR . CRS
RDOWN 3 SHIFT CRSRLEFT CRSRDOWN
4 SHIFT CRSRLEFT SHIFT 1 CTRL
RVSON SHIFT 1 SHIFT CRSRUP 2 SHI
FT CRSRUP CTRL RVSON CRSRDOWN
3 SHIFT CRSRLEFT CRSRDOWN 3 SHI
FT CRSRLEFT CRSRDOWN 3 SHIFT CR
SRLEFT CRSRDOWN 3 SHIFT CRSRLE
FT
RETURN
1970 POKES3280,2:PRINT "HOME 24 CRSRDOWN
10 CRSRRIGHT CENTRAL PROCESSING":
FORLP=1TO50:PRINT "HOME 7 CRSRDOWN
IV=ABS(IV-1):GOSUB 2410
IFIV=0THENFORZ=1TO9:PRINT "8 CRSRRIG
HT CTRL RVSON 24 CMDR + 8 CRSRRIG
HT
NEXT
2000 GOSUB 2410
2010 IFIV=1THENFORZ=1TO9:PRINT "8 CRSRRIG
HT CTRL RVSON 24 CMDR + 8 CRSRRIG
HT
NEXT
2020 GOSUB 2410
2030 IFIV=1THENFORZ=1TO9:PRINT "8 CRSRRIG
HT CTRL RVSON 24 CMDR + 8 CRSRRIG
HT
NEXT
2040 GOSUB 2410
2050 NEXT
2060 FORLP=1TO50:PRINT "HOME 7 CRSRDOWN
IV=IV+1:IFIV>4THENIV=1:GOSUB 2410
2070 FORZ=1TO9
2080 ONIVGOTO 2100, 2110, 2120, 2130
2090

```

```

2100 PRINT "8 CRSRRIGHT 24 CMDR F 8 CRSRR
IGHT
2110 PRINT "8 CRSRRIGHT 24 CMDR D 8 CRSRR
IGHT
2120 PRINT "8 CRSRRIGHT 24 CMDR C 8 CRSRR
IGHT
2130 PRINT "8 CRSRRIGHT 24 CMDR V 8 CRSRR
IGHT
2140 NEXT: NEXT
2150 POKES3281,0:POKE54276,64:POKE54296,
0
2160 PRINT "SHIFT CLR CTRL WHT 7 CRSRDO
WN YOU HAVE MADE IT TO THE
2170 PRINT "2 CRSRDOWN CENTRAL
PROCESSOR
2180 PRINT "2 CRSRDOWN YOUR SCORE IS ";S
=INT(DR/SC/DR+10000):PRINTS
2190 IFS<600THEN 2220
2200 PRINT "2 CRSRDOWN YOU REPAIR THE COM
PUTER AND ARE RETURNED TO THE REAL
WORLD
2210 GOTO 2300
2220 PRINT "2 CRSRDOWN YOU ARE TOO LATE.
THE COMPUTERS DATA HAS
2230 PRINT "DEGRADED TO THE POINT OF NO R
ETURN.":GOTO 2300
2240 POKE 54296,15:POKE54276,129
2250 FORZ=1TO100:POKE53280,INT(RND(0)*10
):POKE53281,INT(RND(0)*10)
2260 POKE 54273,INT(RND(0)*100+100):POKE
54296,INT(RND(0)*10+5):NEXT
2270 POKES3280,2:POKE53281,0:POKE54276,6
4:POKE54296,0
2280 PRINT "SHIFT CLR CTRL WHT 4 CRSRDO
WN YOU'VE BEEN ZAPPED BY A BAD DISK
SECTOR. 3 CRSRDOWN
2290 PRINT "ALL DATA IS LOST. THERE IS NO
HOPE OF RECOVERING THE SYSTEM.
2300 PRINT "5 CRSRDOWN WOULD YOU LIKE TO
PLAY AGAIN(Y/N)?
2310 GETAS:IFAS=" " THEN 2310
2320 IFAS="Y" THEN RUN
2330 IFAS">"N" THEN 2310
2340 PRINT "SHIFT CLR BYE, BYE. SEE YOU
NEXT TIME.":END
2350 POKE 54276,64:POKE 54274,96:POKE 54
275,8:POKE 54277,24:POKE 54278,33
2360 RETURN
2370 POKE 54276,65
2380 POKE54296,15:FORZ=0TO255STEP8:POKE5
4273,2:NEXT:POKE54296,0:RETURN
2390 POKE54276,64:POKE54296,15:FORSN=1TO
20:POKE54273,45+SN:POKE54276,65
2400 FORTD=1TO20:NEXT:POKE54276,64:FORTD
=1TO20:NEXT:RETURN
2410 POKE54296,15:POKE54276,65:POKE54273
,INT(RND(0)*200)+50
2420 POKE54296,INT(RND(0)*10+5):RETURN
2430 DATA KEYBOARD INTERFACE,54870000,IN
PUT PORT,43690000
2440 DATA VIDEO PROCESSING,2574000,SOUND
CONTROL ROOM,1235000
2450 DATA RAM ROOM,31480000,DISK CONTROL
ROOM,78920000
2460 DATA DISK DRIVE,76130000,ROM ROOM,6
9510000,PORT CONTROL,80260000
2470 DATA CENTRAL PROCESSING,90000000

```

HCM

THE BOOLEAN BRAIN

TI-99/4A

```

100 REM *****
110 REM * BOOLEAN BRAIN *
120 REM *****
130 REM BY WILLIAM K. BALTHROP
140 REM HOME COMPUTER MAGAZINE
150 REM VERSION 4.4.1
160 REM TI EXTENDED BASIC
170 REM
180 RANDOMIZE :: CALL CLEAR :: CALL SCR
EEN(2)::RESTORE 930 :: FOR Z=0 TO
20 :: READ A,A$ :: CALL CHAR(A,A$):
NEXT Z
190 CALL MAGNIFY(3)::FOR Z=1 TO 9 :: C
ALL SPRITE(#Z,128,7,210,100)::NEXT
Z
200 DIM R$(24),L$(19),RMS(10,2),GT(9,5)
,G(9)
210 FOR Z=1 TO 24 :: READ R$(Z)::NEXT
Z :: FOR Z=1 TO 19 :: READ L$(Z)::
NEXT Z
220 DEF RV(C)=VAL(SEGS(RMS(R,2),C,1))
230 FOR Z=1 TO 10 :: READ RMS(Z,1),RMS(
Z,2)::NEXT Z :: R=1 :: SC=0 :: DR=
0 :: I=0 :: DIR=1
240 FOR Z=1 TO 9 :: CALL COLOR(Z,2,8)::
READ GT(Z,1),GT(Z,2),GT(Z,3),GT(Z,
4),GT(Z,5)::NEXT Z
250 GOSUB 360
260 GOSUB 920 :: IF K=69 THEN DIR=1 ELS
E IF K=87 THEN DIR=2 ELSE IF K=78 T
HEN DIR=3 ELSE IF K=83 THEN DIR=4 E
LSE GOTO 260
270 IF R<>RV(DIR) THEN 320
280 FOR Z=1 TO 40 :: CALL COLOR(RND*12+
1,RND*12+1,RND*12+1)::CALL SOUND(
100,-3,RND*9)::NEXT Z :: FOR Z=1 T
O 8 :: CALL COLOR(Z,16,2)

```

```

290 NEXT Z :: CALL SCREEN(2)
300 CALL CLEAR :: DISPLAY AT (12,1):"YOU
HAVE BEEN ZAPPED BY A BAD DISK S
ECTOR.":THE COMPUTER LOCKS UP, AND
YOU ARE LOST FOR EVER.
310 GOTO 900
320 IF RV(4+DIR)=1 THEN R=RV(DIR)::GOS
UB 370 :: GOTO 260
330 GOSUB 430 :: RMS(R,2)=SEGS(RMS(R,2)
,1,DIR+3)&"1"&SEGS(RMS(R,2),5+DIR,4
-DIR)::R=RV(DIR)
340 IF DIR=1 OR DIR=3 THEN TD=DIR+1 ELS
E TD=DIR-1
350 RMS(R,2)=SEGS(RMS(R,2),1,TD+3)&"1"&
SEGS(RMS(R,2),5+TD,4-TD)::IF R=0 T
HEN 780 ELSE 250
360 CALL CLEAR :: CALL COLOR(1,2,12,2,1
2,6,4,2,12,8,2,8,9,2,8,10,2,8,11,12
,6,12,12,6)::FOR Z=1 TO 24 :: DISP
LAY AT(Z,1):R$(Z)::NEXT Z
370 CALL VCHAR(1,31,106,96)::DISP
LAY AT(24,2)SIZE(26):RPTS(" ",26)::DISP
LAY AT(24,(28-LEN(RMS(R,1)))/2+1)SI
ZE(LEN(RMS(R,1))):RMS(R,1)
380 ON DIR GOSUB 390,400,410,420 :: RET
URN
390 DISPLAY AT(11,18)SIZE(1):"E" :: CAL
L COLOR(12,12,7-RV(5),11,12,7-RV(7)
,2,12,7-RV(8))::RETURN
400 DISPLAY AT(11,18)SIZE(1):"W" :: CAL
L COLOR(12,12,7-RV(6),11,12,7-RV(8)
,2,12,7-RV(7))::RETURN
410 DISPLAY AT(11,18)SIZE(1):"N" :: CAL
L COLOR(12,12,7-RV(5))::RETURN

```

Continued

[illegible]

PROGRAM LISTING


```

100 REM ***** MARKET MADNESS *****
110 REM *****
120 REM *****
130 REM BY BRIAN LEE
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM APPLE II FAMILY APPLESOFT 11: PR
180 HOME : FLASH : VTAB 11: HTAB 11:
INT : STOCK MARKET : NORMAL
V TAB : 13: HTAB 14: PRINT "BY BRIAN L
EE : V TAB 22: HTAB 13: PRINT "PRESS
ANY
KEY"
200 GOSUB 2540
HOME : INPUT "ENTER NUMBER OF PLAYE
RS (1-10) : NP: IF NP < 1 OR NP >
10 THEN 210
PRINT : INPUT "NUMBER OF WEEKS (2 M
INIMUM) : WS: W = INT ( VAL ( LEFT
$ ( WS, 4 ) ) ) : IF W < 2 THEN V TAB 2:
GOTO 220
230 PR$ = ( HIT ANY KEY ) "
240 DIM S ( 6 ) , SS ( 6 ) , P ( NP , ( NP + 1 ) * 6, 2 )
, T ( NP ) , NAS ( NP ) , CM$ ( 6 ) , CV ( 6 ) , I : I : CH ( 6 )
250 PRINT : PRINT : CM$ ( 6 ) : FOR I = 1 TO NP: PR
INT : ENTER NAME OF PLAYER # : NEXT IN
PUT NAS ( I ) : P ( I , 0, 0 ) = 5000 : NEXT
260 FOR I = 1 TO 6: READ SS ( I ) : S ( I ) =
INT ( 50 * RND ( 1 ) ) + S ( I ) : RND = ( 1
) ) : NEXT
270 FOR I = 1 TO 6: READ CM$ ( I ) , CV ( I ) , C
H ( I ) : NEXT
280 DATA "1. US STEEL", "2. PAN AM", "3
. FORD", "4. SANYO", "5. XEROX", "6. A
T&T"
290 DATA "B) UY", "22, 11", "S) ELL", "22, 24", "T
24, 11", "P) ORTFOLIO", "23, 24", "N) EXT"
300 REM *****
310 REM *****
320 REM *****
330 FOR WE = 1 TO W: FOR P = 1 TO NP: TA
= 0: TM = 1
340 HOME
350 PRINT "-----WALL STREET-----"
360 FLASH : HTAB 3: PRINT NAS ( P ) : NORM
AL : HTAB 30: PRINT "WEEK" : WE
370 FOR I = 1 TO 40: PRINT : NEXT :
V TAB 3: HTAB 13: PRINT "STOCK EXC
HANGE" : PRINT
380 PRINT "-----STOCK-----" : HTAB 24:
PRINT : PRINT "PRICE PER SHARE"
390 INVERSE : HTAB 24: PRINT " "
: : NORMAL
400 FOR I = 1 TO 6: PRINT SS ( I ) : HTAB
27: PRINT " " : S ( I ) :
410 IF P ( P, I, 1 ) > 0 THEN INVERSE
420 HTAB 32: PRINT " ( " : P ( P, I, 1 ) ) : NO
RMAL PRINT " " : HTAB 24:
PRINT : NEXT
440 LW = 10000 : FOR I = 1 TO 6: IF S ( I )
< LW AND S ( I ) > 0 THEN LW = S ( I )
: LO
450 NEXT : HI = 0 : FOR I = 1 TO 6: IF S (
I ) > HI THEN HI = S ( I ) : HG = 1
460 NEXT : PRINT "LOW: " : LW : HTAB 24:
PRINT : PRINT "HIGH: " : HI
470 FOR I = 1 TO 40: PRINT " " : NEXT
480 IF RT I THEN RT = 0: RETURN
490 FOR I = 1 TO 6: V TAB CV ( I ) : HTAB CH
( I ) : PRINT CM$ ( I ) : NEXT
500 V TAB 22: HTAB 1: PRINT "MENU: " : INV
ERSE : V TAB 23: PRINT "SELECT: " : VTA
B 24: PRINT "OPTION: " : NORMAL
510 GOSUB 2540: IF KB < 194 OR KN > 213
THEN 510
520 AS = CHR$ ( KB - 128 ) : IF AS = "B"
THEN A = 1: GOTO 590
530 IF AS = "S" THEN A = 2: GOTO 590
540 IF AS = "T" THEN A = 3: GOTO 590
550 IF AS = "L" THEN A = 4: GOTO 590
560 IF AS = "N" THEN A = 5: GOTO 590
570 IF AS = "P" THEN A = 6: GOTO 590
580 GOTO 510
590 INVERSE : V TAB CV ( A ) : HTAB CH ( A ) : P
RINT CM$ ( A ) : NORMAL
600 FOR T = 1 TO 1000: NEXT : ON A GOTO
610, 620, 630, 640, 650, 660, 670, 680, 690
610 REM ***** BUY *****
620 REM *****
630 REM *****
640 POKE 34, 21: HOME
650 PRINT : PRINT "WHICH S
TOCK?" : GET N$ : S = VAL ( N$ ) : IF S > 0 AND S < 7 THEN 670
660 PRINT : CHR$ ( 7 ) : "ENTER STOCK NUMBER
PLEASE : PRINT PR$ : GOSUB 2540
POKE 34, 0: GOTO 340
670 IF S ( S ) = 0 THEN PRINT : CHR$ ( 7 ) :
THAT STOCK IS BANKRUPT : PRINT PR$
: : GOSUB 2540: POKE 34, 0: GOTO 340
680 PRINT : HOW MANY SHARES : INPUT N$ :
SH = INT ( VAL ( LEFT$ ( N$ , 7 ) ) ) : I
F SH < 1 THEN POKE 34, 0: GOTO 340
690 IF SH < S ( S ) * SH > P ( P, 0, 0 ) THEN PRINT
: CHR$ ( 7 ) : "NOT ENOUGH MONEY : TRY
AGAIN : PRINT PR$ : GOSUB 2540: POK
E 34, 0: GOTO 340

```

```

700 P ( P, 0, 0 ) = P ( P, 0, 0 ) + SH : P ( P, S, 1 ) = S ( S )
S ( S ) : SH : P ( P, S, 2 ) = S ( S )
710 HOME : PRINT "TRANSACTION COMPLETE.
. HIT A KEY : GOSUB 2540: TA = 1: BS
= 1: POKE 34, 0: GOTO 2110
720 REM ***** SELL *****
730 REM *****
740 REM *****
750 POKE 34, 21: HOME : PRINT "---- SELL
----" : PRINT "WHICH STOCK?" : GET N$
: PRINT N$ : S = VAL ( N$ ) : IF S < 1
OR S > 6 THEN 660
760 IF S ( S ) = 0 THEN 670
770 IF P ( P, S, 1 ) = 0 THEN ANY SHARES IN THA
T STOCK : PRINT PR$ : GOSUB 2540: P
OKE 34, 0: GOTO 340
780 PRINT : HOW MANY SHARES : INPUT N$ :
SH = INT ( VAL ( LEFT$ ( N$ , 7 ) ) ) : I
F SH < 1 THEN POKE 34, 0: GOTO 340
790 IF P ( P, S, 1 ) < SH THEN PRINT CHR$
( 7 ) : "YOU DON'T HAVE ENOUGH SHARES.
: : PRINT PR$ : GOSUB 2540: POKE 34
, 0: GOTO 340
800 P ( P, S, 1 ) = P ( P, S, 1 ) - SH : P ( P, 0, 0 ) =
P ( P, 0, 0 ) + SH : S ( S ) : IF P ( P, S, 1 ) =
0 THEN P ( P, S, 2 ) = 0
810 PRINT PR$ : GOSUB 2540: POKE 34, 0: T
A = 1: BS = 1: GOTO 2110
820 REM ***** TRADING *****
830 REM *****
840 REM *****
850 I = 0: FOR TR = 1 TO 6: IF P ( P, TR, 1
) > 0 THEN I = 1
860 NEXT : IF I THEN 880
870 POKE 34, 21: HOME : PRINT CHR$ ( 7 ) :
"YOU HAVE NO SHARES TO TRADE : P
RINT PR$ : GOSUB 2540: POKE 34, 0: GO
TO 340
880 HOME : PRINT NAS ( P ) : " : PRINT : PR
INT "YOU HAVE THE FOLLOWING OPTIONS
:
890 PRINT : PRINT : PRINT "1. TRADE SHA
RES WITH ANOTHER PLAYER : PRINT "2.
TRADE SHARES TO ANOTHER STOCK"
900 PRINT : PRINT "3. CASH TO SHARES INTO BANK" : P
RINT : PRINT "4. RETURN TO MENU"
910 PRINT : PRINT "ENTER 1-4 : " : GET N$
: A = VAL ( N$ ) : IF A < 1 OR A > 4 T
HEN 880
920 ON A GOTO 960, 1520, 1410, 340
930 REM *****
940 REM *****
950 REM *****
960 IF NP < 2 THEN PRINT : PRINT CHR$
( 7 ) : "THERE ARE NO OTHER PLAYERS!
: : PRINT PR$ : GOSUB 2540: GOTO 880
970 IF NP = 2 AND P < 2 THEN NN$ = "
2" : TP = 2: GOTO 1060
980 IF NP = 2 THEN NN$ = "1" : TP = 1: GO
TO 1060
990 PRINT : PRINT "YOU MAY TRADE WITH T
HE FOLLOWING PLAYERS ( " :
1000 FOR TR = 1 TO NP: T ( TR ) = 0 THEN PRIN
T "OR " : NP : AND TR < > P THEN PRIN
T " " : TR : " ) : T ( TR ) = 1: GOTO 1040
1010 IF TR < > P THEN T ( TR ) = 1: PRINT
TR : " :
1020 NEXT : PRINT "WHICH PLAYER" : GET
N$ : TP = VAL ( N$ ) : IF TP < 1 OR TP
> NP THEN 850
1030 IF T ( TP ) < 1 THEN 850
1040 HOME : PRINT : PRINT "1. " :
NAS ( TP ) : " : BUYS SHARES FROM YOU" :
1070 PRINT : PRINT "2. YOU BUY SHARES FROM " : NAS
( TP ) : " :
1080 PRINT : PRINT "ENTER CHOICE : " : GE
T AS : IF VAL ( AS ) < 1 OR VAL ( AS )
> 2 THEN 1060
1090 ON VAL ( AS ) GOTO 1130, 1270
1100 REM *****
1110 REM *****
1120 REM *****
1130 REM *****
1140 HOME : PRINT : PRINT "DEAL WITH WHICH STOCK
: : GET N$ : PRINT N$ : S = VAL ( N$ )
: : IF S < 1 OR S > 6 THEN POKE 34,
0: GOTO 880
1150 IF P ( P, S, 1 ) = 0 THEN POKE 34, 0: PR
INT CHR$ ( 7 ) : "YOU DON'T OWN ANY SHA
RES IN THAT STOCK" : PRINT PR$ : GOSU
B 2540: GOTO 880
1160 PRINT : SELL HOW MANY SHARES TO : NA
S ( TP ) : " : INPUT N$ : SH = INT
( VAL ( LEFT$ ( N$ , 7 ) ) )
1170 IF P ( P, S, 1 ) - SH < 0 THEN POKE 34,
0: PRINT CHR$ ( 7 ) : "YOU DON'T HAVE E
NOUGH SHARES : PRINT PR$ : GOSUB
2540: GOTO 880
1180 PRINT : HOW MUCH PER SHARE : " : INPU
T N$ : PS = INT ( VAL ( LEFT$ ( N$ ,
7 ) ) ) : IF PS < S ( S ) / 2 OR PS > S ( S
) THEN 1180
1190 PRINT : IS THIS OKAY, " : NAS ( TP ) : " ? " :
GET N$ : PRINT N$ : IF N$ = "N" THEN
POKE 34, 0: GOTO 1060

```

Continued


```

1200 IF P(0,0) = 0 THEN PRINT "TRANSACTION COMPLETE... HIT A
1210 OKEY!" : GOSUB 2540 : PRINT PR$ : GOTO 880
1220 P(P,S,1) = SH : PS : P(TP,S,1) = P(TP,S
1230 POKE 34,0 : GOTO 880
1240 REM ***** PART 2 OF TRADING *****
1250 REM ***** PART 2 OF TRADING *****
1260 REM ***** PART 2 OF TRADING *****
1270 RT = 1 : Q = P : P = TP : GOSUB 2110 : P =
1280 POKE 34,22 : PRINT "DEAL WITH WHICH VA
1290 LE (NS) : IF S < 1 OR S > 6 THEN = POK
1300 IF P(TP,S,1) = 0 THEN POKE 34,0 : P O
1310 WN THAT CHRS STOCK (7) : NAS (TP) : "DOES NOT
1320 UB PRINT 2540 : GOTO 880 : PRINT PR$ : GOS
1330 $ (P) : " : INPUT (NS) : SH = INT (
1340 $ VAL (LEFTS (NS,7))) : IF SH < 0 THEN POKE 34
1350 IF 0 : PRINT CHRS (7) : "YOU DON'T HAVE
1360 ENOUGH SHARES." : PRINT PR$ : GOS
1370 UB PRINT 2540 : GOTO 880
1380 T "HOW MUCH PER SHARE : " : INPU
1390 T 7) : NS : PS : INT (VAL (LEFTS (NS,7)))
1400 / 2 OR PS > NS (S
1410 * 2 THEN 1320 OKAY : " : NAS (P) : " ? " :
1420 GET NS : PRINT NS : IF NS = "N" THEN
1430 POKE 34,0 : GOTO 1060
1440 IF P(P,0,0) = 0 THEN SH : PS : < 0 THEN P
1450 OKE 34,0 : PRINT CHRS (7) : NAS (P) : "
1460 DOES NOT HAVE THAT MUCH!" : PRINT PR
1470 $ : INT : "TRANSACTION COMPLETE... HIT A
1480 P KEY : " : GOSUB 2540 : GOTO 880
1490 P (TP,S,1) = SH : P (TP,S,1) = SH : P (TP,S,1)
1500 ) = P (TP,S,1) + SH : P (TP,S,1) = P (TP,S,1)
1510 (SH) : PS :
1520 POKE 34,0 : GOTO 880
1530 REM ***** PART 3 OF TRADING *****
1540 REM ***** PART 3 OF TRADING *****
1550 REM ***** PART 3 OF TRADING *****
1560 RT = 1 : GOSUB 340 : POKE 34,21 : PRIN
1570 T "DEAL IN WHICH STOCK : " : GET
1580 R S : THEN = POKE 34,0 : GOTO 880
1590 IF P(P,S,1) = 0 THEN CHRS (7) : PRINT
1600 CK 1 : "YOU DON'T OWN SHARES IN THAT ST
1610 OCK!" : PRINT PR$ : GOSUB 2540 : POKE
1620 34,0 : GOTO 880
1630 PRINT "SELL HOW MANY SHARES : " : IN
1640 PUT " : NS : SH = INT (VAL (LEFTS (NS,7)))
1650 : IF SH = 0 THEN POKE 34,0 :
1660 GOTO 880
1670 IF P(P,S,1) = SH < 0 THEN PRINT C
1680 HRS (7) : "YOU DON'T HAVE ENOUGH SHAR
1690 ES!" : PRINT PR$ : GOSUB 2540 : POKE
1700 34,0 : GOTO 880
1710 HOME : PRINT "THE BANK WILL BUY YOU
1720 R SHARES FOR" : INT (20 * RND (1)) + 1
1730 : PRINT " : " : B : EACH : SOUND FAIR? "
1740 : GET NS : IF NS = "N" THEN POKE 3
1750 4 : GOTO 880
1760 P(P,S,1) = P(P,S,1) - SH : P(P,0,0) =
1770 P(P,0,0) + B * SH : GOTO 880
1780 POKE 34,0 : GOTO 880
1790 REM ***** PART 2 OF TRADING *****
1800 REM ***** PART 2 OF TRADING *****
1810 REM ***** PART 2 OF TRADING *****
1820 RT = 1 : GOSUB 340 : POKE 34,21 : HOME
1830 GET NS : PRINT NS : VAL (NS) : IF
1840 S < 1 OR S > 6 THEN POKE 34,0 : GOT
1850 O 880
1860 IF P(P,S,1) = 0 THEN PRINT CHRS (
1870 7) : "YOU DON'T OWN SHARES IN THAT ST
1880 OCK!" : PRINT PR$ : GOSUB 2540 : POKE
1890 34,0 : GOTO 880
1900 PRINT "SELL HOW MANY SHARES : " : INPUT
1910 " : NS : SH = INT (VAL (LEFTS (NS,7)))
1920 : IF SH = 0 THEN POKE 34,0 : GOTO
1930 880
1940 IF P(P,S,1) = SH < 0 THEN PRINT C
1950 HRS (7) : "YOU DON'T HAVE ENOUGH SHAR
1960 ES!" : PRINT PR$ : GOSUB 2540 : POK
1970 E 34,0 : GOTO 880
1980 PRINT "TRADE FOR WHICH STOCK : " : GET
1990 NS : S2 = VAL (NS) : IF S2 < 1 OR S2
2000 > 6 THEN POKE 34,0 : GOTO 880
2010 IF S(S2) = 0 THEN PRINT CHRS (7) :
2020 "THAT STOCK IS BANKRUPT" : PRINT PR$
2030 : GOSUB 2540 : POKE 34,0 : GOTO 880
2040 IF S(S) = SH < S(S2) THEN POKE 34,
2050 0 : PRINT CHRS (7) : "NOT ENOUGH SHAR
2060 ES TO TRADE" : PRINT PR$ : GOSUB 2540
2070 : GOTO 880
2080 P(P,S,1) = P(P,S,1) - SH : P(P,S2,1)
2090 ) = P(P,S2,1) + (INT ((S(S) * SH) /

```

```

1600 IF (S(S) * SH) / S(S2) > 0 THEN S(S2)
1610 = S(S2) : S(S) = SH : S(S2) = 0 : P(P
1620 ,S2,1) :
1630 POKE 34,0 : GOTO 880
1640 REM ***** LOANING *****
1650 REM ***** LOANING *****
1660 REM ***** LOANING *****
1670 HOME : PRINT NAS (P) : " :
1680 PRINT : PRINT "YOU ARE " : IF P(P,7
1690 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1700 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1710 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1720 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1730 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1740 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1750 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1760 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1770 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1780 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1790 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1800 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1810 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1820 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1830 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1840 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1850 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1860 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1870 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1880 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1890 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1900 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1910 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1920 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1930 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1940 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1950 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1960 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1970 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1980 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
1990 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2000 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2010 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2020 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2030 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2040 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2050 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2060 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2070 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2080 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2090 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2100 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2110 ,0) = 0 THEN PRINT "NOT" : IF P(P,7
2120 ,0) = 0 THEN PRINT "NOT" : IF P(P,7

```

Continued


```

740 IFAS="L" THEN X=23:Y=22:GOTO 800
750 IFAS="P" THEN X=23:Y=23:GOTO 800
760 POKESC,PEEK(SC)OR128:FORH=1TO60:GET
AS
770 IFAS<>" " THEN POKESC,PEEK(SC)AND127:G
OTO 640
780 NEXT:POKESC,PEEK(SC)AND127:FORH=1TO
60:GETAS:ON-(AS<>" ")GOTO 640:::NEX
T
790 GOTO 630
800 SC=1024+X+40*Y:FORR=SCTOSC+10:IFPEE
K(R)<>32 THEN POKER,PEEK(R)OR128
810 NEXT:L=(-(X=23)*3)+(Y-20):FORH=1TO5
00:NEXT
820 PRINTLEFTS$(SS,22):::FORR=1TO24:PRINT
"NEXT:PRINT"
830 ONLGOTO 870, 1160, 2790, 1020,
2260, 2840
840 REM *****
850 REM *****
860 REM *****
870 PRINTLEFTS$(SS,22)"CTRL WHIT--BUY-
--:PRINT"SHIFT CRSRUP"WHICH STOCK
:GOSUB 3410
880 S=VAL(RIGHTS$(NS,1)):ON-(S>=1ANDS<=6
)GOTO 900
890 PRINT"SHIFT CRSRUP"ENTER"CTRL RVS
ON"STOCK NUMBER"CTRL RVS OFF"PLEAS
E":PRINTPR$:::FL=1:GOSUB 3410:::GO
TO 420
900 ON-(S(S)>0)GOTO 920:::PRINTLEFTS$(S
,23)"THAT STOCK IS BANKRUPT":GOTO 9
10
910 PRINTPR$:::FL=1:GOSUB 3410:::GOTO
420
920 PRINT"SHIFT CRSRUP"HOW MANY SHARES
:6SHIFT CRSRLEFT":GOSUB
3410
930 SH=INT(VAL(LEFTS$(NS,7))):ON-(SH=0)G
OTO 420:::IFASC(LEFTS$(NS,1))>57 THEN
920
940 IFS(S)*SH+(S(S)*SH*.006)<=P(P,0,0)T
HEN 960
950 PRINT"SHIFT CRSRUP"NOT ENOUGH MONE
Y...TRY AGAIN":PRINTPR$:::FL=1:GOSUB
3410:::GOTO 420
960 P(P,0,0)=INT(P(P,0,0)-(S(S)*SH+(S(S
)*SH*.006)):P(P,S,1)=P(P,S,1)+SH
970 P(P,S,2)=S(S)*P(P,S,1)
980 PRINT"SHIFT CRSRUP"TRANSACTION COM
PLETE":PRINTPR$:::FL=1:GOSUB 3410:::
TA=1:BS=1:GOTO 2840
990 REM *****
1000 REM *****
1010 REM *****
1020 PRINTLEFTS$(SS,22)"CTRL WHIT--SELL
--:PRINT"SHIFT CRSRUP"WHICH STOC
K:::GOSUB 3410
1030 S=VAL(RIGHTS$(NS,1)):IFS<=0ORS>6 THEN
890
1040 ON-(S(S)>0)GOTO 1050:::PRINTLEFTS$(S
,23)"THAT STOCK IS BANKRUPT":GOTO 9
10
1050 PRINT"SHIFT CRSRUP"HOW MANY SHARES
:6SHIFT CRSRLEFT":GOSUB
3410
1060 SH=INT(VAL(LEFTS$(NS,7))):ON-(SH=0)G
OTO 420:::IFASC(LEFTS$(NS,1))>57 THEN 1
050
1070 IFP(P,S,1)>=SH THEN 1100
1080 PRINT"SHIFT CRSRUP"YOU DON'T HAVE
ENOUGH SHARES:::PRINTPR$:::FL=1
1090 GOSUB 3410:::GOTO 420
1100 P(P,S,1)=P(P,S,1)-SH:P(P,0,0)=INT(P
(P,0,0)+SH*S(S)-(SH*S(S)*.006)):
1110 P(P,S,2)=P(P,S,1)*S(S)
1120 PRINTPR$:::FL=1:GOSUB 3410:::TA=1:B
S=1:GOTO 2840
1130 REM *****
1140 REM *****
1150 REM *****
1160 PRINT"SHIFT CLRR"CTRL WHIT"PLAYER:"
P:PRINT"2CRSRDOWN"YOU HAVE THE FOL
LOWING OPTIONS:
1170 PRINT"2CRSRDOWN"1. TRADE SHARES WI
TH ANOTHER PLAYER"
1180 PRINT"2. TRADE SHARES FOR ANOTHER S
TOCK":PRINT"3. CASH SHARES INTO BAN
K"
1190 PRINT"4. RETURN TO MENU"
1200 PRINT"CRSRDOWN"ENTER 1-4:":FL=1:
GOSUB 3410:::IFAS<"1"ORAS>"4" THEN
1160
1210 ONVAL(AS)GOTO 1250, 2060, 1910,
420
1220 REM *****
1230 REM *****
1240 REM *****
1250 IFNP>1 THEN 1280
1260 PRINT:PRINT"CRSRDOWN"NO OTHER PLAY
ERS TO TRADE WITH...:PRINTPR$:::FL=
1
1270 GOSUB 3410:::GOTO 1160
1280 IFNP=2ANDP<>2 THEN NS="2":TP=2:GOTO
1390
1290 IFNP=2ANDP=2 THEN NS="1":TP=1:GOTO 1
390
1300 PRINT:PRINT"YOU MAY TRADE WITH THE
FOLLOWING PLAYERS(
1310 FORTR=1TONP:T(TR)=0
1320 IFTR=NPANDTR<>P THEN PRINT"OR"TR:::T(T
R)=1:GOTO 1350

```

```

1330 IFTR<>P THEN T(TR)=1:PRINTRIGHT$(STR$
(TR),LEN(STR$(TR))-1)"
1340 NEXT
1350 PRINT")":PRINT"WHICH PLAYER: ";:GOS
UB 3410
1360 TP=VAL(NS):IFTTP>NPORTP<1ORTP=P THENP
RINT"SHIFT CRSRUP":GOTO 1350
1370 IFT(TP)=1 THEN 1390
1380 GOTO 1160
1390 PRINT"SHIFT CLRR"2CRSRDOWN"1. SELL
SHARES TO NS
1400 PRINT"2. BUY SHARES FROM NS
1410 PRINT"CRSRDOWN"ENTER CHOICE: "FL
=1:GOSUB 3410:::IFAS<"1"ORAS>"2" TH
EN 1390
1420 ONVAL(AS)GOTO 1460, 1660
1430 REM *****
1440 REM *****
1450 REM *****
1460 RT=1:GOSUB 2840
1470 PRINTLEFTS$(SS,23)"DEAL WITH WHICH S
TOCK:::GOSUB 3410
1480 S=INT(VAL(LEFTS$(NS,1))):IFS<=0ORS>6
THEN 1160
1490 PRINT:PRINT"2SHIFT CRSRUP"SELL HOW
MANY SHARES TO PLAYER"TP":GOSUB
3410
1500 SH=INT(VAL(LEFTS$(NS,7))):IFP(P,S,1)
-SH>0 THEN 1520
1510 PRINT"SHIFT CRSRUP"YOU DON'T HAVE
ENOUGH SHARES":PRINTPR$:::FL=1:GOSUB
3410:::GOTO 1160
1520 PRINT:PRINT"SHIFT CRSRUP"HOW MUCH
PER SHARE:::GOSUB 3410:::PS=INT(
VAL(NS)/S)
1530 IFPS<S(S)/2ORPS>S(S)*2 THEN PRINT"BAD
VALUE"SHIFT CRSRUP":FORTD=1TO1000
0:NEXT:GOTO 1520
1540 IFP(TP,0,0)-(SH*PS)>0 THEN 1570
1550 PRINT"PLAYER"TP" DOES NOT HAVE THAT
MUCH":PRINT"money"
1560 PRINTTAB(20)PR$:::FL=1:GOSUB 3410:::
GOTO 1160
1570 PRINT"SHIFT CRSRUP"DO YOU AGREE #
TP"SHIFT CRSRLEFT"(Y/N)?":INPUT
ANS:IFLEFT$(ANS,1)="N" THEN 1460
1580 IFLEFT$(ANS,1)<>"Y" THEN PRINT"SHIFT
CRSRUP":GOTO 1570
1590 PRINT:PRINT"2SHIFT CRSRUP"TRANSACTION
HAS BEEN COMPLETED...
1600 P(P,S,1)=P(P,S,1)-SH:P(P,0,0)=P(P,0
,0)+SH*PS:P(TP,0,0)=P(TP,0,0)-SH*PS
1610 P(TP,S,1)=P(TP,S,1)+SH:PRINTPR$:::FL
=1:GOSUB 3410
1620 ON-(P(P,S,1)=0)GOTO 420:::GOTO 11
60
1630 REM *****
1640 REM *****
1650 REM *****
1660 RT=1:GOSUB 420:::FORR=1TO6:PRINTLE
FTS$(SS,6+(S*2))SPC(20):
1670 PRINTCHR$(20)CHR$(20)CHR$(20)SPC(11
)CHR$(148)CHR$(148)CHR$(32):
1680 PRINT"RIGHTS$(STR$(P(P,S,1)),LEN(S
TR$(P(P,S,1))-1))"
1690 PO=37-POS(0):IFPO<0 THEN PO=PO+37
1700 PRINTSPC(PO)"CTRL VEL"RIGHTS$(ST
R$(P(TP,S,1)),LEN(STR$(P(TP,S,1)))-
1)"CTRL WHIT":NEXT
1710 PRINTLEFTS$(SS,22)"PLAYER"P":BUY WH
ICH STOCK?:GOSUB 3410
1720 S=INT(VAL(LEFTS$(NS,1))):IFS<=0ORS>6
THEN 1160
1730 PRINT"HOW MANY SHARES":GOSUB 3410:
SH=INT(VAL(LEFTS$(NS,7)))
1740 IFSH<=P(TP,S,1) THEN 1760
1750 PRINT"SHIFT CRSRUP"NOT ENOUGH SHAR
ES:::PRINTPR$:::FL=1:GOSUB 3410:::
GOTO 1160
1760 PRINT"SHIFT CRSRUP"HOW MUCH PER SH
ARE?:GOSUB 3410:::PS=INT(VAL(LEFTS
$(NS,7)))
1770 IFPS<S(S)/2ORPS>S(S)*2 THEN PRINT"BAD
VALUE"SHIFT CRSRUP":FORTD=1TO1000
0:NEXT:GOTO 1760
1780 IFPS*SH<=P(P,0,0) THEN 1800
1790 PRINT"SHIFT CRSRUP"YOU DON'T HAVE
ENOUGH MONEY":FORTD=1TO1000:NEXT:GO
TO 1160
1800 PRINT"SHIFT CRSRUP"PLAYER #TP"SH
IFT CRSRLEFT"DO YOU AGREE (Y/N)?":
1810 INPUTANS:IFLEFT$(ANS,1)="N" THEN 284
0
1820 IFLEFT$(ANS,1)<>"Y" THEN PRINT"SHIFT
CRSRUP":GOTO 1800
1830 PRINT"SHIFT CRSRUP"
1840 P(P,S,1)=P(P,S,1)+SH:P(TP,S,1)=P(TP
,S,1)-SH:P(TP,0,0)=P(TP,0,0)+SH*PS
1850 P(P,0,0)=P(P,0,0)-SH*PS:PRINT"2SHI
FT CRSRUP"CTRL WHIT"TRANSACTION HAS
BEEN COMPLETED...
1860 PRINTPR$:"5SHIFT CRSRLEFT":GOSUB
3410
1870 ON-(P(P,S,1)=0)GOTO 420:::GOTO 11
60
1880 REM *****
1890 REM *****
1900 REM *****

```

Continued


```

1910 RT=1:GOSUB 420::PRINTLEFT$(S$,23)
    0"DEAL IN WHICH STOCK:":GOSUB 341
1920 S=VAL(RIGHT$(N$,1)):IFS<=0ORS>6THEN
    1160
1930 PRINT:PRINT"2SHIFT CRSRUPSELL HOW
    MANY SHARES:6SHIFT CRSRLEF
1940 T":GOSUB 3410:IFSH<0THEN 1160
    SH=INT(VAL(N$)):IFSH<0THEN 1160
1950 IFP(P,S,1)-SH>0THEN 1980
1960 PRINT:PRINT"2SHIFT CRSRUPYOU DON'
    T HAVE ENOUGH SHARES...:PRINTPR$;:
    FL=1
1970 GOSUB 3410::GOTO 1160
1980 PRINT:PRINT"SHIFT CRSRUPTHE BANK
    WILL BUY YOUR SHARES FOR"
1990 B=S(S)+INT(10*RDND(1)-5):PRINT"$"RIG
    HT$(STR$(B),LEN(STR$(B))-1)"EACH."
2000 PRINTTAB(20)"SOUND FAIR?":;FL=1:GO
    SUB 3410::IFAS<>"Y"THEN 1160
2010 P(P,S,1)=P(P,S,1)-SH:P(P,0,0)=P(P,0
    ,0)+B*SH
2020 ON-(P(P,S,1)=0)GOTO 420::GOTO 11
    60
2030 REM *****
2040 REM *****PART 2 OF LOANING*****
2050 REM *****
2060 RT=1:GOSUB 420::PRINTLEFT$(S$,23)
    0"SHIFT CRSRUPTRADE IN WHICH STOCK
    ":GOSUB 3410
2070 S=VAL(RIGHT$(N$,1)):IFS<=0ORS>6THEN
    1160
2080 PRINT"FOR WHICH STOCK:":;GOSUB 341
    0::TS=VAL(LEFT$(N$,1))
2090 IFTS<1ORTS>6ORTS=1THEN 2080
2100 PRINT:PRINT"2SHIFT CRSRUPTRADE HO
    W MANY SHARES:11SHI
2110 SH=INT(VAL(N$)):IFSH<0THEN 1160
2120 IFP(P,S,1)-SH>0THEN 2150
2130 PRINT:PRINT"2SHIFT CRSRUPYOU DON'
    T HAVE ENOUGH SHARES...:PRINTPR$;:
    FL=1
2140 GOSUB 3410::GOTO 1160
2150 V1=INT((S(S)+SH)/NT):NT=INT(V1/S(TS)):RS
    =INT(V1-(S(TS)*NT))
2160 PRINT"2SHIFT CRSRUPYOU TRADE"SH"
    SHIFT CRSRLEFT"SHARES OF STOCK"S"
2170 PRINT"FOR"NT"SHIFT CRSRLEFT"SHAR
    ES OF STOCK"TS":PRINT"$"RS"IS
    LEFT AS CASH"
2180 PRINTTAB(20)"SOUND FAIR?":;FL=1:GO
    SUB 3410::IFAS<>"Y"THEN 1160
2190 P(P,S,1)=P(P,S,1)-SH:P(P,0,0)=P(P,0
    ,0)+RS:P(P,TS,1)=P(P,TS,1)+NT=P(P,0
    ,0)+P(P,S,1)=0
2200 IFP(P,S,1)=0THENP(P,S,0)=0
2210 P(P,TS,0)=S(TS)
2220 FORTD=1TO2000:NEXT:GOTO 420
2230 REM *****
2240 REM *****LOANING*****
2250 REM *****
2260 PRINT"SHIFT CLRCTRL WHTPLAYER:"
    P
2270 PRINT"2CRSRDOWNYOU ARE":IFP(P,7
    ,0)=0THENPRINT"NOT":GOTO 2290
2280 PRINT"$"RIGHT$(STR$(P(P,7,0)),LEN(S
    TR$(P(P,7,0))-1))
2290 PRINT"IN DEBT"
2300 PRINT"2CRSRDOWNYOU HAVE THE FOLLO
    WING OPTIONS:DOWN1.MAKE A LOAN WITH
    THE BANK":PRINT"2.PAY BACK A LOAN
2320 PRINT"3.COMPOUND INTEREST ON A FUT
    URE LOAN"
2330 PRINT"4.RETURN TO MAIN MENU"
2340 PRINT"2CRSRDOWNENTER CHOICE:":;F
    L=1:GOSUB 3410::C=VAL(AS):IFC<1OR
    C>4THEN 2260
2350 PRINT:ONCGOTO 2390, 2510, 2670,
    420
2360 REM *****
2370 REM *****PART 1 OF LOANING*****
2380 REM *****
2390 PRINT"2CRSRDOWNHOW MUCH TO BORROW
    (N$,7)):GOSUB 3410::B=INT(VAL(LEFT$(
    N$,7))
2400 IFB=0THEN 2260
2410 SV=0:FORZZ=1TO6:SV=SV+P(P,ZZ,0):NEX
    T:IFSV<5000THENLI=5000:GOTO 2430
2420 LI=SV
2430 IFB+P(P,7,0)<=LI THEN 2460
2440 PRINT"2CRSRDOWNSORRY...YOU ARE A
    CREDIT RISK":PRINT"YOUR LIMIT IS"L
    I
2450 PRINT"CRSRDOWN"PR$;:FL=1:GOSUB 3
    410::GOTO 2260
2460 PRINT"OK...TRANSACTION HAS BEEN APP
    ROVED":P(P,7,0)=P(P,7,0)+B
2470 P(P,0,0)=P(P,0,0)+B:N$=STR$(B):GOTO
    2720
2480 REM *****
2490 REM *****PART 2 OF LOANING*****
2500 REM *****
2510 PRINT"2CRSRDOWN":IFP(P,7,0)>0THE
    N 2540
2520 PRINT"YOU DO NOT NEED TO MAKE PAYME
    NTS...:PRINTPR$;:FL=1:GOSUB 3410
    GOTO 2260

```

```

2540 PRINT"YOU OWE $"RIGHT$(STR$(P(P,7,0
    )),LEN(STR$(P(P,7,0))-1))"CRSRDOWN
    N
2550 PRINT"HOW MUCH ARE YOU":PRINT"GOING
    TO PAY BACK:":GOSUB 3410
2560 B=INT(VAL(N$)):IFB<=P(P,0,0)THEN 2
    590
2570 PRINT"YOU CAN'T AFFORD THAT LARGE O
    F A PAYMENT"
2580 PRINTPR$;:FL=1:GOSUB 3410::GOTO
    2260
2590 IFP(P,7,0)-B<=0THENPRINT"ALL, I PRE
    SUME...:P(P,7,0)=P(P,7,0)-B:P(P,0,0)=P(P,0
    ,0)-B
2600 IFP(P,7,0)<0THENP(P,7,0)=0:P(P,0,0)
    =P(P,0,0)-(B):OP=1
2610 IFOP=1THENOP=0:PRINT"YOU HAVE OVERP
    AID, BUT":PRINT"IT HAS BEEN CORRECT
    ED"
2620 PRINT"CRSRDOWN"PR$;:FL=1:GOSUB 3
    410::GOTO 2260
2630 REM *****
2640 REM *****PART 3 OF LOANING*****
2650 REM *****
2660 REM *****
2670 PRINT"SHIFT CLRCTRL WHTPLAYER:"
    P
2680 PRINT"2CRSRDOWNINTEREST WILL BE C
    OMPOUNDED FOR THE":PRINT"DURATION O
    F STARTING"
2690 PRINT"NOW AND UNTIL THE END OF TH
    E GAME ("
2700 PRINTRIGHT$(STR$(W-(WE-1)),LEN(STR$
    (W-(WE-1))-1))WEEK(S)
2710 PRINT"2CRSRDOWNHOW MUCH MONEY WOU
    LD":PRINT"YOU NEED TO BORROW:":GO
    SUB 3410
2720 M=INT(VAL(LEFT$(N$,7))):I=((W-(WE-1
    ))*.01)*M:PRINT"CRSRDOWNINTEREST:
    $"
2730 PRINTRIGHT$(STR$(I),LEN(STR$(I))-1)
    :TAB(20)"TOTAL: $"
2740 PRINTRIGHT$(STR$(I+M),LEN(STR$(I+M)
    )-1)
2750 PRINT"2CRSRDOWN"PR$;:FL=1:GOSUB
    3410::GOTO 2260
2760 REM *****
2770 REM *****SKIP TURN (NEXT)*****
2780 REM *****
2790 PRINTLEFT$(S$,23)"ARE YOU FORFEITIN
    G YOUR TURN:":FL=1:GOSUB 3410
2800 ON-(AS="Y")GOTO 3120::GOTO 420
2810 REM *****
2820 REM *****PORTFOLIO*****
2830 REM *****
2840 PRINT"SHIFT CLRCTRL WHT"
    T
2850 PRINT"CTRL RVSON"PLAYER:"PTAB(30
    )"CTRL RVSOFF"WE:FORB=1TO15:
    PRINT"NEXT
2860 PRINT"CTRL YEL"PORTFOLIO"CTRL WHT
    :FORB=1TO15:PRINT"NEXT:PRINT
2870 POKE646,15:PRINT"CRSRDOWN"STOCK
    PLUS"TAB(23)"NO. OF SHARES"
2880 PRINT"SHIFT CRSRUPCURRENT PRICE
    CHR$(13)TAB(24)"SHIFT CRSRUP/INV
    ESTED VALUE"
2890 PRINT"16CMR U"TAB(23)"17CMR U
    CTRL WHT"
2900 FORS=1TO6:P(P,S,2)=P(P,S,1)*S(S):NE
    XT
2910 FORS=1TO6
2920 PRINTS(S)CHR$(13)TAB(11)"SHIFT CR
    SRUP"RIGHT$(STR$(S(S)),LEN(STR$
    (S(S))-1))
2930 PO=45-POS(0):IFPO<0THENPO=PO+20
2940 PRINTSPC(PO)P(P,S,1)
2950 PO=3-(LEN(STR$(P(P,S,1)))-1):IFPO<0
    THENPO=0
2960 PRINTSPC(PO)"/"RIGHT$(STR$(P(P,
    S,2)),LEN(STR$(P(P,S,2))-1))
2970 PRINT"TAB(23)"
2980 PRINT"OWNED: $"RIGHT$(STR$(P(P,0,0)
    ),LEN(STR$(P(P,0,0))-1))
2990 PRINTSPC(14)"INVEST: $":I=0:FORB=1
    TO6:I=I+(S(S)*P(P,S,1)):NEXT
3000 PRINTRIGHT$(STR$(I),LEN(STR$(I))-1)
3010 PRINT"LOANS: $"RIGHT$(STR$(P(P,7,0)
    ),LEN(STR$(P(P,7,0))-1)):POKE646,1
    4
3020 PRINTTAB(23)"TOTAL: $":
    T=(P(P,0,0)+I)-P(P,7,0):PRINTRIGHT$
    (STR$(T),LEN(STR$(T))-1)
3040 PRINT"CTRL WHT":FORB=1TO40:PRINT
    "NEXT
3050 IFRT=1THENRT=0:RETURN
3060 IFTA<>1ORTM=2THEN 3110
3070 ON-(BS<>1)GOTO 3110::BS=0
3080 TR=0:TM=TM+1
3090 PRINT"CRSRDOWNANOTHER TRANSACTION
    (Y/N)":;FL=1:GOSUB 3410::IFAS=
    "Y"THEN 420
3100 GOTO 3120
3110 PRINTPR$;:FL=1:GOSUB 3410::IFL=6T
    HEN 420
3120 IFP<NPTHEN 3170

```

Continued


```

3130 FOR S=1 TO 6: CG=INT(RND(1)*20/CH(S,1))+
    CH(S,2)*SGN(RND(1)*10)-3: S(S)=S(S)
    +CG
3140 IFS(S)<0 THEN S(S)=0
3150 IFS(S)>150 THEN S(S)=INT(S(S)/2+1): FO
    RZ=1: TONP:P(Z,S,1)=P(Z,S,1)*2: NEXT
3160 NEXT
3170 NEXT P: FORC=1 TONP: P(C,7,0)=P(C,7,0)+
    (P(C,7,0)*.005)
3180 P(C,7,0)=INT(P(C,7,0))
3190 NEXT: ON (W<W-1) GOTO 3220
3200 PRINT "SHIFT CLR LEFTS (S,10) TAB(6
    )" "LAST WEEK FOR TRANSACTIONS..."
3210 PRINT "2 CRSR DOWN TAB(13) PRS: FL=1:
    GOSUB 3410
3220 IF INT(W/4)=WEE/4 THEN 3240
3230 FORZ=1 TO 6: CH(Z,1)=INT(RND(1)*5+1): C
    H(Z,2)=INT(RND(1)*7-3): NEXT
3240 NEXT W
3250 PRINT "SHIFT CLR CRSR DOWN END OF G
    AME..." 2 CRSR DOWN: FORH=1 TO 500: NEXT
3260 W=0: TE=0: TR=0: FORW=1 TONP: P(W,0,0)=
    P(W,0,0)-P(W,7,0)
3270 FORZ=1 TO 6: P(W,0,0)=P(W,0,0)+P(W,Z,1
    )*S(Z): NEXT
3280 IF P(W,0,0)=WITHENTE=1: TR=W: GOTO 33
    00
3290 IF P(W,0,0)>WITHENW=P(W,0,0): WR=W
3300 NEXT: FORW=1 TONP: PRINT "PLAYER: "W"$
    P(W,0,0)
3310 IF P(W,0,0)=5000 THEN PRINT TAB(11)"$ "P
    (W,0,0) "EVEN: "GOTO 3340
3320 IF P(W,0,0)<5000 THEN PRINT TAB(11)"$ "P
    (W,0,0) "LOSS: "GOTO 3340
3330 PRINT TAB(11)"$ "P(W,0,0)-5000 "PROFIT
    "
3340 NEXT
3350 IF TE=1 THEN PRINT "TIE BETWEEN PLAYER:
    "WR" AND "TR: GOTO 3370
3360 PRINT "2 CRSR DOWN WINNER: CTRL RV SO
    N: PLAYER WR
3370 GETAS: ON (AS="") GOTO 3370::: RUN
3380 REM *****
3390 REM ***** CURSOR SUBROUTINE *****
3400 REM *****
3410 NS=""
3420 GETAS: IF AS="" THEN 3490
3430 IFFL THEN FL=0: PRINT "RETURN
3440 IF AS=CHRS(13) THEN PRINT "RETURN
3450 IF AS=CHRS(20) AND LEN(NS)>0 THEN 3480
3460 IF AS<CHRS(31) OR AS>CHRS(91) THEN 342
    0
3470 PRINTAS: NS=NS+AS: GOTO 3420
3480 NS=LEFT$(NS,LEN(NS)-1): PRINTAS: GOT
    O 3420
3490 FORC=1 TO 4: PRINTC$(C): FORH=1 TO 10: GE
    TAS: ON (AS<>"") GOTO 3430::: NEXT: NE
    XT
3500 GOTO 3420
3510 REM *****
3520 REM ***** INSTRUCTIONS *****
3530 REM *****
3540 PRINT TAB(9) "2 CRSR DOWN CTRL WHT IN
    STRUCTIONS (Y/N): "FL=1: GOSUB 3410
    : IF AS<>"Y" THEN RETURN
3550 PRINT "SHIFT CLR CTRL WHT: "FORB=
    1 TO 40: PRINT "NEXT
3560 PRINT "THIS IS A MOCK SIMULATION OF
    HOW A STOCK MARKETING PERIOD:
3570 PRINT "MIGHT GO OVER A PERIOD OF TIM
    E. IN THE SIMULATION, YOU COMPLETE"
3580 PRINT "WITH ANOTHER PLAYER(S) OR YOU
    RSELF IF YOU WISH."
3590 PRINT "CRSR DOWN YOU BEGIN WITH $5.0
    0 AND YOUR GOAL IS TO ACHIEVE THE
    MOST

```

```

3600 PRINT "MONEY BY THE ENDING OF THE PL
    AYING PERIOD (WHICH IS SELECTED BY "
3610 PRINT "YOU). IN ORDER TO DO THIS, H
    ERE ARE SOME GAME-PLAYING CLUES:
3620 PRINT "CRSR DOWN YOUR OBJECTIVE: TO
    BUY AND SELL STOCKS. IN ORDER TO DO
    THIS, WITH SOME
3630 PRINT "BENEFICIARY RESULTS, YOU SHOU
    LD BUY LOW AND SELL HIGH
3650 PRINT "(MEANING THE SHARES OF THE STO
    CKS)"
3660 PRINT "YOU MAY ALSO TRADE OR CASH IN
    YOUR: PRINT "SHARES WHICH YOU HAVE
    "
3670 PRINT "ALREADY PURCHASED. YOU MAY TRA
    DE SHARES WITH ANOTHER PLAYER,
3680 PRINT "ANOTHER STOCK, OR THE BANK:
3690 FORB=1 TO 40: PRINT "NEXT
3700 PRINT "PRESS ANY KEY TO CONTINUE: "
    FL=1: POKE198,0: GOSUB 3410
3710 PRINT "SHIFT CLR: "FORB=1 TO 40: PRIN
    T "NEXT
3720 PRINT "HERE IS A BRIEF DESCRIPTION O
    F EACH: PRINT "OPTION:
3730 PRINT "CRSR DOWN BUY - ALLOWS YOU TO
    PURCHASE A NUMBER OF SHARES (UP TO
    YOUR MONEY:
3740 PRINT "LIMIT): PRINT "CRSR DOWN SEL
    L - ALLOWS YOU TO SELL A NUMBER OF
    SHARES:
3750 PRINT "(UP TO THE AMOUNT OF SHARES Y
    OU HAVE)."
3760 PRINT "CRSR DOWN TRADE - ALLOWS YOU
    TO TRADE SHARES WITH VARIOUS OPTION
    S"
3770 PRINT "CRSR DOWN LOAN - ALLOWS YOU T
    O WITHDRAW A LOAN"
3780 PRINT "FROM THE BANK IF YOU SHOULD F
    IND"
3790 PRINT "YOURSELF BANKRUPT!"
3800 PRINT "CRSR DOWN NEXT - SKIP YOUR TU
    RN"
3810 FORB=1 TO 40: PRINT "NEXT: PRINT "PRE
    SS ANY KEY TO CONTINUE: "FL=1
3820 POKE198,0: GOSUB 3410: PRINT "SHIF
    T CLR OPTIONS (CONT): "FORB=1 TO 25:
    PRINT "NEXT
3830 PRINT "CRSR DOWN PORTFOLIO - THIS IS
    A MENU DISPLAYING THE AMOUNT OF
    MONEY YOU:
3840 PRINT "HAVE, HOW MUCH INVESTED, HOW
    MUCH BORROWED, NUMBER OF SHARES "
3850 PRINT "AND THE TOTAL AMOUNT YOUR: PR
    INT "PORTFOLIO IS WORTH."
3860 PRINT "CRSR DOWN WHENEVER A STOCK IS
    BANKRUPT, MEANING IN THIS CASE THE
    SHARE:
3870 PRINT "EQUALS ZERO, YOU CANNOT BUY O
    R SELL TO THE STOCK. A MESSAGE WILL
    "
3880 PRINT "BE PRINTED IF YOU TRY TO DO
    SO WITHOUT ANY PENALTY."
3890 PRINT "CRSR DOWN UP TO THREE TRANSAC
    TIONS MAY BE MADE PER PLAYER DURING
    THE WEEK.
3900 FORB=1 TO 40: PRINT "NEXT: PRINT "CR
    SR DOWN PRESS ANY KEY TO BEGIN: "FL
    =1
3910 GOSUB 3410::: RETURN

```

HCM

PROGRAM LISTING

MARKET MADNESS

IBM PC & PCjr

```

100 *****
110 ***** MARKET MADNESS *****
120 *****
130 BY BRIAN LEE
140 AND THE HCM STAFF
150 HOME COMPUTER MAGAZINE
160 VERSION 4.4.1
170 IBM PCjr WITH CARTRIDGE BASIC
180 IBM PC WITH BASICA
190
200 WIDTH 40: SCREEN 0: OPTION BASE 1: DIM
    S(6), S$(6), P(10,6), PC(10,2), CH(6,2
    ): KEY OFF: RANDOMIZE TIMER
210 DEF FNMAX(V1,V2)=(V1>V2)*V1+(V
    2>V1)*V2+(V1=V2)*V1+(V1<V2)*V2
220 DEF FNMIN(V1,V2)=(V1<V2)*V1+(V
    2<V1)*V2+(V1=V2)*V1+(V1>V2)*V2
230 RESTORE 1820: READ NGS, TL1$, TL2$, LAA
    $, LAB$, LBAS$, LBS$, FOR Z=1 TO 6: READ
    S$(Z): CH(Z,1)=INT(RND*4)+1: CH(Z,2)=
    INT(RND*7)-3: NEXT
240 CLS: LOCATE 12,14: PRINT "STOCK MARKE
    T": LOCATE 24,9: PRINT "PRESS [ENTER]
    TO START: "GOSUB 1680
250 CLS: LOCATE 1,1: INPUT "HOW MANY PLAY
    ERS (1-10) "AS: GOSUB 1690: IF STAT
    =1 THEN 250 ELSE NP=VAL(AS): IF NP<1
    OR NP>10 THEN 250 ELSE FOR Z=1 TO

```

```

260 LOCATE Z*2+1,1: PRINT "PLAYER #": STR
    $(Z): "S NAME: "INPUT NAMS(Z): IF N
    AMS(Z)="" THEN 260 ELSE NAMS(Z)=LEF
    T$(NAMS(Z),11): NEXT
270 LOCATE 20,1: INPUT "HOW MANY WEEKS (
    AT LEAST 2) "AS: GOSUB 1690: IF STA
    T=1 THEN 270 ELSE NW=VAL(AS): IF NW<
    2 THEN 270
280 FOR Z=1 TO 6: S(Z)=INT(RND*40)+10: FO
    R Z1=1 TO NP: PC(Z1,1)=5000: PC(Z1,2)
    =0: P(Z1,Z)=0: NEXT: NEXT: PN=1: W=1
    TL$=TL1$: LAS=LAAS$: LBS=LAB$: GOSUB 17
    50
300 GOSUB 1770: LSV=S(1): HSV=0: FOR Z=0 T
    O 5: LOCATE 7+Z*2,22: PRINT USING "$$
    #": S(Z+1): PRINT TAB(30): PRINT U
    SING "#####": P(PN,Z+1): LSV=FN
    MIN(S(Z+1),LSV): HSV=FNMAX(HSV,S(Z+1
    )): NEXT: LOCATE 19,6: PRINT LSV: LOCAT
    E 19,34: PRINT HSV
310 GOSUB 1680: IF AS="B" THEN GOSUB 360
    ELSE IF AS="T" THEN GOSUB 440 ELSE IF
    AS="N" THEN GOSUB 1120 ELSE IF
    AS="S" THEN GOSUB 1230 ELSE IF AS="
    L" THEN GOSUB 1310 ELSE IF AS="P" T
    HEN GOSUB 1500 ELSE GOTO 310
320 IF PG=1 THEN 300 ELSE GOTO 290
330 REM *****
340 REM ***** BUY STOCKS *****

```

Continued


```

350 REM **
360 PG=1:GOSUB 1710:LOCATE 22,1:PRINT "
WHICH STOCK (1-6) THEN GOSUB 168
0:IF AS<0 OR AS>6 THEN SOUND 11
0:1:GOTO 360 ELSE IF AS=0 THEN RE
TURN
370 IF T(SN)=0 THEN LOCATE 22,1:PRINT "
THAT STOCK IS BANKRUPT -- TRY AGAIN
380 LOCATE 22,1:PRINT "HOW MANY SHARES
OF S(SN):INPUT AS:GOSUB 1690:IF
STAT=1 THEN 380 ELSE NOS=INT(VAL(A
$)):IF NOS=0 THEN RETURN
390 COST=S(SN)*NOS:CHRG=INT(COST*.006):
IF COST+CHRG>PC(PN,1) THEN PRINT "Y
OU DON'T HAVE ENOUGH MONEY !!":GOTO
380
400 PRINT "TRANSACTION COMPLETE -- FOR
$":COST+CHRG:PC(PN,1)=PC(PN,1)-COST
--CHRG:P(PN,SN)=P(PN,SN)+NOS:FOR TD=
1 TO 1000:NEXT:GOSUB 1500:RETURN
410 REM **
420 REM ** TRADE STOCKS **
430 REM **
440 PG=0:GOSUB 1710:LOCATE 21,1:PRINT "
1) TRADE WITH PLAYERS":PRINT "2) TR
ADE FOR ANOTHER STOCK":PRINT "3) CA
SH SHARES INTO THE BANK":LOCATE 25,
1:PRINT "ENTER SELECTION, OR 0 TO R
ETURN.":
450 GOSUB 1680:IF AS<"0" OR AS>"3" THEN
450 ELSE IF AS="0" THEN RETURN EL
E MD=VAL(AS)
460 CLS:PRINT TAB(16);"TRADING":LOCATE
3,1:PRINT "STOCKS":TAB(13);NAMS(PN)
:FOR Z=1 TO 6:LOCATE 4+Z,1:PRINT US
ING "(PN,Z):NEXT
470 ON MD GOTO 510,900,1020
480 REM **
490 REM ** TRADE WITH PLAYERS **
500 REM **
510 IF NP=1 THEN LOCATE 12,1:PRINT "YOU
ARE THE ONLY PLAYER":FOR TD=1 TO 2
000:NEXT:RETURN ELSE LOCATE 12,1:PR
INT "TRADE WITH THE FOLLOWING PLAYE
RS:":FOR Z=1 TO NP STEP 2
520 IF Z<NP THEN LOCATE INT((Z-1)/2)+1
4,1:PRINT USING "##) &":Z,NAMS(Z)
530 IF (Z-1)/2+1<NP AND Z<NP THEN LOCATE INT
((Z-1)/2)+14,20:PRINT USING "##) &":
Z+1,NAMS(Z+1)
540 NEXT
550 LOCATE 21,1:INPUT "WHICH PLAYER #:"
AS:GOSUB 1690:IF STAT=1 THEN 550 E
LSE TN=VAL(AS):IF TN>NP OR TN=PN OR
TN<1 THEN 550
560 LOCATE 3,25:PRINT NAMS(TN):FOR Z=1
TO 6:LOCATE 4+Z,25:PRINT USING "###
###":P(TN,Z):NEXT
570 LOCATE 23,1:PRINT "1) SELL STOCK TO
NAMS(TN):PRINT "2) BUY STOCK FRO
M NAMS(TN):LOCATE 25,1:PRINT "EN
TER YOUR CHOICE?":
580 GOSUB 1680:IF AS<"1" OR AS>"2" THEN
580 ELSE GOSUB 1710:IF AS="2" THEN
730
590 FOR Z=1 TO 6:IF P(PN,Z)>0 THEN 610
NEXT:LOCATE 25,1:PRINT "YOU DON'T H
AVE ANY STOCK":FOR TD=1 TO 2000:NE
XT:RETURN
610 LOCATE 21,1:PRINT "SELL WHICH STOCK
?":GOSUB 1680:IF AS<"1" OR AS>"6"
THEN 610 ELSE SS=VAL(AS):IF P(PN,SS
)=0 THEN LOCATE 22,1:PRINT "YOU DON
'T HAVE ANY S(S):STOCK":SPC(1
0):GOTO 610
620 LOCATE 21,20:PRINT STRINGS(40,32):
LOCATE 22,1:PRINT "HOW MANY SHARES
OF S(S):INPUT AS:GOSUB 1690:IF
STAT=1 THEN 620 ELSE NOS=INT(VAL(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(S):
:INPUT "SELLING PRICE":AS:GOSUB 169
0:IF STAT=1 THEN 630 ELSE SP=INT(VAL
(AS)):IF SP<S(S)/2 OR SP>S(S)*2
THEN 630
640 IF SP>NOS>PC(TN,1) THEN LOCATE 24,1
:PRINT NAMS(TN):DOESN'T HAVE ENOU
GH MONEY":GOTO 630
650 LOCATE 24,1:PRINT STRINGS(39,32):L
OCATE 24,1:PRINT NAMS(TN):DO YOU
AGREE (Y/N)?":
660 GOSUB 1680:IF AS="Y" THEN 710 ELSE
IF AS<"Y" THEN 660 ELSE LOCATE 23,
1:PRINT SPC(39):LOCATE 24,1:PRINT
SPC(39):
670 LOCATE 23,1:PRINT STRINGS(39,32):L
OCATE 23,1:INPUT "BUYING PRICE":AS:
GOSUB 1690:IF STAT=1 THEN 670 ELSE
SP=INT(VAL(AS)):IF SP<S(S)/2 OR SP
>S(S)*2 THEN 670
680 IF SP>NOS>PC(TN,1) THEN LOCATE 24,1
:PRINT NAMS(TN):DOESN'T HAVE ENOU
GH MONEY":GOTO 670
690 LOCATE 24,1:PRINT STRINGS(39,32):L
OCATE 24,1:PRINT NAMS(PN):DO YOU
AGREE (Y/N)?":

```

```

700 GOSUB 1680:IF AS="N" THEN LOCATE 23
,1:PRINT SPC(39):LOCATE 24,1:PRINT
SPC(39):GOTO 630 ELSE IF AS<"Y"
THEN 690
710 GOSUB 1710:LOCATE 22,10:PRINT "TRAN
SACTION COMPLETE":P(PN,SS)=P(PN,SS)
--NOS:PC(PN,1)=PC(PN,1)+NOS*SP:P(TN,
SS)=P(TN,SS)+NOS:PC(TN,1)=PC(TN,1)
--NOS*SP:FOR TD=1 TO 2000:NEXT
S(SS)=S(SS)+INT((SP-S(SS))/10):RETU
RN
720 FOR Z=1 TO 6:IF P(TN,Z)>0 THEN 750
NEXT:LOCATE 25,1:PRINT "THEY DON'T
HAVE ANY STOCK":FOR TD=1 TO 2000:N
EXT:RETURN
730 LOCATE 21,1:PRINT "BUY WHICH STOCK?
":GOSUB 1680:IF AS<"1" OR AS>"6" T
HEN 730 ELSE BS=VAL(AS):IF P(TN,BS)
=0 THEN LOCATE 22,1:PRINT "THEY DON
'T HAVE ANY S(S(BS)):STOCK":SPC(1
0):GOTO 730
740 LOCATE 22,1:PRINT STRINGS(39,32):L
OCATE 22,1:PRINT "HOW MANY SHARES O
F S(BS):INPUT AS:GOSUB 1690:IF
STAT=1 THEN 740 ELSE NOS=INT(VAL(AS
)):IF NOS>P(TN,BS) THEN LOCATE 23,1
:PRINT "THEY ONLY HAVE":P(TN,BS):
SHARES!":GOTO 740
750 LOCATE 23,1:PRINT STRINGS(39,32):L
OCATE 23,1:INPUT "BUYING PRICE":AS:
GOSUB 1690:IF STAT=1 THEN 770 ELSE
BP=INT(VAL(AS)):IF BP<S(BS)/2 OR B
P>S(BS)*2 THEN 770
760 IF BP>NOS>PC(PN,1) THEN LOCATE 24,1
:PRINT "YOU DON'T HAVE ENOUGH MONEY
":GOTO 760
770 LOCATE 24,1:PRINT STRINGS(39,32):L
OCATE 24,1:PRINT NAMS(TN):DO YOU
AGREE (Y/N)?":
780 GOSUB 1680:IF AS="Y" THEN 850 ELSE
IF AS<"Y" THEN 780 ELSE LOCATE 23,
1:PRINT STRINGS(39,32):
810 LOCATE 23,1:PRINT STRINGS(39,32):L
OCATE 23,1:INPUT "SELLING PRICE":AS:
GOSUB 1690:IF STAT=1 THEN 810 ELSE
BP=INT(VAL(AS)):IF BP<S(BS)/2 OR B
P>S(BS)*2 THEN 810
820 IF BP>NOS>PC(PN,1) THEN LOCATE 24,1
:PRINT NAMS(PN):DOESN'T HAVE ENOU
GH MONEY":GOTO 760
830 LOCATE 24,1:PRINT STRINGS(39,32):L
OCATE 24,1:PRINT NAMS(PN):DO YOU
AGREE (Y/N)?":
840 GOSUB 1680:IF AS="N" THEN LOCATE 23
,1:PRINT STRINGS(39,32):GOTO 770 E
LSE IF AS<"Y" THEN 840
850 GOSUB 1710:LOCATE 22,10:PRINT "TRAN
SACTION COMPLETE":P(PN,BS)=P(PN,BS)
+NOS:PC(PN,1)=PC(PN,1)+NOS*BP:P(TN,
BS)=P(TN,BS)+NOS:PC(TN,1)=PC(TN,1)
+NOS*BP:FOR TD=1 TO 2000:NEXT
S(BS)=S(BS)+INT((BP-S(BS))/5000)+NOS
:RETURN
870 REM **
880 REM ** TRADE FOR ANOTHER STOCK **
890 REM **
900 FOR Z=1 TO 6:IF P(PN,Z)>0 THEN 920
NEXT:LOCATE 25,1:PRINT "YOU DON'T H
AVE ANYTHING TO TRADE":FOR TD=1 TO
2000:NEXT:RETURN
910 LOCATE 3,25:PRINT "CURRENT VALUE":F
OR Z=1 TO 6:LOCATE 4+Z,25:PRINT USI
NG "###) &":Z,NAMS(Z):NEXT
920 LOCATE 21,1:PRINT "TRADE WHICH STOC
K (1-6)?":GOSUB 1680:IF AS<"1" OR
AS>"6" THEN 920 ELSE TS=VAL(AS):PRI
NT S(S(TS)):IF P(PN,TS)=0 THEN LOCATE
22,1:PRINT "DON'T OWN ANY S(S(TS))
:STOCK">RE-ENTER<":GOTO 930
930 LOCATE 22,1:PRINT "FOR WHICH STOCK
(1-6) except 6":TS:GOSUB 1680:IF AS<
"1" OR AS>"6" THEN 940 ELSE FS=VAL(
AS):PRINT S(S(FS))
940 LOCATE 23,1:PRINT "HOW MANY SHARES
OF S(S(TS)):INPUT AS:GOSUB 1690:IF
STAT=1 THEN 950 ELSE NOS=INT(VAL(A
$)):IF P(PN,TS)<NOS THEN LOCATE 24,
1:PRINT "YOU ONLY HAVE":P(PN,TS):
SHARES!":GOTO 950
950 VOS=NOS*S(S(TS)):NOFS=INT(VOS/S(FS)):T
V=NOFS*S(FS):LOCATE 23,1:PRINT SPC(
39):LOCATE 24,1:PRINT SPC(39):LOC
ATE 23,1:PRINT "YOU GOT":NOFS:SHA
RES OF S(S(FS)):LOCATE 24,1:PRINT
FOR:TS:SHARES OF S(S(TS)):
960 LOCATE 25,1:PRINT "S":VOS-TV:HAS
BEEN CONVERTED TO CASH":
970 PC(PN,1)=PC(PN,1)+(VOS-TV):P(PN,TS)
=P(PN,TS)-NOS:P(PN,FS)=P(PN,FS)+NOF
S:FOR Z=1 TO 5000:NEXT:RETURN
980 REM **
990 REM ** CASH IN TO BANK **
1000 REM **
1010 IF BW=W AND BFN=PN THEN RETURN ELSE
BW=W-BFN=PN
1020 FOR Z=1 TO 6:IF P(PN,Z)>0 THEN 1040
NEXT:LOCATE 25,1:PRINT "YOU DON'T H
AVE ANY STOCK":FOR TD=1 TO 2000:NE
XT:RETURN

```

Continued


```

1040 LOCATE 21,1:PRINT "CASH IN WHICH STOCK
LOCK (1-6)? THEN GOSUB 1680:IF AS<"1":SO
R AS>"6":THEN GOSUB 1680:IF AS<"1":SO
PRINT SS(CS):IF P(PN,CS)=0 THEN LOC
ATE 22,1:PRINT "DON'T HAVE ANY";SS
(CS):GOTO 1020
1050 LOCATE 22,1:PRINT "HOW MANY SHARES OF
STAT=1 THEN GOSUB 1690:IF
AS))>P(PN,CS) THEN LOCATE 23
,1:PRINT "YOU ONLY HAVE";P(PN,CS);
SHARES":GOTO 1050
1060 OFR=INT(RND*10)-5:LOCATE 23,1:PRINT
"THE BANK OFFERS YOU $";S(CS)+OFR
1070 LOCATE 24,1:PRINT "DO YOU ACCEPT (
Y/N)?"GOSUB 1680:IF AS<"Y" THEN
N RETURN ELSE IF AS<"Y" THEN
P(PN,CS)=P(PN,CS)+OFR:PC(PN,1)=PC(P
N,1)+(NOS)*(S(CS)+OFR):S(CS)=S(CS)+
OFR:RETURN
1090 REM *****
1100 REM *****
1110 REM *****
1120 PG=1:GOSUB 1640:LOCATE 22,1:PRINT "
DO YOU WISH TO END YOUR TURN (Y/N
)?":GOSUB 1680:IF AS<"Y" THEN RET
URN ELSE IF AS<"Y" THEN 1120 ELSE PN
=PN+1:IF PN<=NP THEN 1180 ELSE PN
=1:W=W+1:IF W>=NP THEN 1580 ELSE GOS
UB 1190:FOR Z=1 TO 6:CHG=INT(RND*20
)/CH(Z,1)+CH(Z,2):SGN(RND*10)-3:S(Z
)=S(Z)+CHG:IF S(Z)<=0 THEN S(Z)=0:F
OR Z=1 TO NP:P(Z,1)=INT(S(Z)/2)+1
IF S(Z)>150 THEN S(Z)=INT(S(Z)/2)+1
EXT:FOR Z=1 TO NP:P(Z,1)=P(Z,1)*2:N
EXT
1150 NEXT Z=1 TO NP:PC(Z,1)=INT(PC(Z,1)-
PC(Z,2)*.005)):IF PC(Z,1)<0 THEN PC
(Z,2)=PC(Z,2)-PC(Z,1):PC(Z,1)=0
1170 NEXT
1180 RETURN
1190 IF INT(W/4)=W/4 THEN FOR Z=1 TO 6:C
H(Z,1)=INT(RND*5)+1:CH(Z,2)=INT(RND
*7)-3:NEXT:RETURN ELSE RETURN
1200 REM *****
1210 REM *****
1220 REM *****
1230 PG=1:GOSUB 1710:LOCATE 22,1:PRINT "
WHICH STOCK (1-6)?":GOSUB 168
0:IF AS<"6" OR AS>"6" THEN SOUND 11
0,1:GOTO 1230 ELSE IF AS="6" THEN R
ETURN ELSE PRINT AS=VAL(AS)
IF S(SN)=0 THEN LOCATE 22,1:PRINT "
THAT STOCK IS BANKRUPT - TRY AGAIN
":FOR TD=1 TO 1000:NEXT:GOTO 1230
1240 LOCATE 22,1:PRINT "HOW MANY SHARES
OF ";SS(SN):INPUT AS:GOSUB 1690:IF
STAT=1 THEN 1250 ELSE NOS=INT(VAL(
AS)):IF NOS=0 THEN RETURN ELSE IF N
OS>P(PN,SN) THEN PRINT "YOU DON'T H
AVE ENOUGH SHARES":GOTO 1250 ELSE
E VALU=S(SN):NOS:CHRG=INT(VALU*.006
)
1260 PRINT "TRANSACTION COMPLETE -- FOR
S":VALU-CHRG:PC(PN,1)=PC(PN,1)+VAL
U-CHRG:P(PN,SN)=P(PN,SN)-NOS
1270 FOR TD=1 TO 1000:NEXT:GOSUB 1500:RE
TURN
1280 REM *****
1290 REM *****
1300 REM *****
1310 PG=0:CLS:LOCATE 1,17:PRINT "LOANS"
1320 LOCATE 5,1:PRINT "1) MAKE A LOAN WI
TH THE BANK:PRINT "2) MAKE A
PAYMENT TO THE BANK:PRINT "3) COMPOUND INTEREST ON A FUTURE LO
AN":LOCATE 11,1:PRINT "ENTER YOUR SE
LECTION:PRINT "OR PRESS 0 TO RETU
RN TO THE MENU":
1330 GOSUB 1680:IF AS<"0" OR AS>"3" THEN
1330 ELSE IF AS="0" THEN RETURN EL
SE ON VAL(AS) GOTO 1340,1400,1460
1340 V=0:FOR Z=1 TO 6:V=V+P(PN,Z)*S(Z)
V=NEXT Z:V=V+PC(PN,1)-PC(PN,2)
IF V<5000 THEN LIM=5000 ELSE LIM=V
IF LIM<=PC(PN,2) THEN LOCATE 13,1:P
RINT "YOU ARE A BAD CREDIT RISK":FO
R TD=1 TO 5000:NEXT:GOTO 1310
1370 LOCATE 14,1:INPUT "HOW MUCH DO YOU
WANT TO BORROW":AS:GOSUB 1690:IF ST
AT=1 THEN 1370 ELSE BOR=INT(VAL(AS)
)
1380 IF BOR+PC(PN,2)>LIM THEN LOCATE 16,
1:PRINT "THAT PUTS YOU OVER YOUR CR
EDIT LIMIT:PRINT "YOUR LIMIT IS":L
IM:FOR TD=1 TO 5000:NEXT:GOTO 1310
1390 PC(PN,2)=PC(PN,2)+BOR:PC(PN,1)=PC(P
N,1)+BOR:PRINT "LOAN IS FINAL
IZED:PRINT "YOU HAVE YOUR MONEY":FOR T
D=1 TO 3000:NEXT:RETURN
1400 IF PC(PN,2)=0 THEN LOCATE 13,1:PRIN
T "YOU DO NOT NEED TO MAKE PAYMENTS
":FOR TD=1 TO 5000:NEXT:GOTO 1310
1410 IF PC(PN,1)=0 THEN LOCATE 13,1:PRIN
T "YOU DO NOT HAVE ANY MONEY":FOR T
D=1 TO 5000:NEXT:GOTO 1310
1420 LOCATE 13,1:PRINT "HOW MUCH WOULD Y
OU LIKE TO PAY?":INPUT "PAYMENT:"A
S:GOSUB 1690:IF STAT=1 THEN 1420 EL
SE PAY=VAL(AS)

```

```

1430 IF PAY>PC(PN,1) THEN LOCATE 16,1:PR
INT "YOU ONLY HAVE $";PC(PN,1):GOTO
1420
1440 IF PAY>PC(PN,2) THEN LOCATE 16,1:PR
INT "YOU ONLY OWED $";PC(PN,2):GOTO
1420
1450 HANK YOU"PC(PN,1)=PC(PN,1):PC(PN,2
)=PC(PN,2):FOR TD=1 TO 5000:NEXT:
RETURN
1460 LOCATE 16,1:PRINT "THANK YOU":PC(PN
,1)=PC(PN,1):PAY:PC(PN,2)=PC(PN,2)-
PAY:FOR TD=1 TO 5000:NEXT:RETURN
1470 LOCATE 13,1:INPUT "SIZE OF LOAN":AS
GOSUB 1690:IF STAT=1 THEN 1460 EL
SE LS=VAL(AS):LOCATE 15,1:PRINT "THE
WEEKLY INTEREST ON THAT AMOUNT IS:
":PRINT "":INT(LS*.05):LOCATE 17,1
:PRESS [ENTER] TO CONTINUE":GOSUB 9
000:RETURN
1470 REM *****
1480 REM *****
1490 REM *****
1500 PG=0:CLS:LOCATE 1,17:PRINT "LOANS"
UB 1750:FOR Z=0 TO 5:LOCATE 7+Z*2,1
3:PRINT USING "####";S(Z+1):AVE=P
(PN,Z+1)*S(Z+1)
1510 LOCATE 7+Z*2,22:PRINT USING "####
#"SS####":P(PN,Z+1):AVE:NEXT
1520 LOCATE 21,1:PRINT "ON HAND:PRINT
USING "#####"PC(PN,1):PRINT
TAB(21):INVESTED:INVS=0:FOR Z=1
TO 6:INV=INV+P(PN,Z)*S(Z):NEXT:PRIN
T USING "#####"INV:
1530 LOCATE 22,3:PRINT "LOANS:PRINT
USING "#####"PC(PN,2):PRINT TA
B(24):TOTAL:TOT=PC(PN,1)+INV-PC
(PN,2):PRINT USING "#####"TOT:
LOCATE 25,1:PRINT "PRESS [ENTER] TO
CONTINUE":GOSUB 1680:RETURN
1550 REM *****
1560 REM *****
1570 REM *****
1580 CLS:PRINT "PLAYER: TAB(17):"NET WOR
TH":TAB(30):PROFIT:PRINT:FOR Z=1
TO NP:NET(Z)=0:FOR ZZ=1 TO 6:NET(Z)
=NET(Z)+P(Z,ZZ)*S(Z):NEXT:NET(Z)=N
ET(Z)+PC(Z,1)-PC(Z,2):PRINT Z:");
NAMS(Z):TAB(20):NET(Z):TAB(30):NET(
Z)-5000:NEXT
1590 WIN=1:FOR Z=1 TO NP:IF NET(Z)>NET(W
IN) THEN WIN=Z
1600 NEXT:LOCATE 15,1:PRINT "THE WINNER
IS":NAMS(WIN):FOR Z=1 TO NP:IF Z=
WIN THEN 1610 ELSE PRINT "AND":NA
MS(Z);
1610 NEXT
1620 PRINT "WITH:PRINT "$:NET(WIN)-50
00:NET PROFIT -- CONGRATULATIONS"
1630 LOCATE 20,1:PRINT "WOULD YOU LIKE T
O PLAY AGAIN (Y/N)?"GOSUB 16
80:IF AS="Y" THEN RUN ELSE IF AS<"
N" THEN 1630 ELSE CLS:PRINT "BYE
":BYE -- SEE YOU AT THE MARKET":END
1640 FOR Z=21 TO 25:LOCATE Z,1:PRINT SPC
(39):NEXT:RETURN
1650 REM *****
1660 REM *****
1670 REM *****
1680 AS=INKEY$:IF AS=" " THEN 1680 ELSE R
ETURN
1690 IF AS=" " THEN STAT=1:RETURN ELSE FO
R Z=1 TO LEN(AS):IF ASC(AS)<48 OR A
SC(AS)>57 THEN STAT=1:RETURN
NEXT:STAT=0:RETURN
1700 FOR Z=21 TO 25:LOCATE Z,1:PRINT SPC
(39):NEXT:RETURN
1710 REM *****
1720 REM *****
1730 REM *****
1740 REM *****
1750 CLS:LOCATE 1,13:PRINT CHR$(186):NGS
:CHR$(186):PRINT TAB(13):CHR$(186):
SPC(15):CHR$(186):PRINT SPC(12):CHR
$(186):CLS:CHR$(186):PRINT STRINGS(
12,205):CHR$(202):STRINGS(6,205):CH
R$(209):STRINGS(8,205):CHR$(202):
PRINT STRINGS(11,205):PRINT LAS:TA
B(20):CHR$(179):LBS:FOR Z=1 TO 6:P
RINT STRINGS(19,196):CHR$(197):STRI
NGS(20,196):PRINT RIGHTS(STRS(Z),1
):");SS(Z):TAB(20):CHR$(179):N
EXT:PRINT STRINGS(19,205):CHR$(207
):STRINGS(20,205):RETURN
1770 LOCATE 2,1:PRINT NAMS(PN):SPC(8-LEN
(NAMS(PN))):LOCATE 2,30:PRINT "WEE
K:W:LOCATE 19,1:PRINT "HIGH:PRINT STRIN
G$(40,223):PRINT
1780 PRINT "MENU: TAB(11):"B)UY":TAB(25
):S)ELL":SPC(10):PRINT "SELECT
TRADE":TAB(25):"LOAN":PRINT "OPTI
ON N)EXT":TAB(25):"P)ORTFOLIO":;
RETURN
1790 REM *****
1800 REM *****
1810 REM *****
1820 DATA "WALL STREET STOCK EXCHA
NGE:PORTFOLIO:STOCK:VALUE
PRICE / SHARES:STOCK:VALUE
SHARES AND VALUE:US STEEL,PAN
AM,FORD,SANYO,XEROX,AT&T

```



```

100 REM *** STOCK MARKET ***
110 REM BY BRIAN LEE
120 REM AND THE HCM STAFF
130 REM HOME COMPUTER MAGAZINE
140 REM VERSION 4.4.1
150 REM TI EXTENDED BASIC
160
170 CALL CLEAR :: DISPLAY AT(12,8):"STO
180 CK MARKET" :: DISPLAY AT(24,2):"PRE
190 SS ANY KEY TO CONTINUE" :: GOSUB 14
200 OPTION BASE 1 :: DIM S(6),SS(6),P(1
210 0,6),PC(10,2),CH(6,2):: RANDOMIZE
220 CALL CHAR(33,"000000FFFF",95,18181
230 8181818181818181):: CALL SCREEN(2):FO
240 R Z=1 TO 8 :: CALL COLOR(Z,16,5)::
250 NEXT Z
260 RESTORE 1490 :: FOR Z=1 TO 6 :: REA
270 D SS(Z):: CH(Z,1)=INT(RND*4+1):: CH
280 (Z,2)=INT(RND*7)-3 :: NEXT Z
290 CALL CLEAR :: DISPLAY AT(1,1):"HOW
300 MANY PLAYERS (1-10)?" :: ACCEPT AT
310 (1,25)SIZE(-2)VALIDATE(DIGIT):NP ::
320 IF NP<1 OR NP>10 THEN 230
330 FOR Z=1 TO NP :: DISPLAY AT(Z+3,1):
340 "PLAYER #:"STR$(Z):"S NAME:"AC
350 CEPT AT(Z+3,19)SIZE(8):NAMS(Z):: NE
360 XT Z
370 FOR Z=1 TO NP :: IF NAMS(Z)="" THEN
380 NAMS(Z)="PLAYER"&STR$(Z)
390 NEXT Z
400 DISPLAY AT(16,1):"HOW MANY WEEKS (2
410 OR MORE)?" :: ACCEPT AT(17,1)V
420 ALIDATE(DIGIT)SIZE(-3):AS :: IF AS<
430 "2" THEN 270 ELSE NW=VAL(AS)
440 CALL CLEAR :: FOR Z=1 TO 6 :: S(Z)=
450 INT(RND*40+10):: FOR Z1=1 TO NP ::
460 P(Z1,Z)=0 :: NEXT Z1 :: NEXT Z
470 FOR Z=1 TO NP :: PC(Z,1)=5000 :: PC
480 (Z,2)=0 :: NEXT Z :: PN=1 :: W=1 ::
490 GOSUB 1460 :: IF K=66 OR K=83 THEN
500 320 ELSE IF K=84 THEN 420 ELSE IF K
510 =76 THEN 970 ELSE IF K=78 THEN 1150
520 IF K=80 THEN 1290 ELSE CALL SOUND(1
530 ,110,0):: GOTO 300
540 M=K :: GOSUB 1470 :: DISPLAY AT(22,
550 1):"WHICH STOCK (1-6)?" :: GOSUB 14
560 60 :: IF K<48 OR K>54 THEN 320 ELSE
570 SN=K-48
580 IF SN=0 THEN GOSUB 1430 :: GOTO 300
590 IF S(SN)=0 THEN DISPLAY AT(23,1):"T
600 HAT STOCK IS BANKRUPT" :: CALL D(50
610 0):: GOTO 320
620 DISPLAY AT(22,20):SS(SN):"HOW MANY
630 SHARES?" :: ACCEPT AT(23,17)VALIDAT
640 E(DIGIT):Z$ :: IF Z$="" THEN 350 EL
650 SE NOS=VAL(Z$)
660 IF NOS=0 THEN GOSUB 1430 :: GOTO 30
670 0
680 IF M=83 THEN 400
690 COST=S(SN)*NOS :: CHRG=INT(COST*.00
700 6) :: IF COST+CHRG>PC(PN,1)THEN DISP
710 LAY AT(24,1):"NOT ENOUGH MONEY" ::
720 CALL D(5000):: GOTO 320
730 DISPLAY AT(24,1):"PURCHASE COMPLETE
740 :: COST+CHRG :: PC(PN,1)=PC(PN,1)-C
750 OST-CHRG :: P(PN,SN)=P(PN,SN)+NOS ::
760 CALL D(5000):: GOTO 1290
770 IF NOS>P(PN,SN)THEN DISPLAY AT(24,1
780 ): "NOT ENOUGH SHARES" :: CALL D(500
790 0):: K=M :: GOTO 320
800 DISPLAY AT(24,1):"STOCK IS SOLD:"N
810 OS*S(SN):: PC(PN,1)=PC(PN,1)+NOS*S(
820 SN):: P(PN,SN)=P(PN,SN)-NOS :: CALL
830 D(7500):: GOTO 1290
840 CALL CLEAR :: DISPLAY AT(1,1):"1) T
850 RADE WITH OTHER PLAYERS" :: "2) TRAD
860 E FOR ANOTHER STOCK" :: "3) SELL TO
870 THE BANK"
880 DISPLAY AT(7,1):"ENTER YOUR CHOICE"
890 :: OR PRESS 0 TO RETURN"
900 GOSUB 1460
910 IF K<48 OR K>51 THEN 440 ELSE IF K=
920 48 THEN GOSUB 1400 :: GOTO 300 ELSE
930 ON K-48 GOTO 460,790,880
940 IF NP=1 THEN DISPLAY AT(10,1):"YOU'
950 RE THE ONLY PLAYER" :: CALL D(750):
960 GOTO 420
970 CALL CLEAR :: GOSUB 960 :: DISPLAY
980 AT(10,1):"TRADE WITH THESE PLAYERS:
990
1000 X=11 :: Y=1 :: FOR Z=1 TO NP :: IF
1010 Z<>PN THEN DISPLAY AT(X,Y):Z:NAMS(Z)
1020 :: X=X+1 :: IF X>15 THEN X=11 :: Y
1030 =15
1040 NEXT Z
1050 DISPLAY AT(17,1):"WHICH PLAYER?" ::
1060 ACCEPT AT(17,15)VALIDATE(DIGIT)SI
1070 Z E(2):AS :: IF AS="" THEN 500 ELSE T
1080 N=VAL(AS) :: IF PN=0 THEN 420
1090 IF TN>NP OR TN=PN THEN 500
1100 FOR Z=1 TO 6 :: DISPLAY AT(Z+2,12):
1110 USING "##"#### "S(Z),P(PN
1120 ,Z),P(TN,Z):: NEXT Z
1130 DISPLAY AT(2,1):"STOCK VAL PL#
1140 :PN:TAB(23):PL#::TN
    
```

```

540 DISPLAY AT(19,1):"1) SELL TO :NAMS
550 (TN):"2) BUY FROM :NAMS(TN):"YOUR
560 CHOICE?" :: GOSUB 1460 :: IF K<49 O
570 R K>50 THEN 540
580 IF K=50 THEN 670
590 CALL HCHAR(19,1,32,192):: DISPLAY A
600 T(19,1):"SELL WHICH STOCK?"
610 GOSUB 1460 :: IF K<48 OR K>54 THEN
620 570 ELSE SN=K-48 :: IF SN=0 THEN 42
630 0 ELSE IF P(PN,SN)=0 OR S(SN)=0 THE
640 N 570
650 DISPLAY AT(19,19):SS(SN):"HOW MANY
660 SHARES?"
670 ACCEPT AT(20,19)VALIDATE(DIGIT):Z$
680 :: IF Z$="" THEN 590 ELSE NOS=VAL(Z
690 $):: IF NOS>P(PN,SN)THEN CALL SOUND
700 (1,110,0):: GOTO 590
710 IF NOS=0 THEN 420
720 DISPLAY AT(21,1):"SELLING PRICE?" :
730 A :: ACCEPT AT(21,16)VALIDATE(DIGIT):A
740 $ :: IF AS="" THEN 610 ELSE SP=VAL(
750 AS)
760 IF SP<S(SN)/2 OR SP>S(SN)*2 THEN 61
770 0
780 IF SP*NOS>PC(TN,1)THEN DISPLAY AT(2
790 2,1):NAMS(TN):"CAN'T AFFORD IT" ::
800 CALL D(5000):: GOTO 560
810 DISPLAY AT(22,1):NAMS(TN):"DO YOU
820 AGREE?"(Y/N)?" :: ACCEPT AT(23,7)V
830 ALIDATE("YN")SIZE(1):AG$ :: IF AG$=
840 "N" THEN 560
850 S(SN)=S(SN)+INT((SP-S(SN))/10)
860 PC(PN,1)=PC(PN,1)+NOS*SP :: PC(TN,1
870 )=PC(TN,1)-NOS*SP :: P(PN,SN)=P(PN,
880 SN)-NOS :: P(TN,SN)=P(TN,SN)+NOS ::
890 GOTO 420
900 CALL HCHAR(19,1,32,192):: DISPLAY A
910 T(19,1):"BUY WHICH STOCK?"
920 GOSUB 1460 :: IF K<48 OR K>54 THEN
930 680 ELSE SN=K-48 :: IF SN=0 THEN 42
940 0 ELSE IF P(TN,SN)=0 OR S(SN)=0 THE
950 N 680
960 DISPLAY AT(19,18):SS(SN):"HOW MANY
970 SHARES?"
980 ACCEPT AT(20,19)VALIDATE(DIGIT):AS
990 :: IF AS="" THEN 700 ELSE NOS=VAL(A
1000 $):: IF NOS>P(TN,SN)THEN CALL SOUND
1010 (1,110,0):: GOTO 700
1020 IF NOS=0 THEN 420
1030 DISPLAY AT(21,1):"SELLING PRICE?" :
1040 A :: ACCEPT AT(21,16)VALIDATE(DIGIT):A
1050 $ :: IF AS="" THEN 720 ELSE SP=VAL(
1060 AS)
1070 IF SP=0 THEN 420
1080 IF SP<S(SN)/2 OR SP>S(SN)*2 THEN 72
1090 0
1100 IF SP*NOS>PC(PN,1)THEN DISPLAY AT(2
1110 2,1):NAMS(PN):"CAN'T AFFORD IT" ::
1120 CALL D(5000):: GOTO 670
1130 DISPLAY AT(22,1):NAMS(PN):"DO YOU
1140 AGREE?"(Y/N)?" :: ACCEPT AT(23,7)V
1150 ALIDATE("YN")SIZE(1):AG$ :: IF AG$=
1160 "N" THEN 670
1170 S(SN)=S(SN)+INT((SP-S(SN))/10)
1180 PC(TN,1)=PC(TN,1)+NOS*SP :: PC(PN,1
1190 )=PC(PN,1)-NOS*SP :: P(TN,SN)=P(TN,
1200 SN)-NOS :: P(PN,SN)=P(PN,SN)+NOS ::
1210 GOTO 420
1220 CALL CLEAR :: GOSUB 960
1230 DISPLAY AT(15,1):"TRADE WHICH STOCK
1240 ?" :: GOSUB 1460 :: IF K<48 OR K>54
1250 THEN 790 ELSE IF K=48 THEN 420 ELS
1260 E SN=K-48
1270 IF P(PN,SN)=0 THEN 800
1280 DISPLAY AT(15,20):SS(SN):"FOR WHICH
1290 STOCK?" :: GOSUB 1460 :: IF K<48 O
1300 R K>54 OR K=48=SN THEN 820 ELSE IF
1310 K=48 THEN 420 ELSE TS=K-48
1320 DISPLAY AT(16,18):SS(TS):"HOW MAN
1330 Y SHARES OF :SS(SN):: ACCEPT AT(19
1340 ,1)VALIDATE(DIGIT):AS :: IF AS="" T
1350 HEN 830 ELSE NOS=VAL(AS)
1360 IF NOS>P(PN,SN)THEN 830
1370 V=S(SN)*NOS :: NNS=INT(V/S(TS)):: T
1380 TB=INT((V/S(TS)-NNS)*S(TS))
1390 DISPLAY AT(20,1):"TRADE :NOS: SHAR
1400 ES OF :SS(SN): FOR :NNS: SHARES
1410 OF :SS(TS):"TRANSFER $:TTB: TO YO
1420 UR ACCOUNT"
1430 P(PN,SN)=P(PN,SN)-NOS :: P(PN,TS)=P
1440 (PN,TS)+NNS :: PC(PN,1)=PC(PN,1)+TT
1450 B :: CALL D(7500):: GOTO 420
1460 CALL CLEAR :: GOSUB 960
1470 DISPLAY AT(11,1):"CASH IN WHICH STO
1480 CK?" :: GOSUB 1460 :: IF K<48 OR K>
1490 54 THEN 880 ELSE IF K=48 THEN 420 E
1500 LSE SN=K-48
1510 IF P(PN,SN)=0 THEN 890
1520 DISPLAY AT(11,22):SS(SN):"HOW MAN
1530 Y SHARES?" :: ACCEPT AT(13,18)VALID
1540 ATE(DIGIT):AS :: IF AS="" THEN 910
1550 ELSE NOS=VAL(AS)
1560 IF NOS>P(PN,SN)THEN 910
1570 OFR=INT(RND*10)-5 :: DISPLAY AT(15,
1580 22):"THE BANK OFFERS YOU $:S(SN)+O
1590 FR
1600 DISPLAY AT(17,1):"DO YOU AGREE (Y/N
1610 )?" :: GOSUB 1460 :: IF K=78 THEN 4
1620 20 ELSE IF K<>89 THEN 940
    
```

Continued

MARKET MADNESS

Continued

TI-99/4A

```

950 P(PN,1)+NOS*(S(S(SN))+OFR):: GOTO 420
960 C(PN,1) TO 6:: DISPLAY AT(Z+2,1): U
SING 1480 Z,S(Z),S(Z),P(PN,Z):: NE
XT Z:: RETURN
970 CALL CLEAR:: DISPLAY AT(1,1): "1"
AKE OUT A LOAN:: "2) PAY BACK A T
AN": "3) COMPOUND INTEREST ON A":
FUTURE LOAN"
980 DISPLAY AT(8,1): "ENTER YOUR CHOICE"
"OR PRESS 0 TO RETURN TO MENU"
990 GOSUB 1460:: IF K<48 OR K>51 THEN
990 ELSE IF K=48 THEN GOSUB 1400::
GOSUB 1430:: GOTO 300 ELSE ON K-4
8 GOTO 1000,1040,1110
1000 V=0:: FOR Z=1 TO 6:: V=V+P(PN,Z)*
S(Z):: NEXT Z:: LIM=MAX(5000-PC(PN
,2),V+PC(PN,1)-PC(PN,2))
1010 DISPLAY AT(11,1): "HOW MUCH MONEY DO
YOU WANT TO BORROW?":: ACCEPT AT
(12,12)VALIDATE(DIGIT):A$:: IF A$=
" THEN 1010 ELSE BOR=VAL(A$)
1020 IF BOR+PC(PN,2)>LIM THEN DISPLAY AT
(14,1): "YOU'RE OVER YOUR LIMIT": "MA
XIMUM LOAN AMOUNT IS": LIM:: CALL
(500):: GOTO 970
1030 DISPLAY AT(14,1): "LOAN IS OK":: PC
(PN,1)=PC(PN,1)+BOR:: PC(PN,2)=PC
(PN,2)+BOR:: CALL D(250):: GOSUB 14
00:: GOTO 300
1040 IF PC(PN,1)=0 THEN DISPLAY AT(11,1)
:"YOU DON'T HAVE ANY MONEY":: CALL
D(750):: GOTO 970
1050 IF PC(PN,2)=0 THEN DISPLAY AT(11,1)
:"YOU DON'T HAVE ANY DEBTS":: CALL
D(750):: GOTO 970
1060 DISPLAY AT(11,1): "AMOUNT OF PAYMENT
?":: ACCEPT AT(11,20)VALIDATE(DIGI
T):A$:: IF A$= " THEN 1060 ELSE LP
=VAL(A$):: IF LP<0 THEN 1060
1070 IF LP>0 THEN 970
1080 IF LP=PC(PN,1) THEN DISPLAY AT(13,1)
:"YOU DON'T HAVE THAT MUCH":: CALL
D(500):: GOTO 970
1090 DISPLAY AT(13,1): "PAYMENT RECEIVED"
:"THANK YOU":: PC(PN,1)=PC(PN,1)-M
IN(LP,PC(PN,2)):: PC(PN,2)=PC(PN,2)
-MIN(LP,PC(PN,2))
1100 CALL D(500):: GOSUB 1400:: GOTO 30
0
1110 DISPLAY AT(11,1): "AMOUNT FOR LOAN C
ALCULATION?":: ACCEPT AT(12,1)VALI
DATE(DIGIT):A$:: IF A$= " THEN 111
0 ELSE BOR=VAL(A$)
1120 TI=0:: B=BOR:: FOR Z=W TO NW:: I
=INT(B*.005):: B=B+I:: TI=TI+I::
NEXT Z
1130 DISPLAY AT(14,1): "WEEKS REMAINING:
":NW-W: "LOAN AMOUNT:":TAB(20):BOR
:"INTEREST:":TAB(20):TI: "WEEKLY CH
ARGE:":INT(BOR*.005)
1140 DISPLAY AT(18,1): "LOAN BALANCE:
":B: "PRESS ANY KEY TO CONTINUE"
:: GOSUB 1460:: GOSUB 1400:: GOT
O 300
1150 GOSUB 1470:: DISPLAY AT(22,1): "WAN
T TO END YOUR TURN(Y/N)?
1160 GOSUB 1460:: IF K=78 THEN GOSUB 14
50:: GOTO 300 ELSE IF K<>89 THEN 1
160
PN=PN+1:: IF PN>NP THEN PN=1:: W=
W+1 ELSE GOSUB 1410:: GOTO 300
1170 IF W>NW THEN 1330
FOR Z=1 TO NP:: PC(Z,1)=PC(Z,1)-IN
T(PC(Z,2)*.005):: IF PC(Z,1)<0 THEN
PC(Z,2)=PC(Z,2)-PC(Z,1):: PC(Z,1)=
0

```

```

12000 NEXT Z : : FOR Z=1 TO 6 : : C=INT(RND*20)/CH(Z,1)+CH(Z,2))*SGN(RND*10)-3
12100 IF S(Z)<=150 THEN 12300
12200 S(Z)=INT(S(Z)/2+1) : : FOR ZZ=1 TO NP : : P(ZZ)=P(Z,Z)*2 : : NEXT ZZ
12300 IF S(Z)>0 THEN 12500 ELSE S(Z)=0
12400 FOR ZZ=1 TO NP : : P(ZZ,Z)=0 : : NEXT Z
12500 NEXT Z
12600 IF INT(W/4)<>W/4 THEN 12800
12700 FOR Z=1 TO 6 : : CH(Z,1)=INT(RND*4+1) : : CH(Z,2)=INT(RND*7)-3 : : NEXT Z
12800 GOSUB 14100 : : GOTO 3000
12900 TOT=0 : : DISPLAY AT(4,1) : "STOCK V
13000 ALUE SHARES WORTH" : : FOR Z=1 TO 6 : : S(Z)=S(Z),P(PN,Z),P(PN,Z))*S(Z) : : NEXT Z
13100 DISPLAY AT(Z*2+4,1) : USING "#.#" : "TOTAL=TOT+PC(PN,1)-PC(PN,2) : : DISPL
13200 AY AT(22,1) : "CASH : " : PC(PN,1) : TAB(16) : "INV : " : TOT : "LOAN : " : PC(PN,2) : TAB(16) : "NET : " : TOTAL
13300 DISPLAY AT(24,1) : "PRESS ANY KEY TO CONTINUE" : : GOSUB 14600 : : GOSUB 14700 : : GOTO 3000
13400 CALL CLEAR
13500 WIN=1 : : HV=0 : : FOR Z=1 TO NP : : T
13600 T=0 : : FOR ZZ=1 TO 6 : : TT=TT+(P(Z,ZZ)*S(ZZ)) : : NEXT ZZ : : TT=TT+PC(Z,1)-PC(Z,2)-5000
13700 IF TT>HV THEN HV=TT : WIN=Z
13800 DISPLAY AT(Z+3,1) : NAMS(Z) : "GAINED S : TT : NEXT Z
13900 DISPLAY AT(20,1) : "THE WINNER IS : " : NAMS(WIN)
14000 DISPLAY AT(24,1) : "PLAY AGAIN(Y/N)?" : : GOSUB 14600 : : IF K=78 THEN END
14100 ELSE IF K<>89 THEN 13800
14200 FOR Z=1 TO NP : : FOR ZZ=1 TO 6 : : P(Z,ZZ)=0 : : S(ZZ)=0 : : NEXT ZZ : : P(C(Z,1)=5000 : : PC(Z,2)=0 : : NEXT Z : : W=1 : : PN=1 : : GOTO 2800
14300 CALL CLEAR : : FOR Z=3 TO 21 STEP 2 : : CALL HCHAR(Z,1,33,32) : : NEXT Z : : DISPLAY AT(1,8) : "WALL STREET" : : CALL VCHAR(1,31,95,96)
14400 DISPLAY AT(4,1) : "STOCK VALUE SHARES : : FOR Z=1 TO 6 : : DISP
14500 LAY AT(Z*2+4,1) : USING 1480 : Z, S$(Z), S(Z), P(PN,Z) : : NEXT Z
14600 HI=0 : : LO=200 : : FOR Z=1 TO 6 : : H
14700 I=MAX(HI, S(Z)) : : LO=MIN(LO, S(Z)) : : NEXT Z : : DISPLAY AT(20,1) : "HIGH : " : HI : TAB(18) : "LOW : " : LO
14800 DISPLAY AT(22,1) : MENU : B)UY S)ELL : " : TRADE L)OAN : " : N)EXT P)ORTFOLIO : "
14900 DISPLAY AT(2,1) : NAMS(PN) : TAB(19) : "WEEK : " : W
15000 RETURN
15100 CALL KEY(0,K,ST) : : IF ST<>1 THEN 14600 ELSE CALL SOUND(50,3000,5) : : RET
15200 DISPLAY AT(22,1) : RPT$( " ", 84) : : RE
15300 TURN
15400 IMAGE # ##### #
15500 DATA US STEEL,PAN AM,FORD,SANYO,XEROX,AT&T
15600 SUB D(TDV)
15700 FOR TD=1 TO TDV : : NEXT TD : : SUBEN

```

HCM

STADIUM JUMPING

APPLE II Family

```

100 REM ** STADIUM JUMPING **
110 REM ** ** ** **
120 REM ** ** **
130 REM BY KENT & KATHY GEMMEL
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM VERSION 4.4.1
170 REM APPLE II FAMILY APPLESOFT
180 REM
190 REM
200 IF PEEK (104) < 64 THEN POKE 1
04,64: POKE 103,1: POKE 16384,0: PR
INT : CHR$ (4): "RUN STADIUM"
210 TEXT : NORMAL: NOTRACE
220 HCHAR = 2048: GCHAR = 2108: SOUND = 2
118:SPKR = 2130:RESTR = 2138: DIM
T(11,5):SPED(1) = 200:SPED(2) = 120
=:SPED(3) = 80:SPED(4) = 30:SPED(5)
= 1
230 GNUM(1) = 4:GNUM(2) = 7:GNUM(3) = 1
1:CRSH(1) = 11:CRSH(2) = 13:CRSH(3)
= 12:CRSH(4) = 14
240 IF PEEK (2048) > 32 THEN FOR K
NEXT TO 2392: < READ P: POKE K,P:

```

[illegible]

Continued

Continued

APPLE II Family

[illegible]

```

810 FOR I = 1 TO 16: FOR J = 1 TO 19: READ X,Y: CALL HC
820 FOR I = 1 TO 3: NEXT J = 1 TO 5: REA
830 DATA GT(1,1),6,10,9,18,12,18,20,20,22,
20
840 DATA 10,16,11,16,4,12,5,12,13,4,13,
5,13,6,21,18
850 DATA 10,17,11,17,4,11,5,11,12,4,12,
5,12,6,21,19
860 DATA 12,17,3,11,13,3,22,19
870 DATA 10,11,18,18,7,4,5,10,10,3,14,1
5,4,6
880 GOTO 1050
890 CALL RESTR,970: FOR I = 1 TO 10: RE
AD X,Y: CALL HCHAR,X,Y + 4,5: NEXT
900 FOR I = 1 TO 9: READ X,Y: CALL HCHA
R,X,Y + 4,6: NEXT
910 FOR I = 0 TO 1: CALL HCHAR,11 + I,3
CHAR,11,32 + I,4: CALL HCHAR,12,32
+ I,4: NEXT I
920 CALL HCHAR,10,14,7: CALL HCHAR,13,1
1,7: CALL HCHAR,12,12,8: CALL HCHAR
,11,13,8
930 FOR I = 1 TO 16: READ X,Y: CALL HCH
AR,X,Y + 4,0: NEXT
940 FOR I = 1 TO 15: READ X,Y: CALL HCH
AR,X,Y + 4,1: NEXT
950 FOR I = 16 TO 22: READ X,Y: CALL HC
HAR,X,Y + 4,1: NEXT
960 FOR I = 1 TO 6: FOR J = 1 TO 5: REA
D X:GT(1,1),6,11,3,21,6,21,14,18,16,1
8,19,13,23,13,20,20,22,20
980 DATA 4,11,5,11,4,21,5,21,15,18,20,1
3,21,13,22,13,21,20
990 DATA 11,7,10,8,15,16,13,28,13,29,4,
23,5,23,4,13,5,13,20,11,21,11,22,11
,21,18,11,6,15,17,7,9,8
1000 DATA 4,12,5,12,20,12,21,12,22,12,21,1
9,12,7,11,8,10,9
1010 DATA 13,6,16,17,13,30,3,22,3,12,23,
12,22,19
1020 DATA 10,12,7,9,6,15,15,18,18,7
1030 DATA 11,12,28,29,1,4,5,21,21,3
1040 DATA 4,5,11,11,3,20,22,13,13,7
1050 FS = 0:GT = 0:SPD = 2:GOSUB 1820:
GOSUB 1780:A = 5:X = 1:Y = 3:HO = 1
1060 CALL HCHAR,X,Y + 4,HO: POKE - 1638
4,0
1070 IF PEEK ( - 16384) < > 203 THEN 1
070
1080 CALL HCHAR,X,Y + 4,25
1090 CALL GCHAR,X,Y + 4:M = PEEK (0): I
F M = 160 THEN M = 25
1100 IF M > 1 AND M < 11 THEN = 1270
1110 CALL HCHAR,X,Y + 4,HO:K = PEEK ( -
16384): POKE - 16368,0: FOR LP =
1 TO SPD(SPD): NEXT LP
1120 IF K > 176 AND K < 182 THEN SPD = K
- 176
1130 IF K = 160 OR (KJS = "J" AND PEEK
(- 16287) > 127) THEN CALL HCHAR,
X,Y + 4,25: GOTO 1430
1140 IF K = 136 OR (KJS = "J" AND PDL (
0) < 80) THEN A = A + 1: IF A = 9 T
HEN A = 1
1150 IF K = 149 OR (KJS = "J" AND PDL (
0) < 170) THEN A = A - 1: IF A = 0
THEN A = 8
1160 CALL HCHAR,X,Y + 4,M: CALL SOUND,8
1170 ON A GOSUB 1190,1200,1210,1220,1230
,1240,1250,1260
1180 GOTO 1090
1190 X = X - 1:HO = 11: RETURN
1200 X = X - 1:Y = Y - 1:HO = 14: RETURN
1210 Y = Y - 1:HO = 12: RETURN
1220 X = X + 1:Y = Y - 1:HO = 13: RETURN
1230 X = X + 1:HO = 11: RETURN
1240 Y = Y + 1:X = X + 1:HO = 14: RETURN
1250 Y = Y + 1:HO = 12: RETURN
1260 Y = Y + 1:X = X - 1:HO = 13: RETURN
1270 CALL HCHAR,X,Y + 4,M
1280 CALL GCHAR,X,Y + 4:M = PEEK (0): I
F M = 160 THEN M = 25
1290 FOR C1 = 1 TO 6: FOR C2 = 1 TO 4: C
ALL HCHAR,X,Y + 4,CRSH(C2): CALL S
OUND,126: NEXT Y + C2,C1
1300 IF X < 2 OR Y < 2 OR X > 23 OR Y >
31 THEN 1400
1310 IF INT (RND (1) * 15) + 1 = 1 THE
N 1400
1320 FS = FS + 4: GOSUB 1780
1330 IF (X < 23) * (X > 19) * (Y = 20) A
ND GT > GNUM(L) - 1 THEN 1720
1340 CALL HCHAR,X,Y + 4,M
1350 ON A GOSUB 1190,1200,1210,1220,1230
,1240,1250,1260
1360 CALL GCHAR,X,Y + 4:M = PEEK (0): I
F M = 160 THEN M = 25
1370 IF X < 2 OR Y < 2 OR X > 23 OR Y >
31 THEN 1400

```

Continued

STADIUM JUMPING *Continued*

```

1380 IF M < 0 AND M < 1 AND M < 1
1390 CALL HCHAR,X,Y + 4,M: GOTO 1110
1400 FS = 4: PRINT "RIDER HAS FALLEN 14: HE
LIMITATION 11: VTAB 18: HTAB 9: PR
INT "YOU RECEIVED FS" FAULTS 11"
1410 CALL SO,127: VTAB 24: FOR K = 1 TO
17: CALL SO,2: PRINT: FOR KK = 1 TO T
O 25: NEXT KK,K
1420 GOTO 1750
1430 CALL HCHAR,X,Y + 4,HO
1440 IF M = 0 THEN 1640
1450 IF M = 1 THEN 1540
1460 FOR I = 1 TO 4: CALL HCHAR,X,Y + 4,
ON A GOSUB 1190,1200,1210,1220,1230
1470 CALL GCHAR,X,Y + 4:M = PEEK (0): I
F M = 160 THEN M = 25
1480 IF (X = 21) * (Y = 20) AND GT = >
GNUM(L) = 1: THEN 1720
1490 GJ = GT + 1: IF GT(GJ,1) < X AND
GT(GJ,2) < Y AND GT(GJ,3) < X AND
Y = 5) THEN GT = GJ
1510 GOSUB 1820: IF X < 2 OR Y < 2 OR X
V 23 OR Y > 31 THEN 1400
1520 IF M < 0 AND M < 1 AND M < 1
25 THEN 1270
1530 CALL HCHAR,X,Y + 4,HO: FOR LP = 1 TO
O SPED(SPD) / 2: NEXT LP,I: GOTO 12
70
1540 CALL HCHAR,X,Y + 4,HO: FOR I = 1 TO
4: CALL HCHAR,X,Y + 4,M
1550 ON A GOSUB 1190,1200,1210,1220,1230
1560 CALL GCHAR,X,Y + 4:M = PEEK (0): I
F M = 160 THEN M = 25
1570 IF (X = 23) * (Y = 19) * (Y = 20) A
ND GT = 1: GNUM(L) = 1: THEN 1720
1580 GJ = GT + 1: IF GT(GJ,1) < X AND
GT(GJ,2) < Y AND GT(GJ,3) < X AND
Y = 5) THEN GT = GJ
1590 GOSUB 1820: IF X < 2 OR Y < 2 OR X
V 23 OR Y > 31 THEN 1400
1600 CALL SOUND,32: CALL HCHAR,X,Y + 4,H
O: FOR LP = 1 TO SPED(SPD) / 2: NEX
T LP,I
1610 CALL SOUND,255: FS = FS + 4: CALL SO
UND,255: CALL Y < 2 OR X > 23 OR Y >
1620 IF X < 2 OR Y < 2 OR X > 23 OR Y >
31 THEN 1400

```

APPLE II Family

```

1630 CALL HCHAR,X,Y + 4,M: GOTO 1090
1640 CALL HCHAR,X,Y + 4,HO: FOR I = 1 TO
4: CALL HCHAR,X,Y + 4,M
1650 ON A GOSUB 1190,1200,1210,1220,1230
1660 CALL GCHAR,X,Y + 4:M = PEEK (0): I
F M = 160 THEN M = 25
1670 IF (X = 23) * (Y = 19) * (Y = 20) A
ND GT = 1: GNUM(L) = 1: THEN 1720
1680 GJ = GT + 1: IF GT(GJ,1) < X AND
GT(GJ,2) < Y AND GT(GJ,3) < X AND
Y = 5) THEN GT = GJ
1690 GOSUB 1820: IF X < 2 OR Y < 2 OR X
V 23 OR Y > 31 THEN 1400
1700 CALL SOUND,32: CALL HCHAR,X,Y + 4,H
O: FOR LP = 1 TO SPED(SPD) / 2: NEX
T LP,I
1710 CALL HCHAR,X,Y + 4,M: CALL HCHAR,X,
Y + 4,M: GOTO 1090
1720 GOSUB 1800: VTAB 14: HTAB 4: PRINT
ROUND COMPLETE WITH "FS" FAULTS 11
1730 IF FS = 0 THEN VTAB 10: HTAB 11: P
RINT "A PERFECT ROUND 111": CALL S
OUND,127: CALL SOUND,127: CALL SOUN
D,127
1740 VTAB 24: FOR K = 1 TO 17: CALL SO,2
: PRINT: FOR KK = 1 TO 50: NEXT KK
1750 HOME: VTAB 10: PRINT "DO YOU WANT
TO PLAY AGAIN? (Y/N) > ": POKE
16368,0: GET AS: PRINT AS: IF AS
< "Y" AND AS < "N" THEN 1750: P
RINT "BYE. SEE YOU AT THE ARENA.":
GOTO 350
1760 E = INT (FS / 100): F = INT ((FS -
(E * 100)) / 10): G = FS - (E * 100)
- (F * 10)
1770 CALL HCHAR,24,1,E + 15: CALL HCHAR,
24,2,F + 15: CALL HCHAR,24,3,G + 15
1780 RETURN
1790 HOME: TEXT: HOME: FLASH: VTAB 1
2: PRINT SPC(40): VTAB 16: PRINT
SPC(40)
1800 FOR K = 12 TO 16: VTAB K: HTAB 1: P
RINT "": HTAB 40: PRINT "": NEX
T: NORMAL: RETURN
1810 E = INT (GT / 10): F = GT - (E * 10)
: CALL HCHAR,23,2,E + 15: CALL HCH
AR,23,3,F + 15: RETURN

```

HCM

STADIUM JUMPING

COMMODORE 64

```

100 REM *****
110 REM * STADIUM JUMPING *
120 REM *****
130 REM BY KENT AND KATHY GEMMELL
140 REM AND THE HCM STAFF
150 REM HOME COMPUTER MAGAZINE
160 REM C-64 BASIC
170 REM JOYSTICK OPTIONAL
180 REM TITLE SCREEN
190 PRINT "SHIFT CLR: POKE 53280,6: PO
KE 53281,5: POKE 646,0: POKE 649,1
200 POKE 56,48: CLR: Z = RND(-TI)
210 PRINT TAB(11) "STADIUM JUMPING*"
220 PRINT "5 CRSRDOWN TAB(7) "PRESS RETU
RN TO CONTINUE"
230 GET IN$: IF IN$ = " THEN 230
240 IF ASC(IN$) < 13 THEN 230
250 PRINT TAB(7) "SHIFT CRSRUP"
260 REM INPUT LEVEL OF DIFFICULTY
270 PRINT "LEVEL 1: SCHOOLING JUMPER
S": PRINT "LEVEL 2: OPEN JUMPERS"
280 PRINT "LEVEL 3: GRAND PRIX": PRIN
T "5 CRSRDOWN WHICH LEVEL (1, 2, 3
)?":
290 POKE 198,0
300 GET IN$: IF IN$ = " THEN 300
310 IF ASC(IN$) < 49 OR ASC(IN$) > 51 THEN 3
00
320 PRINT IN$: PRINT "2 CRSRDOWN PLEASE
WAIT WHILE I PREPARE YOUR HORSE"
330 REM GRAPHICS DATA FOR CHARACTER SHA
PES
340 IF PEEK(53272) = 29 THEN 410
350 POKE 56334,PEEK(56334) AND 254: POKE 1,
PEEK(1) AND 254
360 FOR I = 0 TO 2047: POKE 12288 + I, PEEK(5
3248 + I): NEXT I
370 POKE 1,PEEK(1) OR 4: POKE 56334,PEEK(5
6334) OR 1
380 READ CHAR: IF CHAR = -1 THEN 400
390 FOR I = 0 TO 7: READ D: POKE 12288 + CHAR
* 8 + I, D: NEXT: GOTO 380
400 POKE 53272, (PEEK(53272) AND 240) OR 12
410 REM DRAW SCREEN BORDER
420 PRINT "SHIFT CLR: SC = 1024: CS = 55296
FOR I = 0 TO 39

```

```

440 POKE SC + I, 160: POKE CS + I, 6
450 POKE SC + I + 960, 160: POKE CS + 960 + I, 6
460 NEXT I
470 FOR I = 40 TO 920 STEP 40
480 POKE SC + I, 160: POKE CS + I, 6
490 POKE SC + 39 + I, 160: POKE CS + 39 + I, 6
500 NEXT I
510 J1$ = "SHIFT CRSRDOWN SHIFT CRSR
LEFT SHIFT A CRSRDOWN SHIFT CRSR
LEFT SHIFT A CRSRDOWN SHIFT CRSR
LEFT SHIFT C: J2$ = "SHIFT CRSR
DOWN SHIFT CRSRLEFT SHIFT C: J3$
= "SHIFT V CRSRDOWN 2 SHIFT CRSRL
EFT SHIFT N CRSRDOWN 2 SHIFT CRSRL
EFT SHIFT N CRSRDOWN 2 SHIFT CRSRL
EFT SHIFT V"
520 ON VAL (IN$) GOTO 530, 640, 830
530 REM DRAW LEVEL 1
540 NF = 4
550 PRINT "HOME 2 CRSRDOWN TAB(14) J1$"
560 PRINT "3 CRSRDOWN TAB(24) J1$"
570 PRINT "2 CRSRDOWN CRSRRIGHT SHIFT
C 4 SHIFT B SHIFT C CRSRDOWN 6 S
HIFT CRSRLEFT SHIFT C 4 SHIFT B S
HIFT C 5 CRSRDOWN TAB(25) J2$"
580 PRINT "CTRL GRN HOME 3 CRSRDOWN T
AB(15) B CRSRDOWN SHIFT CRSRLEFT B
CRSRDOWN SHIFT CRSRLEFT B CRSRDO
WN SHIFT CRSRLEFT B CRSRDOWN TAB(2
3) A CRSRDOWN SHIFT CRSRLEFT A CRS
RDOWN SHIFT CRSRLEFT A CRSRDOWN S
HIFT CRSRLEFT A"
590 PRINT "2 CRSRDOWN 3 CRSRRIGHT CCCCCC
6 CRSRDOWN TAB(24) D CRSRDOWN SHI
FT CRSRLEFT D CRSRDOWN SHIFT CRSRL
EFT D CTRL BLK"
600 POKE SC + 95, 178: POKE CS + 95, 2
610 POKE SC + 463, 177: POKE CS + 463, 2
620 POKE SC + 522, 179: POKE CS + 522, 2
630 POKE SC + 904, 180: POKE CS + 904, 2
640 GOTO 1060
650 REM DRAW LEVEL 2
660 NF = 7
670 PRINT "HOME 2 CRSRDOWN TAB(14) J1$"
680 PRINT "3 SHIFT CRSRUP TAB(28) J1$" 2 CRSRDO
WN

```

Continued

[illegible]

```

10900 POKER 650,128:POKER 198,0
11000 REM READ KEYBOARD AND JOYSTICK
11100 GET INS:IF INS<>"K" THEN1110
11200 POKESC+4,5
11300 M=PEEK(SC+XY):C=PEEK(CS+XY):IF (M<>
32) AND (M>NF) THEN1480
IF M=FJ+1 THEN D=1:GOTO1160
D=0
IF A AND 1 THEN1220
Y1=1:Y2=40:IF A/2=2 THEN Y2=-Y2
IF A/2=3 THEN Y1=Y2=-Y2
IF A/2=4 THEN Y1=-Y1
Y1=PEEK(SC+XY+Y1):Y2=PEEK(SC+XY+Y2)
IF (Y1<>32) AND (Y1>NF) AND (Y2<>32)
AND (Y2>NF) THEN1480
POKESC+XY,HO:POKECS+XY,9:IFINS="J"
THENPOKE198,0
FOR P=1 TO 100:NEXT
GET INS:POKE 198,0:IF INS="J" THEN1
690
IF INS="S" THEN1320
IF INS="D" THEN1340
FR=PEEK(56320) AND 16:JK=15-(PEEK(5
6320) AND 15)
IF FR<>16 THEN1690
IF JK=0 THEN1360
IF JK=8 THEN1340
REM CHANGE DIRECTION POINTER
A=A+1:IF A<9 THEN1360
A=1:GOTO1360
A=A-1:IF A>0 THEN1360
A=8-A
A=8-A
REM UPDATE POSITION POINTERS DEPEND
ING ON DIRECTION
POKESC+XY,M:POKECS+XY,C
ON A GOSUB1400,1410,1420,1430,1440,
1450,1460,1470
GOTO1130
XY=XY-40:HO=28:RETURN
XY=XY-41:HO=30:RETURN
XY=XY-1:HO=27:RETURN
XY=XY+39:HO=29:RETURN
XY=XY+40:HO=28:RETURN
XY=XY+41:HO=30:RETURN
XY=XY+1:HO=27:RETURN
XY=XY-39:HO=29:RETURN
REM CRASH ROUTINES
C=PEEK(CS+XY):IF D=1 THEN FJ=FJ+1
POKESC+XY,HO:POKECS+XY,9:FOR P=1
TO 50:NEXT
POKESC+XY,42:POKECS+XY,9:FOR P=1
TO 50:NEXT
POKESC+XY,HO:POKECS+XY,9
PRINT"SHIFT CRSRUP":PRINTTAB(15)"
CTRL RVSON"CRASH"FOR
P=1 TO 900:NEXT
IF (M<128) AND (M<176) THEN1660
Z=INT(RND(1))*(20-1)+1:IF Z=1 THEN1
660
FS=FS+4
PRINT"SHIFT CRSRUP":PRINTTAB(15)"
CTRL RVSON"FAULTS="FS"SHIFT CRSRL
EFT"
IF FJ=NF THEN2020
GOTO1610
POKESC+XY,HO:POKECS+XY,9:FOR P=1
TO 100:NEXT
POKESC+XY,M:POKECS+XY,C
ON A GOSUB1400,1410,1420,1430,1440,
1450,1460,1470
C=PEEK(CS+XY):M=PEEK(SC+XY):IF (M>1
28) AND (M<176) THEN1660
IF M<>32 THEN1600
PRINT"SHIFT CRSRUP":PRINTTAB(13)"
CTRL RVSON"RIDER*HAS*FALLEN"FOR
P=1 TO 900:NEXT
PRINT"SHIFT CRSRUP":PRINTTAB(13)"
CTRL RVSON"ELIMINATION"
FOR P=1 TO 900:NEXT
GOTO2080
REM JUMP ROUTINES
W=XY:ON A GOSUB1400,1410,1420,1430,
1440,1450,1460,1470
V=PEEK(SC+XY):XY=W
POKESC+XY,HO:POKECS+XY,9:S=1:R=0:
Q=0:L=D:IF M<NF+1 THEN D=1
IF (V>128) AND (V<176) THEN1480
IF (V=86) OR (V=67) THEN R=1
FOR I=1 TO 4:POKESC+XY,M:POKECS+X
Y,C
ON A GOSUB1400,1410,1420,1430,1440,
1450,1460,1470
M=PEEK(SC+XY):C=PEEK(CS+XY)
IF M=FJ+1 THEN L=1
POKESC+XY,HO:POKECS+XY,9:FOR P=1
TO 50:NEXT
POKESC+XY,M:POKECS+XY,C
IF M<128 THEN D=0:GOTO1480
IF (M=86) OR (M=67) THEN R=1
IF A AND 1 THEN1890
Y1=1:Y2=40:IF A/2=2 THEN Y2=-Y2
IF A/2=3 THEN Y1=Y2=-Y2
IF A/2=4 THEN Y1=-Y1
Y1=PEEK(SC+XY+Y1):Y2=PEEK(SC+XY+Y2)
IF (Y1<>32) AND (Y1>NF) AND (Y2>NF)
AND (Y2<>32) THEN S=0:IF D=0 THEN
Q=1
IF M<NF+1 THEN D=1

```

Continued

STADIUM JUMPING *Continued*

COMMODORE 64

```

1900 IF (M<V) THEN S=0: IF D
1910 GETZZ$=NEXT
1920 IF L AND (S=0) THEN FJ=FJ+1: IF FJ=N
1930 F THEN 2020
1940 IF S THEN M=160: D=0: GOTO 1480
1950 IF O THEN D=0: GOTO 1480
1960 IF R=0 THEN D=0: GOTO 1130
1970 FS=FS+4
1980 POKE SC+XY,HO: POKE CS+XY,9
1990 PRINT "SHIFT CRSRUP": PRINTTAB(15)"
2000 CTRL RVSON "RAIL*DOWN*": F
2010 OR P=1 TO 1000: NEXT
2020 PRINT "SHIFT CRSRUP": PRINTTAB(15)"
2030 CTRL RVSON "FS": SHIFT CR SRL
2040 EFT "FOR P=1 TO 1000: NEXT
2050 POKE SC+XY,M: POKE CS+XY,C
2060 D=0: GOTO 1130
2070 REM END OF THE ROUND. DISPLAY SCORE
2080 AND GIVE OPTION TO PLAY AGAIN
2090 IF FS=0 THEN 2060
2100 PRINT "SHIFT CRSRUP": PRINTTAB(5)"
2110 CTRL RVSON "ROUND COMPLETE WITH 'FS'
2120 SHIFT CR SRL "FAULTS":
2130 GOTO 2070
2140 PRINT "SHIFT CRSRUP": PRINTTAB(3)"
2150 CTRL RVSON "CONGRATULATIONS*A*PERFEC
2160 T*ROUND":
2170 FOR P=1 TO 3000: NEXT
2180 PRINT "SHIFT CRSRUP": PRINTTAB(3)"
2190 CTRL RVSON "AGAIN?":
2200 POKE 198,0

```

```

2100 GET IN$: IF IN$<>"Y" AND IN$<>"N" TH
2110 EN2100
2120 IF IN$="Y" THEN 2130
2130 PRINT "SHIFT CLR BYE, SEE YO
2140 U AT THE ARENA. "END
2150 REM ENTER LEVEL FOR A NEW GAME
2160 FS=0
2170 PRINT "SHIFT CRSRUP": PRINTTAB(12)"
2180 CTRL RVSON "LEVEL 1, 2, 3": CTRL BLK
2190 GET IN$: IF IN$=" " THEN 2160
2200 IF VAL(IN$)<1 AND VAL(IN$)>1 THEN 21
2210 GOTO 410
2220 REM GRAPHICS DATA FOR CHARACTER SHA
2230 PES
2240 DATA 027,000,000,231,060,060,231,00
2250 0,000
2260 DATA 028,036,036,060,024,024,060,03
2270 6,036
2280 DATA 029,012,024,017,123,222,136,02
2290 4,048
2300 DATA 030,048,024,136,222,123,017,02
2310 4,012
2320 DATA 065,102,102,102,102,102,102,10
2330 2,102
2340 DATA 066,000,255,255,000,000,255,25
2350 5,000
2360 DATA 067,102,255,255,102,102,255,25
2370 5,102
2380 DATA -1

```

HCM

STADIUM JUMPING

IBM PC & PCjr

```

1000 *****STADIUM JUMPING*****
1100 *****BY KENT & KATHY GEMMELL*****
1200 *****AND THE HCM STAFF*****
1300 *****HOME COMPUTER MAGAZINE*****
1400 *****VERSION 4.4.1*****
1500 *****PCjr CARTRIDGE BASIC OR*****
1600 *****IBM PC BASICA WITH*****
1700 *****COLOR/GRAPHICS MONITOR ADAPTER*****
1800 *****AND COLOR MONITOR*****
1900 *****KEY OFF:CLS:WIDTH 40:SCREEN 0:LOCAT*****
2000 *****E 12,13:PRINT "STADIUM JUMPING":LOC*****
2100 *****INUE":GOSUB 1450:CLS:PLAY "MB"*****
2200 *****SCREEN 1:COLOR 0,0:DIM H1%(25),H2%(*****
2300 *****25),H3%(25),H4%(25),H5%(25),H6%(25)*****
2400 *****H7%(25),H8%(25),H9%(25),H10%(25),H*****
2500 *****11%(25),H12%(25),H13%(25),H14%(25),*****
2600 *****H15%(25),H16%(25),F1%(12),F2%(12)*****
2700 *****F3%(12),F4%(12),M1%(35),M2%(35)*****
2800 *****GNM(3,11,2),*****
2900 *****ON KEY(1) GOSUB 460:ON KEY(11) GOSU*****
3000 *****B 430:ON KEY(12) GOSUB 440:ON KEY(1*****
3100 *****3) GOSUB 450:KEY(1) ON:KEY(11) ON:K*****
3200 *****EY(12) ON:KEY(13) ON*****
3300 *****P1$="R3DL3FD11LDR3ULU12":P2$="R3DL3*****
3400 *****FD33LDR3ULU13":P3$="D3RU3FR11URD3LU*****
3500 *****L12":P4$="D3RU3FR33URD3ULU13":P5$="*****
3600 *****F3GH3FDG9DFGH3FEF2":P6$="F3GH3FDG24D*****
3700 *****FGH3FEF2":P7$="C3FE3GDF9REFG3HE2":P8*****
3800 *****$="G3FE3GDF24REFG3HE2":*****
3900 *****H1$="C3E3R2D2HL2D6NF2HUL7D1NG2UL2UL*****
4000 *****GDUEURER8BDP3,3":H3$="C3H3L2D2ER2D6N*****
4100 *****G2EUR7D1NFD2URFDUELHL8BDP3,3":H5$*****
4200 *****="C3D3R2NELD4RNFLDLNDUL5L2NHRD4LNGR*****
4300 *****R":H7$="C3U3R2NFLU4RNEUL5L2NHRD*****
4400 *****4LNRHUR*****
4500 *****H9$="C3FDHGD2FDNR2LG3ND3GLNE5UNE4LE*****
4600 *****5U2E2":H11$="C3GDEFD2GDNL2RF3ND3FRN*****
4700 *****SUENH4RH5U2H2":H13$="C3F5R2F2GLEHL2*****
4800 *****GLND2UH3L3HUNF5RNF4U":H15$="C3G5L2*****
4900 *****G2FRHER2FRN2UE3R3EUNG5LNG4U*****
5000 *****H2$="C3E3R2D2HL2D5GNG2EU2L7D2RNF2L2*****
5100 *****U2LULGDUERER8DL8":H4$="C3H3L2D2ER2D*****
5200 *****5FNFD2HUR7D2LNG2R2U2RURFDUELHL9DR9*****
5300 *****H6$="C3D3R2RD3L3DLNDLUL3RURRBD2P3,3*****
5400 *****H8$="C3U3R2RUL3LULNULDL3RDRBU2P3,*****
5500 *****3*****
5600 *****H10$="C3FDHGD2FDNR2LG3ND3GLNE5UNE4LE*****
5700 *****5U2E2":H12$="C3GDEFD2GDNL2RF3ND3FRN*****
5800 *****SUENH4RH5U2H2":H14$="C3F5R2F2GLEHL2*****
5900 *****NL3H4LND2U2NF5RNF4U":H16$="C3G5L2*****
6000 *****FRHER2FRN2UE3R3EUNG5LNG4U*****
6100 *****DRAW "BM102,100,XH1$:BM142,100,XH2$*****
6200 *****":GET(94,94):H107,105,H11%:GET(134*****
6300 *****94):H1107,105,H12%:CLS:DRAW "BM102,*****
6400 *****100,XH3$:BM142,100,XH4$":GET(97,94*****
6500 *****):H1110,105,H13%:GET(137,94):H1510,*****
6600 *****5),H4%*****
6700 *****CLS:DRAW "BM99,96,XH5$:BM139,96,XH6*****
6800 *****$":GET(96,96):H1102,105,H5%:GET(13*****
6900 *****6,96):H11142,105,H6%:CLS:DRAW "BM99,*****
7000 *****105,XH7$:BM139,105,XH8$":GET(96,96*****
7100 *****):H11102,105,H7%:GET(136,96):H11142,*****
7200 *****5),H8%*****
7300 *****CLS:DRAW "BM103,96,XH9$:BM143,96,XH*****
7400 *****10$":GET(96,96):H11105,108,H9%:GET(*****
7500 *****136,96):H11145,108,H10%:CLS:DRAW "BM*****
7600 *****98,96,XH11$:BM138,96,XH12$":GET(96*****
7700 *****96):H11105,108,H11%:GET(136,96):H1*****
7800 *****5,108,H12%

```

```

320 CLS: DRAW "BM99,96,XH13$:BM139,96,XH
330 14$:":GET(96,96):H1108,105,H13%:GET
340 (136,96):H11148,105,H14%:CLS:DRAW "B
350 M105,96,XH15$:BM145,96,XH16$":GET(
360 96,96):H1108,105,H15%:GET(136,96):H
370 148,105,H16%
380 CLS: DRAW "BM96,96,C3D4RU4FD2RU2FG":G
390 ET(96,96):H11100,100,F1%:DRAW "BM104
400 ,100R4UL4ER2UL2EF":GET(104,96):H11108
410 ,100,F2%:DRAW "BM112,100R4UL3ER2UL
420 ED":GET(112,96):H11116,100,F3%:DRAW
430 "BM120,96R4DL3FR2DLFU":GET(120,96):
440 (124,100),F4%
450 CLS: DRAW "BM200,100,C3FL2ED2C2NLNRD2C1R2
460 C2FND2H2C1LD4C2ND2C1HNU3LNU3GC2ND2C1
470 U4LGC2D2U":GET(196,98):H11204,110,M1
480 %:DRAW "BM200,150C3FL2ED2C2NLNRD2C1
490 R3C2ENU2GC1L2D3FC2NR2C1HLNU3LGC2NL2
500 C1EU3L2C2HNU2":GET(196,150):H11204,16
510 2),M2%
520 RESTORE 1510:FOR Z=1 TO 3:READ J:FO
530 RZZ=1 TO J:READ GNM(Z,ZZ,1),GNM(Z,
540 ZZ,2):NEXT:
550 CLS:LOCATE 5,1:PRINT "LEVEL 1: SCH
560 OOLING JUMPERS":PRINT:PRINT "LEVEL
570 2: OPEN JUMPERS":PRINT:PRINT "LEVE
580 L 3: GRAND PRIZ":LOCATE 20,1:PRINT
590 "WHICH LEVEL (1, 2, OR 3)?":
600 GOSUB 1450:IF K$<"1" OR K$>"3" THEN
610 370 ELSE L=VAL(K$):GOSUB 470:MX=0:
620 MY=0:X=0:Y=8:DIR=5:NBAR=1:SND=1:CR=
630 0:DONE=0:FOUL=0:PUT(Y,X),H8%,PSET
640 GOSUB 1450:LEFT=0:RIGHT=0:JUMP=0
650 IF JUMP=1 THEN BAR=0:GOSUB 680:IF B
660 AR=NBAR AND J(BAR,6)=DIR THEN NBAR=
670 NBAR+1:SOUND 440,4:SOUND 660,8:IF N
680 BAR>JMP THEN 1420
690 IF LEFT=1 THEN GOSUB 1020 ELSE IF R
700 IGH=1 THEN GOSUB 1040
710 GOSUB 1150:IF CR=1 THEN GOTO 1340
720 IF DONE=1 THEN 1360 ELSE GOTO 390
730 IF JUMP=1 THEN RETURN ELSE JUMP=1:S
740 OUND 440,1:RETURN
750 IF LEFT=1 THEN RETURN ELSE LEFT=1:S
760 OUND 880,1:RETURN
770 IF RIGHT=1 THEN RETURN ELSE RIGHT=1
780 :SOUND 880,1:RETURN
790 IF SND=1 THEN SND=0:RETURN ELSE SND
800 =1:RETURN
810 CLS:FOR LN=0 TO 2:LINE (LN,LN)-(319
820 -LN,199-LN),1,B:NEXT:ON L GOSUB 480
830 ,490,500:RETURN
840 JMP=4:RESTORE 1470:GOSUB 1460:GOTO
850 510
860 JMP=7:RESTORE 1480:GOSUB 1460:GOTO
870 510
880 JMP=11:RESTORE 1490:GOSUB 1460
890 FOR Z=1 TO JMP:ON J(Z,7) GOSUB 520,
900 540,560,580,600,620,640,660:LOCATE
910 GNM(L,Z,1),GNM(L,Z,2):PRINT RIGHTS(
920 STR$(Z),LEN(STR$(Z))-1):NEXT:RETUR
930 N
940 IF J(Z,6)=3 THEN ST=-1 ELSE ST=1
950 LINE (J(Z,1)-(6*ST),-(4*(ST<0)),J(Z,
960 2))-(J(Z,3),J(Z,4)),1,BF:FOR Z1=0 T
970 O J(Z,5)-1:PRESET(J(Z,1)+(Z1*4*ST),
980 J(Z,2)):DRAW "C2:XP1$":NEXT:PUT(J(
990 Z,8),J(Z,9)),F2%:RETURN
1000 IF J(Z,6)=3 THEN ST=-1 ELSE ST=1

```

Continued

PROGRAM LISTING


```

550 LINE (J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
560 IF J(Z,6)=1 THEN ST=-1 ELSE ST=1
570 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
580 IF J(Z,6)=1 THEN ST=-1 ELSE ST=1
590 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
600 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
610 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
620 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
630 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
640 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
650 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
660 LINE (J(Z,1),J(Z,2),J(Z,3),J(Z,4),J(Z,5),J(Z,6),J(Z,7),J(Z,8),J(Z,9))
670 JUMP=0:ON DIR GOTO 690,730,770,810,850,890,930,970
680 PUT(Y,X),H6%:FOR T=1 TO 8:IF X-4<10
690 THEN DONE=1:RETURN
700 X=X-4:PUT(Y,X),H5%:IF CR=0 THEN GOS
710 UB 1240:PUT(Y,X),H5%:IF CR=1 AND J(
720 BAR,6)<>DIR THEN GOTO 1060 ELSE 720
730 NEXT:PUT(Y,X),H6%:RETURN
740 OR X-4<10 THEN DONE=1:RETURN
750 X=X-4:Y=Y-4:PUT(Y,X),H11%:IF CR=0 T
760 HEN GOSUB 1240:PUT(Y,X),H11%:IF CR=
770 1 AND J(BAR,6)<>DIR THEN GOTO 1060
780 ELSE 760
790 PUT(Y,X),H11%
800 NEXT:IF CR=0 THEN RETURN 1360 ELSE
810 PUT(Y,X),H12%:RETURN
820 PUT(Y,X),H4%:FOR T=1 TO 8:IF Y-4<10
830 THEN DONE=1:RETURN
840 Y=Y-4:PUT(Y,X),H3%:IF CR=0 THEN GOS
850 UB 1240:PUT(Y,X),H3%:IF CR=1 AND J(
860 BAR,6)<>DIR THEN GOTO 1060 ELSE 800
870 PUT(Y,X),H3%
880 NEXT:IF CR=0 THEN RETURN 1360 ELSE
890 PUT(Y,X),H4%:RETURN
900 PUT(Y,X),H16%:FOR T=1 TO 8:IF Y-4<1
910 0 OR X+4>186 THEN DONE=1:RETURN
920 Y=Y-4:X=X+4:PUT(Y,X),H15%:IF CR=0 T
930 HEN GOSUB 1240:PUT(Y,X),H15%:IF CR=
940 1 AND J(BAR,6)<>DIR THEN GOTO 1060
950 ELSE 830
960 NEXT:IF CR=0 THEN RETURN 1360 ELSE
970 PUT(Y,X),H16%:RETURN
980 PUT(Y,X),H15%:FOR T=1 TO 8:IF X+T>18
990 6 THEN DONE=1:RETURN
1000 X=X+4:PUT(Y,X),H7%:IF CR=0 THEN GOS
1010 UB 1240:PUT(Y,X),H7%:IF CR=1 AND J(
1020 BAR,6)<>DIR THEN GOTO 1060 ELSE 880
1030 PUT(Y,X),H7%
1040 NEXT:IF CR=0 THEN RETURN 1360 ELSE
1050 PUT(Y,X),H8%:RETURN
1060 PUT(Y,X),H14%:FOR T=1 TO 8:IF X+4>1
1070 86 OR Y+4>308 THEN DONE=1:RETURN
1080 X=X+4:Y=Y+4:PUT(Y,X),H13%:IF CR=0 T
1090 HEN GOSUB 1240:PUT(Y,X),H13%:IF CR=
1100 1 AND J(BAR,6)<>DIR THEN GOTO 1060
1110 ELSE 920
1120 PUT(Y,X),H13%
1130 NEXT:IF CR=0 THEN RETURN 1360 ELSE
1140 PUT(Y,X),H14%:RETURN
1150 PUT(Y,X),H2%:FOR T=1 TO 8:IF Y+4>30
1160 8 THEN DONE=1:RETURN
1170 Y=Y+4:IF Y>306 THEN Y=306
1180 PUT(Y,X),H1%:IF CR=0 THEN GOSUB 124
1190 0:PUT(Y,X),H1%:IF CR=1 AND J(BAR,6)
1200 <>DIR THEN GOTO 1060 ELSE 970
1210 SCREEN 0,0,0
1220 PUT(Y,X),H1%
1230 NEXT:IF CR=0 THEN RETURN 1360 ELSE
1240 PUT(Y,X),H2%:RETURN
1250 PUT(Y,X),H10%:FOR T=1 TO 8:IF Y+4>3
1260 08 OR X-4<10 THEN DONE=1:RETURN
1270 Y=Y+4:X=X-4:PUT(Y,X),H9%:IF CR=0 TH
1280 EN GOSUB 1240:PUT(Y,X),H9%:IF CR=1
1290 AND J(BAR,6)<>DIR THEN GOTO 1060 EL
1300 SE 1010
1310 PUT(Y,X),H9%
1320 NEXT:IF CR=0 THEN RETURN 1360 ELSE
1330 PUT(Y,X),H10%:RETURN
1340 GOSUB 1060:LEFT=0:DIR=DIR+1:IF DIR>
1350 8 THEN DIR=1
1360 GOSUB 1060:RETURN
1370 GOSUB 1060:RIGHT=0:DIR=DIR-1:IF DIR
1380 <1 THEN DIR=8
1390 GOSUB 1060:RETURN
1400 ON DIR GOTO 1070,1080,1090,1100,111
1410 0,1120,1130,1140
1420 PUT(Y,X),H6%:RETURN
1430 PUT(Y,X),H12%:RETURN
1440 PUT(Y,X),H4%:RETURN
1450 PUT(Y,X),H16%:RETURN
1460 PUT(Y,X),H8%:RETURN
1470 PUT(Y,X),H14%:RETURN
1480 PUT(Y,X),H2%:RETURN
1490 PUT(Y,X),H10%:RETURN
1500 ON DIR GOTO 1160,1170,1180,1190,120
1510 0,1210,1220,1230
1520 IF X-3*L>=0 THEN PUT(Y,X),H6%:PUT(Y
1530 ,X),H5%:FOR TD=1 TO 40:NEXT:PUT(Y,X
1540 ,X),H5%:X=X-3*L:PUT(Y,X),H6%:GOSUB 12
1550 40:RETURN ELSE DONE=1:RETURN
1560 IF X-3*L>=0 AND Y-3*L>3 THEN PUT(Y,
1570 X),H12%:PUT(Y,X),H11%:FOR TD=1 TO 4
1580 0:NEXT:PUT(Y,X),H11%:X=X-3*L:Y=Y-3*
1590 L:PUT(Y,X),H12%:GOSUB 1240:RETURN E
1600 LSE DONE=1:RETURN
1610 IF Y-3*L>3 THEN PUT(Y,X),H4%:PUT(Y,
1620 X),H3%:FOR TD=1 TO 40:NEXT:PUT(Y,X
1630 ,X),H3%:Y=Y-3*L:PUT(Y,X),H4%:GOSUB 124
1640 0:RETURN ELSE DONE=1:RETURN
1650 IF X+3*L<186 AND Y-3*L>3 THEN PUT(Y
1660 ,X),H16%:PUT(Y,X),H15%:FOR TD=1 TO
1670 40:NEXT:PUT(Y,X),H15%:X=X+3*L:Y=Y-3
1680 *L:PUT(Y,X),H16%:GOSUB 1240:RETURN
1690 ELSE DONE=1:RETURN
1700 IF X+3*L<186 THEN PUT(Y,X),H8%:PUT(
1710 Y,X),H7%:FOR TD=1 TO 40:NEXT:PUT(Y,
1720 X),H7%:X=X+3*L:PUT(Y,X),H8%:GOSUB 1
1730 240:RETURN ELSE DONE=1:RETURN
1740 IF X+3*L<186 AND Y+3*L<308 THEN PUT
1750 (Y,X),H14%:PUT(Y,X),H13%:FOR TD=1 T
1760 O 40:NEXT:PUT(Y,X),H13%:X=X+3*L:Y=Y
1770 +3*L:PUT(Y,X),H14%:GOSUB 1240:RETUR
1780 N ELSE DONE=1:RETURN
1790 IF Y+3*L<308 THEN PUT(Y,X),H2%:PUT(
1800 Y,X),H1%:FOR TD=1 TO 40:NEXT:PUT(Y,
1810 X),H1%:Y=Y+3*L:PUT(Y,X),H2%:GOSUB 1
1820 240:RETURN ELSE DONE=1:RETURN
1830 IF X-3*L>=0 AND Y+3*L<308 THEN PUT(
1840 Y,X),H10%:PUT(Y,X),H9%:FOR TD=1 TO
1850 40:NEXT:PUT(Y,X),H9%:X=X-3*L:Y=Y+3*
1860 L:PUT(Y,X),H10%:GOSUB 1240:RETURN E
1870 LSE DONE=1:RETURN
1880 IF SND=1 THEN SOUND 330,1:SOUND 220
1890 ,1
1900 CR=0:C1=POINT(Y+1,X+1):C2=POINT(Y+7
1910 ,X+7):IF C=1 OR C=2 OR C2=1 OR C2=2
1920 THEN CR=1 ELSE RETURN
1930 CR=0:X1=X+4:Y1=Y+4:FOR Z=1 TO JMP:O
1940 N J(Z,6) GOTO 1290,1330,1300,1320,1
1950 310,1330,1280,1320
1960 NEXT:RETURN
1970 IF Y1<=J(Z,3) AND Y1>=J(Z,1)-4 AND
1980 X1>=J(Z,2)-4 AND X1<=J(Z,4)+4 THEN CR
1990 =1:BAR=Z:RETURN ELSE GOTO 1270
2000 IF Y1>=J(Z,1) AND Y1<=J(Z,3) AND X1
2010 <=J(Z,2)+4 AND X1>=J(Z,4) THEN CR=1
2020 :BAR=Z:RETURN ELSE GOTO 1270
2030 IF X1>=J(Z,2)-16 AND X1<=J(Z,4) AND
2040 Y1<=J(Z,1)+4 AND Y1>=J(Z,3) THEN C
2050 R=1:BAR=Z:RETURN ELSE GOTO 1270
2060 IF X1>=J(Z,2)-4 AND X1<=J(Z,4) AND
2070 Y1>=J(Z,1) AND Y1<=J(Z,3) THEN CR=1
2080 :BAR=Z:RETURN ELSE GOTO 1270
2090 IF X1>=J(Z,2) AND X1<=J(Z,2)+J(Z,3)
2100 AND Y1<=J(Z,1) AND Y1>=J(Z,1)-J(Z,
2110 3) THEN CR=1:BAR=Z:RETURN ELSE GOTO
2120 1270
2130 IF X1>=J(Z,2) AND X1<=J(Z,2)+J(Z,3)
2140 AND Y1<=J(Z,1) AND Y1>=J(Z,1)-J(Z,
2150 3) THEN CR=1:BAR=Z:RETURN ELSE GOTO
2160 1270
2170 SOUND 110,1:SOUND 330,1:SOUND 110,1
2180 FOUL=FOUL+4:IF INT(RND*10)<>5 THEN
2190 420
2200 IF Y<155 THEN FO=20 ELSE FO=-30
2210 IF X>175 THEN X=175
2220 FOR Z=1 TO 10:SOUND 320-(Z*20),2:PU
2230 T(Y+FO,X),M1%,PSET:FOR TD=1 TO 200:
2240 NEXT:PUT(Y+FO,X),M2%,PSET:FOR TD=1
2250 TO 50:NEXT:NEXT:PUT(Y+FO,X),M1%,PSE
2260 T
2270 LINE(0,75)-(319,125),0,BF:LOCATE 12
2280 ,1:PRINT "YOUR RIDER HAS FALLEN AND
2290 "HAS BEEN ELIMINATED FROM THE
2300 "COMPETITION."
2310 PRINT:PRINT "WOULD YOU LIKE TO PLAY
2320 AGAIN (Y/N)?"

```

Continued

STADIUM JUMPING *Continued*

IBM PC & PCjr

1410	GOSUB 1450: IF K\$="Y" OR K\$="Y" THEN	1480	DATA 174, 80, 80, 27, 8, 1, 6, 6, 57, 111, 162, 12
1420	EN 1410 ELSE IF K\$="N" AND K\$="Y" THEN	1490	DATA 225, 94, 10, 107, 46, 2, 7, 2, 103, 42, 204, 12
1430	PRINT: PRINT "YOUR SCORE = " FOU: " FOU: " LL	1500	DATA 120, 27, 12, 2, 167, 23, 15, 3, 24, 30, 7, 16, 8
1440	GOTO 1400	1510	DATA 19, 23, 5, 30, 9, 33, 9, 3, 24, 23, 7, 14, 12
1450	K\$=INKEY\$: IF K\$=" " THEN 1450 ELSE R		
1460	ETURN		
1470	DATA 20, 110, 70, 123, 106, 2, 7, 2, 118, 102, 1, 6		
	7, 1, 226, 182		

HCM

STADIUM JUMPING

TI-99/4A

100	REM ** STADIUM JUMPING **	890	CALL VCHAR(11, 4, 96, 3)
110	REM ** KENT & KATHY GEMMELL	900	FOR I=1 TO 11
120	REM BY HOME COMPUTER MAGAZINE	910	READ X, Y
130	REM VERSION 4.4.1	920	CALL HCHAR(X, Y, 33)
140	REM TI BASIC EXTENDED BASIC	930	NEXT I
150	REM TI JOYSTICKS OPTIONAL	940	FOR I=1 TO 10
160	CALL CLEAR	950	READ X, Y
170	CALL SCREEN(4)	960	CALL HCHAR(X, Y, 34)
180	FOR I=1 TO 8	970	NEXT I
190	CALL COLOR(I, 4, 16)	980	FOR I=49 TO 57
200	NEXT I	990	READ X, Y
210	CALL CHAR(33, "FFFFFFFFFFFFFFFFFFFF")	1000	CALL HCHAR(X, Y, I)
220	CALL VCHAR(1, 31, 33, 96)	1010	NEXT I
230	PRINT "STADIUM JUMPING": : : :	1020	CALL HCHAR(20, 21, 49, 2)
240	PRINT: : : PRESS ENTER TO CONTINUE"	1030	CALL HCHAR(20, 27, 49)
250	CALL KEY(0, K, S)	1040	CALL HCHAR(20, 28, 48)
260	IF (S=0)+(K<49)+(K>51) THEN 360	1050	CALL HCHAR(1, 3, 104)
270	L=K-48	1060	DATA 2, 8, 4, 8, 6, 17, 8, 17, 20, 20, 22, 20
280	CALL CLEAR	1070	DATA 2, 2, 2, 2, 6
290	RESTORE 3880	1080	DATA 3, 8, 7, 17, 21, 20, 21, 8, 11, 20, 6, 22, 6
300	FOR Z=1 TO 3	1090	DATA 2, 7, 11, 3, 16, 21, 6, 26, 12, 24, 7, 19, 7, 13
310	READ Z1, Z1\$	1100	DATA 2, 2, 7, 4, 25, 10, 26, 12, 27, 7, 28
320	CALL CHAR(Z1, Z1\$)	1110	DATA 3, 6, 3, 15, 3, 24, 13, 27, 7, 19, 7, 13
330	NEXT Z	1120	DATA 10, 3, 21, 5, 15, 14, 21, 21, 21, 28
340	PRINT "LEVEL 1: SCHOOLING JUMPERS"	1130	DATA 3, 7, 3, 14, 3, 25, 12, 26, 7, 18, 7, 14
350	PRINT "LEVEL 2: OPEN JUMPERS": : : :	1140	DATA 21, 4, 16, 13, 21, 22, 21, 27, 6, 18, 6, 13
360	PRINT "LEVEL 3: GRAND PRIX": : : :	1150	DATA 10, 2, 2, 5, 16, 16, 19, 23
370	PRINT "WHICH LEVEL (1, 2, 3)?"	1160	DATA 1, 1, 7, 1, 17, 1, 27, 13, 24, 9, 14, 12, 5
380	CALL KEY(0, K, S)	1170	DATA 19, 6, 12, 16, 19, 23
390	IF (S=0)+(K<49)+(K>51) THEN 360	1180	GOTO 1790
400	L=K-48	1190	RESTORE 1400
410	CALL CLEAR	1200	FOR I=1 TO 6
420	RESTORE 3880	1210	READ X, Y
430	FOR Z=1 TO 3	1220	CALL VCHAR(X, Y, 98)
440	READ Z1, Z1\$	1230	NEXT I
450	CALL CHAR(Z1, Z1\$)	1240	CALL VCHAR(4, 10, 99, 2)
460	NEXT Z	1250	CALL VCHAR(10, 18, 99, 2)
470	FOR Z=95 TO 107	1260	CALL VCHAR(21, 20, 99)
480	READ Z1, Z1\$	1270	CALL VCHAR(14, 3, 96, 2)
490	CALL CHAR(Z, Z1\$)	1280	CALL VCHAR(14, 7, 96, 2)
500	NEXT Z	1290	CALL HCHAR(14, 4, 97, 3)
510	FOR Z=1 TO 10	1300	CALL HCHAR(15, 4, 97, 3)
520	READ Z1, Z2	1310	FOR I=1 TO 8
530	CALL COLOR(Z, Z1, Z2)	1320	READ X, Y
540	NEXT Z	1330	CALL HCHAR(X, Y, 33)
550	CALL CLEAR	1340	NEXT I
560	CALL SCREEN(16)	1350	FOR I=1 TO 8
570	CALL HCHAR(1, 1, 32, 768)	1360	READ X, Y
580	CALL HCHAR(1, 1, 102, 32)	1370	CALL HCHAR(X, Y, 34)
590	CALL HCHAR(24, 1, 102, 32)	1380	NEXT I
600	CALL VCHAR(1, 1, 102, 24)	1390	FOR I=49 TO 52
610	CALL VCHAR(1, 32, 102, 24)	1400	READ X, Y
620	ON L GOTO 1160, 1450, 610	1410	CALL HCHAR(X, Y, I)
630	RESTORE 1060	1420	NEXT I
640	FOR I=1 TO 8	1430	DATA 3, 10, 6, 10, 9, 18, 12, 18, 20, 20, 22
650	READ X, Y	1440	DATA 5, 13, 6, 21, 18
660	CALL VCHAR(X, Y, 98)	1450	DATA 5, 12, 6, 21, 19
670	NEXT I	1460	DATA 12, 17, 3, 11, 13, 3, 22, 19
680	FOR I=1 TO 4	1470	GOTO 1790
690	READ X, Y	1480	RESTORE 1740
700	CALL VCHAR(X, Y, 99)	1490	FOR I=1 TO 10
710	NEXT I	1500	READ X, Y
720	FOR I=1 TO 6	1510	CALL HCHAR(X, Y, 98)
730	READ X, Y	1520	NEXT I
740	CALL HCHAR(X, Y, 98, 2)	1530	FOR I=1 TO 9
750	NEXT I	1540	READ X, Y
760	FOR I=1 TO 3	1550	CALL HCHAR(X, Y, 99)
770	READ X, Y	1560	NEXT I
780	CALL HCHAR(X, Y, 99, 2)	1570	CALL VCHAR(11, 27, 96, 2)
790	NEXT I	1580	CALL VCHAR(11, 30, 96, 2)
800	FOR I=1 TO 4	1590	CALL HCHAR(11, 28, 97, 2)
810	READ X, Y	1600	CALL HCHAR(12, 28, 97, 2)
820	CALL VCHAR(X, Y, 100)	1610	CALL HCHAR(10, 10, 100)
830	NEXT I		
840	CALL HCHAR(3, 26, 101)		
850	CALL HCHAR(11, 25, 101)		
860	CALL HCHAR(13, 14, 103)		
870	CALL HCHAR(14, 15, 95)		
880	CALL HCHAR(15, 16, 103)		
	CALL VCHAR(11, 2, 96, 3)		
	CALL VCHAR(11, 3, 97, 3)		

Continued


```

1620 FOR I=1 TO 13
1630 READ X,Y
1640 CALL HCHAR(X,Y,33)
1650 NEXT I
1660 FOR I=1 TO 13
1670 READ X,Y
1680 CALL HCHAR(X,Y,34)
1690 NEXT I
1700 FOR I=49 TO 55
1710 READ X,Y
1720 CALL HCHAR(X,Y,I)
1730 NEXT I
1740 DATA 3,11,6,11,3,21,6,21,14,18,16,1
      8,19,13,23,13,20,20,22,20
1750 DATA 3,21,13,22,13,21,20,21,5,21,15,18,20,1
      23,5,23,4,13,5,13,20,11,21,11,22,11
1760 DATA 21,18
1770 DATA 10,7,11,8,15,17,14,28,14,29,4,
      22,5,22,4,12,5,12,20,12,21,12,22,12
1780 DATA 12,22,19,13,30,3,22,3,12,23,
      12,22,19
1790 AS="FAULTS="&CHR$(102)&STR$(FS)
1800 X1=24
1810 Y1=1
1820 GOSUB 3830
1830 FOR D=1 TO 200
1840 NEXT D
1850 A=5
1860 X=1
1870 Y=3
1880 HO=104
1890 CALL HCHAR(X,Y,HO)
1900 CALL KEY(0,K,S)
1910 IF (S=0)+(K<>75) THEN 1900
1920 CALL HCHAR(X,Y,32)
1930 CALL GCHAR(X,Y,M)
1940 IF M<36 THEN 1960
1950 IF M<104 THEN 2470
1960 CALL HCHAR(X,Y,HO)
1970 CALL KEY(0,K,S)
1980 IF K<>74 THEN 2010
1990 CALL HCHAR(X,Y,32)
2000 GOTO 2890
2010 IF K=83 THEN 2120
2020 IF K=68 THEN 2080
2030 CALL JOYST(2,J1,J2)
2040 CALL KEY(2,K,S)
2050 IF (S<>0)+(K=18) THEN 1990
2060 IF J1=0 THEN 2150
2070 IF J1=-4 THEN 2120
2080 A=A-1
2090 IF A>0 THEN 2150
2100 A=8-A
2110 GOTO 2150
2120 A=A+1
2130 IF A<9 THEN 2150
2140 A=1
2150 CALL HCHAR(X,Y,M)
2160 CALL SOUND(-1000,-4,10)
2170 ON A GOSUB 2190,2220,2260,2290,2330
      ,2360,2400,2430
2180 GOTO 1930
2190 X=X-1
2200 HO=104
2210 RETURN
2220 X=X-1
2230 Y=Y-1
2240 HO=107
2250 RETURN
2260 Y=Y-1
2270 HO=105
2280 RETURN
2290 X=X+1
2300 Y=Y-1
2310 HO=106
2320 RETURN
2330 X=X+1
2340 HO=104
2350 RETURN
2360 Y=Y+1
2370 X=X+1
2380 HO=107
2390 RETURN
2400 Y=Y+1
2410 HO=105
2420 RETURN
2430 Y=Y+1
2440 X=X-1
2450 HO=106
2460 RETURN
2470 CALL HCHAR(X,Y,M)
2480 CALL GCHAR(X,Y,M)
2490 CALL HCHAR(X,Y,111)
2500 CALL SOUND(-750,-7,10)
2510 AS="**CRASH**"
2520 X1=24
2530 Y1=15
2540 GOSUB 3830
2550 FOR D=1 TO 100
2560 NEXT D
2570 CALL HCHAR(24,13,102,32)
2580 IF M=102 THEN 2750
2590 IF (M>57)*(M<95) THEN 2750
2600 RANDOMIZE
2610 Z=INT(RND*15)+1
2620 IF Z=1 THEN 2750
2630 FIS=FIS+4
2640 AS=STR$(FIS)
2650 Y1=9
2660 GOSUB 3830
2670 IF (X<23)*(X>19)*(Y=20) THEN 3390
2680 CALL HCHAR(X,Y,M)
2690 ON A GOSUB 2190,2220,2260,2290,2330
      ,2360,2400,2430
2700 CALL GCHAR(X,Y,M)
2710 IF M=102 THEN 2750
2720 IF M>36 THEN 2690
2730 CALL HCHAR(X,Y,M)
2740 GOTO 1960
2750 CALL HCHAR(24,1,102,32)
2760 AS="RIDER*HAS*FALLEN"
2770 Y1=8
2780 GOSUB 3830
2790 FOR D=1 TO 100
2800 NEXT D
2810 CALL HCHAR(24,1,102,32)
2820 AS="ELIMINATION"
2830 Y1=10
2840 GOSUB 3830
2850 FOR D=1 TO 100
2860 NEXT D
2870 CALL HCHAR(X,Y,M)
2880 GOTO 3590
2890 CALL HCHAR(X,Y,HO)
2900 IF M=33 THEN 3260
2910 IF M=34 THEN 3030
2920 FOR I=1 TO 4
2930 CALL HCHAR(X,Y,M)
2940 ON A GOSUB 2190,2220,2260,2290,2330
      ,2360,2400,2430
2950 CALL GCHAR(X,Y,M)
2960 IF X=(X=21)*(Y=20) THEN 3390
2970 IF M=102 THEN 2750
2980 IF (M>57)*(M<95) THEN 2750
2990 IF M>35 THEN 2470
3000 CALL HCHAR(X,Y,HO)
3010 NEXT I
3020 GOTO 2470
3030 CALL HCHAR(X,Y,HO)
3040 FOR I=1 TO 4
3050 CALL HCHAR(X,Y,M)
3060 ON A GOSUB 2190,2220,2260,2290,2330
      ,2360,2400,2430
3070 CALL GCHAR(X,Y,M)
3080 IF (X<23)*(X>19)*(Y=20) THEN 3390
3090 IF M<95 THEN 3120
3100 IF M=102 THEN 2750
3110 CALL SOUND(100,-7,5)
3120 CALL HCHAR(X,Y,HO)
3130 NEXT I
3140 FS=FS+4
3150 AS="*RAIL*DOWN*"
3160 X1=24
3170 Y1=12
3180 GOSUB 3830
3190 IF M=102 THEN 2760
3200 AS=STR$(FIS)
3210 Y1=9
3220 GOSUB 3830
3230 CALL HCHAR(24,12,102,20)
3240 CALL HCHAR(X,Y,M)
3250 GOTO 1930
3260 CALL HCHAR(X,Y,HO)
3270 FOR I=1 TO 4
3280 CALL HCHAR(X,Y,M)
3290 ON A GOSUB 2190,2220,2260,2290,2330
      ,2360,2400,2430
3300 CALL GCHAR(X,Y,M)
3310 IF (X<23)*(X>19)*(Y=20) THEN 3390
3320 IF M=102 THEN 2750
3330 IF (M>57)*(M<95) THEN 2750
3340 CALL HCHAR(X,Y,HO)
3350 NEXT I
3360 CALL HCHAR(X,Y,M)
3370 CALL HCHAR(X,Y,M)
3380 GOTO 1930
3390 CALL HCHAR(24,1,102,32)
3400 IF FS=0 THEN 3520
3410 AS="*ROUND*COMPLETE*WITH*"
3420 X1=24
3430 Y1=1
3440 GOSUB 3830
3450 AS=STR$(FIS)
3460 Y1=21
3470 GOSUB 3830
3480 AS="FAULTS"
3490 Y1=24
3500 GOSUB 3830
3510 GOTO 3560
3520 AS="CONGRATULATIONS*A*PERFECT*ROUND"
3530 X1=24
3540 Y1=1
3550 GOSUB 3830
3560 FOR D=1 TO 100
3570 NEXT D
3580 CALL HCHAR(X,Y,M)
3590 CALL HCHAR(24,1,102,32)
3600 AS="*AGAIN?"
3610 X1=24
3620 Y1=13
3630 GOSUB 3830
3640 CALL KEY(0,K,S)
3650 IF S=0 THEN 3640
3660 IF (K<>89)*(K<>78) THEN 3640

```

Continued


```

36700 IF K=89 THEN 3710
36800 CALL "CLEAR
36900 PRINT : "BYE, SEE YOU AT THE ARENA. ":
      :
37000 END
37100 FS=0
37200 J=0
37300 CALL "HCHAR(24,1,102,32)
37400 AS="LEVEL*1*2*OR*3",
37500 Y1=9
37600 GOSUB 3830
37700 CALL KEY(0,K,S)
37800 IF (S=0)+(K<49)+(K>51) THEN 3770
37900 CALL "HCHAR(24,1,102,32)
38000 IF K=48=L THEN 1790
38100 L=K-48
38200 GOTO 530

```

```

3830 FOR I=1 TO LEN(AS$)
3840 C=ASC(SEG$(AS$,I,1))
3850 CALL HCHAR(X1,Y1+I,C)
3860 NEXT I
3870 RETURN
3880 DATA 33,0,34,0,111,,083CFF,F3CFF,1C0808,
    ,COE0F70381COE0F703,CFFFFF3C3CFFFFF3CE,
    ,OFFFFF0000OFFFFF,66666FFF,FFFFFFFF66666
3890 DATA 66666666666666666666,E3F7FE7C3E7FC,
    FC7,0307081C3870EOEC,O,FFFFFFF7FFFFFFFFF
    FF,C7EF7F3E7CFEF7E3,10387C38387C3C1
    0
3900 DATA 00247EFF7624,,011E1E7E7E787880,
    8078787E7E1E1E01
3910 DATA 16,4,9,16,16,9,16,9,16,9,16,9,
    16,9,16,4,16,4,2,4

```

HCM

TAX DEDUCTION

```

100 REM ***** TAX DEDUCTION FILER *****
110 REM ***** BY THE HCM STAFF *****
120 REM ***** HOME COMPUTER MAGAZINE *****
130 REM ***** VERSION 4.4.1 *****
140 REM ***** APPLE II FAMILY APPLESOFT *****
150 DIM AS(500), T(17), NS(17): DS = CHR$(4)
160 FOR I = 1 TO 10: READ MS(1): NEXT I
170 FOR I = 1 TO 17: READ NS(1): NEXT I
180 HOME: HTAB 10: VTAB 12: PRINT MS(1)
190 : WAS THIS PROGRAM LOADED UNDER:
200 HTAB 10: PRINT PRODOS 3.3: HTAB 10: PRINT "2"
210 : NORMAL: GET K$: IF K$ < "1" OR K$ > "2"
220 : THEN GET 210
230 IF K$ = "1" THEN OP = 1: GOTO 240
240 OP = 2
250 REM MAIN MENU
HOME: HTAB 10: VTAB 2: PRINT MS(1)
: VTAB 6: FOR I = 1 TO 8: PRINT TA
: B(6) I: ) MS(1) I = + 1: PRINT: NEX
: T I: ) PRINT: HTAB 6: INVERSE: PRIN
: T MS(10): NORMAL
260 GET K$: IF K$ < "1" OR K$ > "8" THEN
270 ON VAL(K$) GOSUB 410, 490, 740, 810,
870, 1110, 1310, 1530
280 GOTO 250
290 REM DISPLAY CATEGORIES
300 FOR Z = 1 TO 17: PRINT Z; "- "; NS(Z)
310 : NEXT Z
320 IF R = 0 THEN RETURN
AC = ASC(AS(Z)), 3, ASC(Z), 100: DAS = MIDS
(AS(Z), 3, ASC(Z), 100): V$ = ASC(RIGHT$
(Z), 100) - ASC(MID$(AS(Z), 2, 1)) -
(98) - RETURN
330 N = 0: FOR Z = 1 TO R: IF ASC(AS(
Z)) < N THEN N = ASC(AS(Z))
340 N = N + 1: GOSUB 310: PRINT Z; "- ";
DAS TAB(30) " $ " V$: IF N / 20 < >
350 INT(N / 20) THEN
INVERSE: PRINT PRESS RETURN TO
360 CONTINUE
GET K$: IF K$ < > CHR$(13) THEN
370 NEXT Z: IF N = 0 THEN PRINT "NO DA
380 TA IN THIS CATEGORY" PRINT "PRESS RETU
390 RN TO CONTINUE": NORMAL
GET K$: IF K$ < > CHR$(13) THEN
390 RETURN
400 REM ADD DATA
410 HOME: HTAB 13: PRINT MS(2): PRINT
420 : GOSUB 290: PRINT: INVERSE: PRIN
: T MS(10): NORMAL
430 INPUT CS: IF CS = " " THEN RETURN
440 C = VAL(CS): IF C < 1 OR C > 17 T
HEN: VTAB 21: INVERSE: PRINT MS(10)
: ) NORMAL: VTAB 15:
450 HOME: HTAB (15): PRINT MS(2): PRIN
: T: PRINT NS(C): PRINT "ITE
M DESCRIPTION": INPUT DAS: IF DAS
= " " THEN RETURN
460 IF LEN(DAS) > 27 THEN 440
470 IF INT V$ ENTER ITEM RETURN
480 V$ = STR$(V) + R + 1: AS(R) = C
+ (100) + DAS + V$: CHR$(LEN(DAS)) +
GOTO 410
490 REM CHANGE DATA
500 HOME: HTAB 17: PRINT MS(3): PRINT
: GOSUB 290: PRINT: INVERSE: PRIN
: T MS(10): NORMAL
510 INPUT CS: IF CS = " " THEN RETURN
520 C = VAL(CS): IF C < 1 OR C > 17 T
HEN: VTAB 21: INVERSE: PRINT MS(10)
: ) NORMAL: VTAB 21: GOTO 510
530 HOME: HTAB (20) LEN(NS(C)) / 2)
: PRINT NS(C)

```

APPLE II Family

```

540 VTAB 4: IF AS(1) = " " THEN PRINT "
550 NO DATA IN MEMORY": FOR DE = 1 TO
560 2000: NEXT: RETURN
570 GOSUB 330: IF N = 0 THEN 500
580 PRINT: PRINT "CHANGE WHICH ITEM?":
INPUT CH$: IF CH$ = " " THEN RETURN
590 GOSUB 670: IF CH = 0 THEN 500
600 Z = CH: GOSUB 310: IF AC < C THEN
610 N PRINT "THE RECORD YOU REQUESTED
620 IS NOT IN THIS CATEGORY": PRINT: P
630 RINT "TRY AGAIN": FOR DE = 1 TO 100
640 0: NEXT DE: GOTO 500
650 PRINT "EITHER: ENTER /D/ TO DELETE
660 ITEM": PRINT: PRESS RETURN
670 TO LEAVE ENTRY UNCHANGE
680 D: PRINT: ENTER NEW DATA: A
690 T: PROMPT: PRINT: PRINT "OLD DATA:
700 ": DAS: PRINT "NEW DATA: ": INPUT ND
710 $
720 IF LEN (ND$) > 27 THEN 590
730 IF ND$ < " /D/ " THEN 630
740 FOR I = CH TO R - 1: AS(I) = AS(I +
750 1): NEXT I: AS(R) = " ": R = R - 1: GO
760 TO 500
770 IF ND$ < " " THEN DAS = ND$
780 PRINT "OLD AMOUNT: ": VS: PRINT "NEW
790 AMOUNT: ": INPUT NV$: IF NV$ = " " T
800 HEN 660
810 VS = STR$ (VAL (NV$))
820 AS(CH) = CHR$ (AC + 100) + CHR$ (
830 LEN (DAS) + 100) + DAS + VS: GOTO
840 500
850 REM SEARCH FOR RECORD
860 IF ABS (ASC (CH$) - 53) > 4 THEN
870 710
880 CH = VAL (CH$): IF CH > R THEN PR
890 INT "INVALID RECORD NUMBER": CH = 0:
900 GOTO 730
910 RETURN
920 FOR I = 1 TO R: CH = I: IF CH$ = MI
930 DS (AS(I), 3, LEN (CH$)) THEN RETUR
940 N
950 NEXT I: PRINT "RECORD NOT FOUND": CH
960 = 0
970 FOR DE = 1 TO 2000: NEXT: RETURN
980 REM DISPLAY DATA
990 HOME: HTAB 12: PRINT MS(4): PRINT
1000 : GOSUB 290: PRINT: INVERSE: PRIN
1010 T MS(10): NORMAL
1020 INPUT CS: IF CS = " " THEN RETURN
1030 C = VAL (CS): IF C < 1 OR C > 17 T
1040 HEN VTAB 21: INVERSE: PRINT MS(10
1050 ): NORMAL: VTAB 21: GOTO 760
1060 HOME: HTAB (20 - LEN (NS(C)) / 2)
1070 : PRINT NS(C)
1080 VTAB 4: IF AS(1) = " " THEN PRINT "
1090 NO DATA IN MEMORY": FOR DE = 1 TO 2
1100 000: NEXT: RETURN
1110 GOSUB 330: GOTO 750
1120 REM TOTALS
1130 HOME: HTAB 15: PRINT MS(5): PRINT
1140 : FOR I = 1 TO 17: T(I) = 0: NEXT I
1150 FOR Z = 1 TO R: GOSUB 310: T(AC) = T
1160 (AC) + VAL (VS): NEXT Z
1170 VTAB 4: FOR Z = 1 TO 17: PRINT NS(Z
1180 ) TAB (28) $": T(Z): NEXT Z: PRINT:
1190 INVERSE: PRINT "PRESS RETURN TO G
1200 O BACK TO MENU": NORMAL
1210 GET KS: IF KS < > CHR$ (13) THEN
1220 850
1230 RETURN
1240 REM PRINT REPORT
1250 HOME: HTAB 10: PRINT MS(6): IF SN
1260 > 0 THEN 910
1270 IF AS(1) = " " THEN PRINT: PRINT "
1280 THERE IS NO DATA IN MEMORY": FOR DE
1290 = 1 TO 2000: NEXT: RETURN
1300 PRINT: INPUT "WHAT SLOT # IS YOUR
1310 PRINTER? ": SN: IF SN < 1 OR SN > 6
1320 THEN 880
1330 PRINT: PRINT "1) TOTALS": PRINT:
1340 PRINT "2) ALL RECORDS IN A CATEGORY
1350 ": PRINT: PRINT "3) PRINT ALL RECO
1360 RDS": PRINT: INVERSE: PRINT MS(10
1370 ): NORMAL

```

Continued


```

920 GET TURN CS: IF CS = CHR$(13) THEN RE
930 IF CS < "1" OR CS > "3" THEN 920
940 ON VAL(10) GOTO 950, 970, 1050
950 GOSUB NEXT I: FOR Z = 1 TO 17: T(I) =
0: T(AC) = T(AC) + VAL(V$): NEXT Z
: PRINT "CATEGORY": POKE 36, 40: PRIN
T "AMOUNT"
960 FOR I = 1 TO 17: PRINT NS(I): POKE
36, 40: GOTO 880
970 HOME: PRINT: TAB(7): PRINT "CATEG
ORY": PRINT: GOSUB 290: PRINT: IN
VERSE: PRINT MS(10): NORMAL
980 INPUT CS: IF CS = "1" THEN RETURN
990 C = VAL(C$): IF C < 1 OR C > 17 T
HEN VTAB 21: INVERSE: PRINT MS(10)
: NORMAL: VTAB 21: GOTO 980
1000 GOSUB 1070: POKE 36, 13: PRINT NS(C)
: PRINT: PRINT "RECORD": POKE 36, 4
0: PRINT "AMOUNT": POKE 36, 4
1010 N = 0: FOR Z = 1 TO R: IF ASC(AS(
Z)) < "1" OR ASC(AS(Z)) > "3" THEN 1030
1020 N = N + 1: GOSUB 310: PRINT Z: " "
: POKE 36, 10: PRINT DS: POKE 36, 4
0: PRINT "VS"
1030 NEXT Z: IF N = 0 THEN PRINT "NO DA
TA IN THIS CATEGORY"
1040 GOSUB 1090: GOTO 880
1050 GOSUB 1070: POKE 36, 15: PRINT "ALL
CATEGORIES": PRINT: PRINT "RECORD"
: POKE 36, 10: PRINT "CATEGORY": PO
KE 36, 20: PRINT "DESCRIPTION": PO
KE 36, 50: PRINT "AMOUNT"
1060 FOR Z = 1 TO R: GOSUB 310: PRINT Z:
: POKE 36, 10: PRINT AC: POKE 36, 20
: PRINT DS: POKE 36, 50: PRINT "S"
: V$: NEXT Z: GOSUB 1090: GOTO 880
1070 IF OP < "1" THEN PRINT DS: "PR#": S
N: PRINT CHR$(9): "80N": RETURN
1080 PR# SN: PRINT CHR$(9): "80N": RETU
RN
1090 PRINT CHR$(9): "1": IF OP < "1" T
HEN PRINT DS: "PR# 0": RETURN
1100 PR# 0: RETURN
1110 REM LOAD DISK FILE
1120 HOME: HTAB 12: PRINT MS(7): PRINT
: PRINT "WHAT FILENAME DO YOU WIS
H TO LOAD": INPUT FS: IF FS = "T
HEN RETURN
ONERR GOTO 1200
1130 PRINT DS: LOAD: FS: REM FORCE E
1140 ERROR
1150 PRINT DS: "OPEN": FS
1160 PRINT DS: "READ": FS
1170 INPUT R: FOR I = 1 TO R: INPUT AS(I)
: NEXT I
1180 PRINT DS: "CLOSE": FS
1190 POKE 216, 0: GOTO 250
1200 REM ERROR HANDLING ROUTINE
1210 ER = PEEK(222): POKE 216, 0
1220 IF ER = 13 THEN GOTO 1150: REM
FILE EXISTS
1230 PRINT
1240 IF ER = 6 THEN PRINT "EITHER FILE
NOT FOUND OR WRONG TYPE": PRINT "DO
YOU WANT A CATALOG OF THE DISK? Y/
N"
GET KS: IF KS = "Y" THEN PRINT
CHR$(13): DS: "CATALOG"

```

```

1250 IF ER = 8 THEN PRINT "DRIVE NOT RE
ADY"
1260 IF ER = 5 THEN PRINT "FILE NOT PRO
PERLY SAVED FROM BEFORE"
1270 PRINT: PRINT "PRESS <RETURN> TO TR
Y AGAIN: PRESS <ESC> TO
GO BACK TO MENU": GET KS
1280 IF KS = CHR$(13) THEN GOTO 1120
1290 IF KS = CHR$(27) THEN GOTO 250
1300 GOTO 1270
1310 REM SAVE FILE TO DISK
1320 IF AS(1) = " " THEN HOME: PRINT "T
HERE IS NO DATA IN MEMORY TO SAVE":
FOR DE = 1 TO 2000: NEXT DE: RETUR
N
1330 HOME: HTAB 10: PRINT MS(8): PRINT
: PRINT "TO WHAT FILENAME DO YOU
WISH TO SAVE THE DATA?": INPUT F
S: IF FS = " " THEN RETURN
1340 ONERR GOTO 1430
1350 PRINT DS: "OPEN": FS
1360 PRINT DS: "CLOSE": FS
1370 PRINT DS: "DELETE": FS
1380 PRINT DS: "OPEN": FS
1390 PRINT DS: "WRITE": FS
1400 PRINT R: FOR I = 1 TO R: PRINT AS(I)
: NEXT I
1410 PRINT DS: "CLOSE": FS
1420 POKE 216, 0: GOTO 250
1430 REM ERROR HANDLING ROUTINE
1440 ER = PEEK(222): POKE 216, 0
1450 IF ER = 13 THEN PRINT "FILE EXISTS
BUT IS NOT A TEXT FILE"
1460 IF ER = 9 THEN PRINT "DISK FULL"
1470 IF ER = 10 THEN PRINT "FILE LOCKED
: SAVE WITH A DIFFERENT NAME"
1480 IF ER = 4 THEN PRINT "DISK IS WRIT
E PROTECTED: REMOVE TAB"
1490 PRINT: PRINT "PRESS <RETURN> TO TR
Y AGAIN: PRESS <ESC> TO
GO BACK TO MENU": GET KS
1500 IF KS = CHR$(13) THEN GOTO 1330
1510 IF KS = CHR$(27) THEN GOTO 250
1520 GOTO 1490
1530 REM END PROGRAM
1540 HOME: VTAB 2: PRINT "ARE YOU SUR
E YOU WANT TO EXIT THE PROGRA
M AND ERASE THE CONTENTS OF
MEMORY? (Y/N)"
GET KS: IF KS < "Y" AND KS > "
N" THEN 1550
1560 IF KS = "N" THEN RETURN
1570 HOME: END
1580 DATA TAX DEDUCTION FILER, ADD DATA
, CHANGE DATA, DISPLAY DATA, TOTALS, PR
INT REPORT, LOAD DATA FILE, SAVE DATA
FILE, EXIT PROGRAM, YOUR CHOICE
1590 DATA MEDICINE AND DRUGS, "DOCTORS,
DENTISTS, ETC.", MEDICAL TRANSPORTAT
ION, OTHER MEDICAL, STATE AND LOCAL I
NCOME TAX, REAL ESTATE, MOTOR VEHICLE
SALES TAX, OTHER TAXES
1600 DATA HOME MORTGAGE INTEREST, CREDI
T CARD INTEREST, OTHER INTEREST, CASH
CONTRIBUTIONS, OTHER CONTRIBUTIONS,
CASUALTY AND THEFT, UNION AND PROFES
SIONAL DUES, TAX PREPARATION FEES, MI
SCellaneous

```

HCM

TAX DEDUCTION

COMMODORE 64

```

100 REM ** TAX DEDUCTION FILER **
110 REM ** **
120 REM ** **
130 REM BY THE HCM STAFF
140 REM HOME COMPUTER MAGAZINE
150 REM VERSION 4.4.1
160 REM C-64 BASIC
170 REM
180 DIM AS(500), T(17), NS(17)
190 FOR Z = 1 TO 10: READ MS(Z): NEXT Z
200 REM TITLE SCREEN
210 PRINT "SHIFT CLR 7CRSRDOWN"
: MS(1)
220 PRINT "6CRSRDOWN" PRESS AN
Y KEY TO BEGIN
230 GET KS: IF KS = " " THEN 230
240 PRINT "SHIFT CLR 2CRSRDOWN"
: MS(1)
250 FOR Z = 1 TO 8: PRINT "CRSRDOWN":
Z: " ": MS(Z+1): NEXT Z: PRINT "CRSRDOW
N": MS(10)
260 GET CH$: IF CH$ = " " THEN 260
270 IF CH$ < "1" OR CH$ > "8" THEN 260
280 ON VAL(CH$) GOSUB 350, 460, 850, 10
10, 1090, 1420, 1590, 1840
290 GOTO 240
300 REM DISPLAY CATEGORIES
310 FOR Z = 1 TO 17: PRINT " ": Z: " ": NS(Z)
: NEXT Z: RETURN
320 IF AC = 0 THEN RETURN
330 AC = ASC(AS(Z)) - 100: DS = MID$(AS(Z), 3, A
SC(MID$(AS(Z), 2, 1)) - 100)

```

```

340 VS = RIGHTS(AS(Z), LEN(AS(Z))) - (ASC(MID
$(AS(Z), 2, 1)) - 98)): RETURN
350 REM ADD DATA
360 PRINT "SHIFT CLR": GOSUB 300: PRIN
T "CRSRDOWN YOUR CHOICE"
370 CS = " ": INPUT CS: C = VAL(C$): IF CS = " " T
HEN RETURN
380 IF C < 1 OR C > 17 THEN PRINT "SHIFT C
RSRUP": PRINT
390 PRINT "2SHIFT CRSRUP": GOTO 370
400 PRINT "ENTER ITEM DESCRIPTION": DS = " ": IN
PUT DS
410 IF DS = " " THEN RETURN
420 IF LEN(DS) > 27 THEN PRINT "SHIFT CRS
RUP"
430 PRINT "SHIFT CRSRUP": GOTO 390
440 INPUT V: IF V = 0 THEN RETURN
450 VS = STR$(V)
460 R = R + 1: AS(R) = CHR$(100 + C) + CHR$(LEN(DS)
+ 100) + DS + VS: GOTO 360
470 REM CHANGE DATA
480 PRINT "SHIFT CLR"
490 PRINT "CRSRDOWN THERE IS NO DATA
IN MEMORY TO CHANGE": FOR I = 1 TO 100
: NEXT I
RETURN

```

Continued


```

500 GOSUB 300:PRINT "YOUR CHOICE"
510 CS="":INPUT CS:C=VAL(C$):IF CS="" T
HEN RETURN
520 IF C<1 OR C>17 THEN PRINT "SHIFT C
RSRUP":PRINT "2SHIFT CRSRUP":GOTO 470
530 PRINT "SHIFT CLR":TAB(20-LEN(NS(C
)))/2)NS(C)
540 N=0:FOR Z=1 TO R
550 IF ASC(AS(Z))<>C+100 THEN 610
560 N=N+1:GOSUB 320
570 PRINT Z:DS=TAB(28)"$":VS
580 IF N/20<>INT(N/20) THEN 610
590 PRINT "PRESS RETURN TO CONTINUE"
600 GET K$:IF K$<>CHR$(13) THEN 600
610 NEXT Z:IF N=0 THEN PRINT "CRSRDOWN
NO DATA IN THIS CATEGORY"
620 PRINT "CRSRDOWN CHANGE WHICH ITEM
?"
630 CHS="":INPUT CHS:IF CHS="" THEN RET
URN
640 GOSUB 780:IF CH=0 THEN 470
650 Z=CH:GOSUB 320:IF AC=C THEN 680
660 PRINT "THE RECORD YOU SELECTED IS N
OT IN THIS CATEGORY"
670 PRINT "CRSRDOWN TRY AGAIN":FOR DE=
1 TO 1000:NEXT DE:GOTO 470
680 PRINT "CRSRDOWN EITHER: ENTER /D/
TO DELETE ITEM
690 PRINT "PRESS RETURN TO LEAV
E ENTRY
700 PRINT "ENTER NEW DATA AT PR
OMPT CRSRDOWN"
710 PRINT "OLD DATA: ";DS:PRINT "NEW DA
TA:":INPUT DS
720 IF LEN(DS)>27 THEN PRINT "SHIFT CRS
RUP":SHIFT CRSRUP:GOTO 710
730 IF DS<>"/D/" THEN 750
740 FOR I=CH TO R-1:AS(I)=AS(I+1):NEXT:
AS(R)=DS:R=R-1:GOTO 470
750 PRINT "OLD AMOUNT: ";VS:PRINT "NEW
AMOUNT:":INPUT VS
760 VS=STR$(VAL(VS))
770 AS(CH)=CHR$(100+AC)+CHR$(LEN(DS)+10
0)+DS+VS:GOTO 470
780 REM SEARCH FOR RECORD
790 IF ABS(ASC(CHS)-53)>4 THEN 820
800 CH=VAL(CHS):IF CH>R THEN PRINT "INV
ALID RECORD NUMBER":CH=0:GOTO 840
810 RETURN
820 FOR I=1 TO R:CH=I:IF CHS=MID$(AS(I)
,3,LEN(CHS)) THEN RETURN
830 NEXT I:PRINT "RECORD NOT FOUND":CH=
0
840 FOR DE=1 TO 2000:NEXT:RETURN
850 REM DISPLAY DATA
860 PRINT "SHIFT CLR":GOSUB 300:DISP
LAY DATA "CRSRDOWN YOUR CHOICE"
870 CS="":INPUT CS:C=VAL(C$):IF CS="" T
HEN RETURN
880 IF C<1 OR C>17 THEN PRINT "SHIFT C
RSRUP":PRINT "2SHIFT CRSRUP":GOTO 870
890 PRINT "SHIFT CLR":TAB(20-LEN(NS(C
)))/2)NS(C)
900 IF AS(1)= THEN PRINT "2CRSRDOWN
NO DATA IN MEMORY":FOR DE=1 TO 200
0:NEXT:RETURN
910 N=0:FOR Z=1 TO R:IF ASC(AS(Z))<>C+1
00 THEN 970
920 N=N+1:GOSUB 320
930 PRINT Z:DS=TAB(28)"$":VS
940 IF N/20<>INT(N/20) THEN 970
950 PRINT "CRSRDOWN PRESS RETURN TO C
ONTINUE"
960 GET K$:IF K$<>CHR$(13) THEN 960
970 NEXT Z:IF N=0 THEN PRINT "CRSRDOWN
NO DATA IN THIS CATEGORY"
980 PRINT "CRSRDOWN PRESS RETURN TO C
ONTINUE"
990 GET K$:IF K$<>CHR$(13) THEN 990
1000 GOTO 860
1010 REM TOTALS
1020 PRINT "SHIFT CLR":* TO
TALS "CRSRDOWN"
1030 FOR I=1 TO 17:T(I)=0:NEXT I
1040 FOR Z=1 TO R:GOSUB 320:T(AC)=T(AC)
+VAL(VS):NEXT Z
1050 FOR Z=1 TO 17:PRINT NS(Z)TAB(28)"
$:T(Z):NEXT
1060 PRINT "CRSRDOWN PRESS RETURN TO GO
BACK TO MENU"
1070 GET K$:IF K$<>CHR$(13) THEN 1070
1080 RETURN
1090 REM PRINT DATA
1100 PRINT "SHIFT CLR":* LIST
TO PRINTER "
1110 PRINT "2CRSRDOWN 1) TOTALS CRSRD
OWN:PRINT 2) ALL RECORDS IN A
CATEGORY CRSRDOWN"
1120 PRINT "3) PRINT ALL RECORDS CRSRD
OWN:PRINT YOUR CHOICE:
1130 K$="":INPUT K$:IF K$= THEN RETURN
1140 IF K$<1 OR K$>3 THEN PRINT "2S
HIFT CRSRUP":GOTO 1130
1150 IF AS(1)= THEN PRINT "2CRSRDOWN
NO DATA IN MEMORY":FOR DE=1 TO 200
0:NEXT:RETURN

```

```

1160 ON VAL(K$) GOTO 1170, 1230, 1360
1170 OPEN 4,4:PRINT #4,"* TOT
ALS *":CHR$(13)
1180 PRINT #4,"CATEGORY":CHR$(16)"10DESCR
IPTION":CHR$(16)"40AMOUNT":CHR$(13)
1190 FOR I=1 TO 17:T(I)=0:NEXT
1200 FOR Z=1 TO R:GOSUB 320:T(AC)=T(AC)
+VAL(VS):NEXT Z
1210 FOR Z=1 TO 17:PRINT #4,Z:CHR$(16)"10
":NS(Z):CHR$(16)"40$":T(Z):NEXT
1220 PRINT #4,":CLOSE4:GOTO 1100
1230 PRINT "SHIFT CLR":* PRINT
A CATEGORY "CRSRDOWN"
1240 GOSUB 300:PRINT "CRSRDOWN YOUR C
HOICE:":INPUT CS:C=VAL(C$):IF CS="" T
HEN RETURN
1250 IF C<1 OR C>17 THEN PRINT "SHIFT C
RSRUP":PRINT "2SHIFT CRSRUP":GOTO 1250
1270 OPEN 4,4:CHR$(13):CHR$(13):CHR$(16)"
1280 12":NS(C):CHR$(13)
1290 PRINT #4,"RECORD":CHR$(16)"10DESCRIP
TION":CHR$(16)"40AMOUNT":CHR$(13)
1300 N=0:FOR Z=1 TO R:IF ASC(AS(Z))<>C+1
00 THEN 1330
1310 N=N+1:GOSUB 320
1320 PRINT #4,Z:CHR$(16)"10":DS:CHR$(
16)"40$":VS
1330 NEXT Z:IF N=0 THEN PRINT #4,CHR$(13)
:PRINT #4,":CLOSE4
1340 GOTO 1230
1350 OPEN 4,4:PRINT #4,CHR$(13):
1360 ALL CATEGORIES":CHR$(13)
1370 PRINT #4,"RECORD":CHR$(16)"08CATEGO
RY":CHR$(16)"20DESCRIPTION":CHR$(13)
1380 PRINT #4,CHR$(16)"50AMOUNT":CHR$(13)
1390 FOR Z=1 TO R:GOSUB 320:PRINT #4,Z:
CHR$(16)"08":AC:CHR$(16)"20":D
S:PRINT #4,CHR$(16)"50$":VS:NEXT Z
1400 PRINT #4,":CLOSE4:GOTO 1060
1410 REM LOAD DATA FILE
1420 PRINT "SHIFT CLR:2CRSRDOWN DO YO
U WISH TO LOAD DATA FROM:
1430 PRINT "CRSRDOWN 1) TAPE":PRINT "
CRSRDOWN 2) DISK":PRINT "
TF=0:INPUT TF:IF TF<1 OR TF>2 THEN
1440 PRINT "2SHIFT CRSRUP":GOTO 1420
1450 PRINT "2CRSRDOWN WHAT DATA FILE
DO YOU WISH TO LOAD?
1460 F2$="":INPUT F2$:IF F2$="" THEN RET
URN
1470 ON TF GOTO 1490,1500
1480 OPEN 2,1,0,F2$:GOTO 1540
1490 F3$="S,R":F2$:GOTO 1540
1500 OPEN 15,8,15,"10"
1510 OPEN 2,8,F2$+F3$
1520 GOSUB 1770:IF EN>20 THEN PRINT "
SHIFT CLR":GOTO 1430
1530 PRINT "LOADING DATA"
1540 INPUT #2,R
1550 FOR Z=1 TO R:INPUT #2,AS(Z):NEXT Z
1560 IF TF=2 THEN CLOSE15
1570 CLOSE2:RETURN
1580 REM SAVE DATA FILE
1590 PRINT "SHIFT CLR:2CRSRDOWN DO Y
OU WISH TO SAVE DATA FILE TO:
1600 PRINT "CRSRDOWN 1) TAPE":PRINT "
CRSRDOWN 2) DISK":PRINT "
TF=0:INPUT TF:IF TF<1 OR TF>2 THEN
1610 PRINT "2SHIFT CRSRUP":GOTO 1590
1620 PRINT "2CRSRDOWN TO WHAT FILE NA
ME DO YOU WISH TO SAVE THE DATA?
1630 F2$="":INPUT F2$:IF F2$="" THEN RET
URN
1640 ON TF GOTO 1660,1670
1650 OPEN 2,1,1,F2$:GOTO 1720
1660 F1$="S,W":F3$="S,W"
1670 OPEN 15,8,15,"10"
1680 OPEN 2,8,F1$+F2$+F3$
1690 GOSUB 1770:IF EN>20 AND EN<>63 TH
ENPRINT "SHIFT CLR":GOTO 1600
1700 IF EN=63 THEN 1680
1710 PRINT "SAVING DATA"
1720 PRINT #2,R
1730 FOR Z=1 TO R:PRINT #2,AS(Z):NEXT Z
1740 IF TF=2 THEN CLOSE15
1750 CLOSE2:RETURN
1760 REM DISK ERROR ROUTINE
1770 INPUT #15,EN,EMS,ET,ES:IF EN<20 THEN
1780 PRINT "TRANSFER STARTED":RETURN
1790 IF EN=63 THEN F1$="@":CLOSE2:CLOS
E15:RETURN
1800 PRINT "DISK ERROR #":EN,EMS,ET,ES
1810 IF EN=62 THEN PRINT "PRESS ANY KEY
TO TRY AGAIN"
1820 GET K$:IF K$="" THEN 1820
1830 CLOSE 2:CLOSE15:RETURN
1840 REM END PROGRAM
1850 PRINT "SHIFT CLR:2CRSRDOWN ARE
YOU SURE YOU WANT TO EXIT THE
1860 PRINT "PROGRAM AND ERASE THE CONT
ENTS OF MEMORY? (Y/N)"

```

Continued


```

1870 GET K$:IF K$<>"Y" AND K$<>"N" THEN
1880 IF K$="N" THEN RETURN
1890 IF K$="Y" THEN PRINT "SHIFT CLR":
END
PRINT "SHIFT CLR":END
1900 DATA TAX DEDUCTION FILER,ADD DATA,C
1910 HANGE DATA,DISPLAY DATA,TOTALS
1920 DATA PRINT REPORT,LOAD DATA FILE,SA
VE DATA FILE,EXIT PROGRAM,YOUR CHOI
CE
1930 DATA MEDICINE AND DRUGS,"DOCTORS,DE
NTISTS,ETC."

```

```

1940 DATA MEDICAL,STATE AND TRANSPORTATION,OTHER M
EDICAL,REAL ESTATE TAX,LOCAL INCOME TAX
1950 DATA SALES TAX,OTHER TAXES,MOTOR VEHICLE
1960 DATA HOME MORTGAGE INTEREST,CREDIT
CARD INTEREST,OTHER INTEREST
1970 DATA CASH CONTRIBUTIONS,OTHER CONTR
IBUTIONS,CASUALTY AND THEFT
1980 DATA UNION AND PROF. DUES,TAX PREPA
RATION FEES,MISCELLANEOUS

```

HCM

TAX DEDUCTION

IBM PC & PCjr

```

100 **TAX DEDUCTION FILER**
110 **BY THE HCM STAFF**
120 **VERSION 4.4.1**
130 **IBM PCjr - CASSETTE & CARTRIDGE BA
140 SIC**
150 **IBM PC - CASSETTE BASIC & BASICA
160 DIM AS(500),T(17),NS(17)
170 FOR Z=1 TO 10:READ MS(Z):NEXT FOR Z
180 1 TO 17:READ NS(Z):NEXT FOR Z
190 CLS:KEY OFF:LOCATE 12,10:PRINT MS(1)
):LOCATE 23,9:PRINT "PRESS ANY KEY
TO BEGIN"
200 K$=INKEY$:IF K$=" " THEN 200
210 REM MAIN MENU
220 CLS:LOCATE 2,10:PRINT MS(1):LOCATE
6,1:FOR I=1 TO 8:PRINT TAB(6) I;"
MS(1+I):PRINT NEXT I:PRINT:PRINT
TAB(6) MS(10)
230 K$=INKEY$:IF K$=" " THEN 230
240 IF K$<"1" OR K$>"8" THEN 230
250 ON VAL(K$) GOSUB 310,460,670,740,79
0,960,1070,1180
260 GOTO 220
270 REM DISPLAY CATEGORIES
280 FOR Z=1 TO 17:PRINT " ";Z;" ";NS(Z)
):NEXT Z:RETURN
290 IF R=0 THEN RETURN
300 AC=ASC(AS(Z))-100:DS=MID$(AS(Z),3,AC
SC(MID$(AS(Z),2,1))-100):VS=RIGHT$(
AS(Z),LEN(AS(Z))-(ASC(MID$(AS(Z),2,
1))-98)):RETURN
310 REM ADD DATA
320 CLS:LOCATE 2,13:PRINT MS(2):PRINT:G
OSUB 270:PRINT:PRINT MS(10)
330 LOCATE 23,1:CS="":INPUT CS:IF CS=" "
THEN RETURN ELSE C=VAL(CS):IF C<1
OR C>17 THEN 330
340 CLS:LOCATE 2,15:PRINT MS(2):PRINT:P
RINT NS(C):PRINT:PRINT "ITEM DESCRIP
TION: ";DS:"":INPUT DS:IF DS=" " TH
EN RETURN
350 IF LEN(DS)>27 THEN 340
360 PRINT "ENTER ITEM VALUE: ";V=0:INPU
T V:IF V=0 THEN RETURN ELSE VS=STR$(
T(V)-R+1:AS(R)=CHR$(100+C)+CHR$(LE
N(DS)+100)+DS+VS:GOTO 310
370 N=0:FOR Z=1 TO R:IF ASC(AS(Z))<C+1
N=0 THEN 400
380 N=N+1:GOSUB 290:PRINT Z;" ";DS TAB
(30) $:VS:IF N/20<>INT(N/20) THEN
400 ELSE PRINT "PRESS ENTER TO
CONTINUE"
390 K$=INKEY$:IF INKEY$<>CHR$(13) THEN
390
400 NEXT Z:IF N=0 THEN PRINT " NO DATA
IN THIS CATEGORY"
410 PRINT:PRINT "PRESS ENTER TO CONTI
NUE"
420 K$=INKEY$:IF K$<>CHR$(13) THEN 420
430 ELSE RETURN
440 DATA TAX DEDUCTION FILER,ADD DATA,C
HANGE DATA,DISPLAY DATA,TOTALS,PRIN
T REPORT,LOAD DATA FILE,SAVE DATA F
ILE,EXIT PROGRAM,YOUR CHOICE
440 DATA MEDICINE AND DRUGS,"DOCTORS, D
ENTISTS, ETC.",MEDICAL TRANSPORTATI
ON,OTHER MEDICAL,STATE AND LOCAL IN
COME TAX,REAL ESTATE TAX,MOTOR VEHI
CLE SALES TAX,OTHER TAXES
450 DATA HOME MORTGAGE INTEREST,CREDIT
CARD INTEREST,OTHER INTEREST,CASH C
ONTRIBUTIONS,OTHER CONTRIBUTIONS,CA
SUALTY AND THEFT,UNION AND PROF. DU
ES,TAX PREPARATION FEES,MISCELLANEO
US
460 REM CHANGE DATA
470 CLS:LOCATE 2,17:PRINT MS(3):PRINT:G
OSUB 270:PRINT:PRINT MS(10)
480 CS="":INPUT CS:C=VAL(CS):IF CS=" " T
HEN RETURN
490 IF C<1 OR C>17 THEN 470
500 CLS:LOCATE 2,(20-LEN(NS(C)))/2):PRIN
T NS(C)
510 LOCATE 4,1:IF AS(1)=" " THEN PRINT "
NO DATA IN MEMORY":FOR DELAY=1 TO 2
000:NEXT:RETURN
520 GOSUB 370:IF N=0 THEN GOTO 470
530 PRINT:PRINT "CHANGE WHICH ITEM?":CH
$="":INPUT CH$:IF CH$=" " THEN RETUR

```

```

540 GOSUB 620:IF CH=0 THEN 470
550 Z=CH:GOSUB 290:IF C<>AC THEN PRINT IN
"THIS RECORD YOU REQUESTED IS NOT IN
THE CATEGORY":PRINT:TRY AG
AIN:FOR DELAY=1 TO 1000:NEXT DELAY
560 PRINT:PRINT "EITHER: ENTER /D/ TO D
ELETE ITEM:PRINT "PRESS RE
TURN TO LEAVE ENTRY ENTER NEW DA
TA AT PROMPT:PRINT
PRINT "OLD DATA: ";DS:PRINT "NEW DA
TA: ";INPUT ND$:DS:PRINT "NEW DA
IF LEN(ND$)>27 THEN 570 ELSE IF ND$
=/D/ THEN 590 ELSE IF ND$<>" " THE
N FOR I=CH:GOTO 600 ELSE 600
590 I:AS(R) " :R=R-1:GOTO 470
PRINT: "OLD AMOUNT: ";VS:PRINT "NEW
AMOUNT: ";INPUT NV:IF NV=0 THEN 610
ELSE VS=STR$(NV)
610 AS(CH)=CHR$(100+AC)+CHR$(LEN(DS)+10
0)+DS+VS:GOTO 470
620 REM SEARCH FOR RECORD
630 IF ABS(ASC(CH$)-53)>4 THEN 650
640 CH=VAL(CH$):IF CH>R THEN PRINT "INV
ALID RECORD NUMBER":CH=0:GOTO 660 E
LSE RETURN
650 FOR I=1 TO R:CH=I:IF CH$=MID$(AS(I)
,3,LEN(CH$)) THEN RETURN ELSE NEXT
I:PRINT:RECORD NOT FOUND:CH=0
660 FOR DELAY=1 TO 2000:RETURN
670 REM DISPLAY DATA
680 CLS:LOCATE 2,14:PRINT MS(4):PRINT:G
OSUB 270:PRINT:PRINT MS(10)
690 CS="":INPUT CS:C=VAL(CS):IF CS=" " T
HEN RETURN
700 IF C<1 OR C>17 THEN 680
710 CLS:LOCATE 2,(20-LEN(NS(C)))/2):PRIN
T NS(C)
720 LOCATE 4,1:IF AS(1)=" " THEN PRINT "
NO DATA IN MEMORY":FOR DELAY=1 TO 2
000:NEXT:RETURN
730 GOSUB 370:GOTO 680
740 REM TOTALS
750 CLS:LOCATE 2,15:PRINT MS(5):FOR I=1
TO 17:T(I)=0:NEXT I
760 FOR Z=1 TO R:GOSUB 290:T(AC)=T(AC)+
VAL(VS):NEXT Z
770 LOCATE 4,1:FOR Z=1 TO 17:PRINT NS(Z)
)TAB(28) $:T(Z):NEXT:PRINT:PRINT
"PRESS ENTER TO RETURN TO MENU":
780 K$=INKEY$:IF K$<>CHR$(13) THEN 780
790 ELSE RETURN
800 REM PRINT REPORT
810 CLS:LOCATE 2,15:PRINT MS(6):PRINT:P
RINT " 1) TOTALS":PRINT:PRINT " 2
) ALL RECORDS IN A CATEGORY":PRINT
:PRINT " 3) PRINT ALL RECORDS":PRI
NT:PRINT MS(10)
810 CH$=INKEY$:IF CH$=CHR$(13) THEN RET
URN ELSE IF CH$<"1" OR CH$>"3" THEN
820 IF AS(1)=" " THEN PRINT:PRINT "THER
E IS NO DATA IN MEMORY":FOR DELAY=1
TO 2000:NEXT:RETURN
830 ON VAL(CH$) GOTO 840,870,940
840 FOR I=1 TO 17:T(I)=0:NEXT I:FOR Z=1
TO R:GOSUB 290:T(AC)=T(AC)+VAL(VS)
:NEXT Z
850 LPRINT TAB(20) MS(6):CHR$(13):CHR$(
13):"CATEGORY" TAB(10) "DESCRIPTION"
TAB(40) "AMOUNT":CHR$(13)
860 FOR I=1 TO 17:LPRINT I;" TAB(10)
NS(I) TAB(40) $:T(I):NEXT I:GOTO
800
870 CLS:LOCATE 2,10:PRINT "PRINT A CATE
GORY":PRINT:GOSUB 270:PRINT:PRINT
MS(10)
880 INPUT CH$:IF CH$=" " THEN 800 ELSE C
H=VAL(CH$):IF CH<1 OR CH>17 THEN 89
0
890 LPRINT TAB(12) NS(CH):CHR$(13):CHR$(
13):"RECORD" TAB(10) "DESCRIPTION"
TAB(40) "AMOUNT"
900 N=0:FOR Z=1 TO R:IF ASC(AS(Z))<>CH+
100 THEN 920 ELSE N=N+1:GOSUB 290
910 LPRINT Z;" TAB(10) DS TAB(40) $
VS

```

Continued


```

920 NEXT Z:IF N=0 THEN LPRINT "NO DATA"
930 IN THIS CATEGORY"
940 GOTO 800
LPRINT TAB(15); "ALL CATEGORIES"; CHR
$(13); CHR$(13); "RECORD" TAB(8); CAT
EGORY; TAB(20); "DESCRIPTION" TAB(50
) "AMOUNT"
950 FOR Z=1 TO R:GOSUB 290:LPRINT Z TAB
(10); AC TAB(20); DS TAB(50); "$"; V$:N
EXT Z:GOTO 800
960 REM LOAD FILE
970 CLS:LOCATE 2,2:PRINT "DO YOU WISH T
O LOAD DATA FROM:";LOCATE 4,2:PRIN
T "1) TAPE";LOCATE 6,2:PRINT "2) DI
SK"
980 INPUT "YOUR CHOICE";TF:IF TF<1 OR T
F>2 THEN 970
990 IF TF=2 THEN F1$="A":GOTO 1010
1000 F1$="CAS1":PRINT "PLEASE REWIND
CASSETTE AND PRESS"
1010 F2$="":PRINT "WHAT DATA FI
LE DO YOU WISH TO LOAD?";INPUT F2$
:IF F2$="" THEN RETURN
1020 OPEN "1",#2,F1$+F2$
1030 INPUT #2,R
1040 FOR I=1 TO R
1050 INPUT #2,AS(I):NEXT I
1060 CLOSE #2:RETURN
1070 REM SAVE FILE

```

```

1080 CLS:LOCATE 2,2:PRINT "DO YOU WISH T
O SAVE DATA FILE TO:";LOCATE 4,2:P
RINT "1) TAPE";LOCATE 6,2:PRINT "2)
DISK"
1090 INPUT "YOUR CHOICE";TF:IF TF<1 OR T
F>2 THEN 1080
1100 IF TF=2 THEN F1$="A":GOTO 1120
1110 F1$="CAS1":PRINT "PLEASE REWIND
CASSETTE AND PRESS"
1120 F2$="":PRINT "TO WHAT FILE
NAME?";INPUT F2$:IF F2$="" THEN RETUR
N
1130 OPEN "O",#2,F1$+F2$
1140 WRITE #2,R
1150 FOR I=1 TO R
1160 WRITE #2,AS(I):NEXT I
1170 CLOSE #2:RETURN
1180 REM END PROGRAM
1190 CLS:LOCATE 2,2:PRINT "ARE YOU SUR
E YOU WANT TO EXIT THE PROGRA
M AND ERASE THE CONTENTS OF
MEMORY? (Y/N)"
1200 K$=INKEY$:IF K$<"Y" AND K$<"N" THEN 1200:EL
SE IF K$="Y" OR K$="N" THEN RETURN
ELSE CLS:END

```

HCM

TAX DEDUCTION

TI-99/4A

```

100 REM *** TAX DEDUCTION FILER ***
110 REM *****
120 REM BY THE HCM STAFF
130 REM HOME COMPUTER MAGAZINE
140 REM VERSION 4.4.1
150 REM TI EXTENDED BASIC
160 REM
170 REM
180 OPTION BASE 1::DIM AS$(500),T(17),
NS(17)
190 FOR Z=1 TO 8::READ MS(Z)::NEXT Z
::FOR Z=1 TO 17::READ NS(Z)::N
EXT Z
200 CALL CLEAR::DISPLAY AT(1,10):"MAI
N MENU":FOR Z=1 TO 8::DISP
LAY AT(2+Z*2,1):STR$(Z);":MS$(Z)::NE
XT Z
210 GOSUB 860::IF K<49 OR K>56 THEN C
ALL SOUND(50,220,0)::GOTO 210
220 CALL CLEAR::ON K-48 GOTO 230,290,
450,510,530,690,750,940
230 GOSUB 820
240 DISPLAY AT(20,1):"YOUR CHOICE:"::
ACCEPT AT(20,14)VALIDATE(DIGIT)SIZE
(2):CS::IF CS="" THEN 200 ELSE C=
VAL(CS)
250 IF C<1 OR C>17 THEN CALL SOUND(50,2
20,0)::GOTO 240
260 DISPLAY AT(20,1):"ENTER ITEM DESCI
PTION":ACCEPT AT(21,1):DS::IF
DS="" THEN 200
270 DISPLAY AT(23,1):"ENTER ITEM VALUE:
C":ACCEPT AT(24,1)VALIDATE(NUMERI
C)::VS::IF VS="" THEN 200
280 R=R+1::AS(R)=CHR$(100+C)&CHR$(LEN
(DS)+100)&DS&VS::CALL HCHAR(20,1,
32,160)::GOTO 230
290 GOSUB 820
300 DISPLAY AT(20,1):"YOUR CHOICE:"::
ACCEPT AT(20,14)VALIDATE(DIGIT)SIZE
(2):CS::IF CS="" THEN 200 ELSE C=
VAL(CS)
310 IF C<1 OR C>17 THEN CALL SOUND(50,2
20,0)::GOTO 300
320 CALL CLEAR::L=1::FOR Z=1 TO R::
GOSUB 810::IF C=AC THEN DISPLAY
AT(L,1):STR$(Z);":DS";":VS":
L=L+2
330 IF L/20<>INT(L/18)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="" THEN 200 ELSE E=V
AL(ES)::IF E>R OR E<1 THEN CALL SO
UND(50,220,0)::GOTO 340
350 Z=E::GOSUB 810::IF AC<>C THEN 3
40 ELSE CALL HCHAR(20,1,32,160)::D
ISPLAY AT(20,1):"ENTER NEW I
TEM"
360 DISPLAY AT(21,1):"PRESS ENTER TO NO
T CHANGE":OR /D/ TO DELETE:"ENTER
NEW ITEM FOR #":E::ACCEPT AT(24,
1):DN$
370 IF DN$="" THEN 410
380 IF DN$<>"/D/" THEN 400
390 FOR N=E TO R::AS(N)=AS(N+1)::NEX
T N::R=R-1::GOTO 200
400 DS=DN$
410 CALL HCHAR(20,1,32,160)::DISPLAY A
T(23,1):"ENTER NEW VALUE FOR #":E:
ACCEPT AT(24,1)VALIDATE(NUMERIC):
NV$
420 IF NV$<>" " THEN VS=NV$
430 AS(E)=CHR$(100+C)&CHR$(LEN(DS)+100)
&DS&VS::GOTO 200

```

```

440 NEXT Z::GOTO 290
450 GOSUB 820
460 DISPLAY AT(20,1):"YOUR CHOICE:"::
ACCEPT AT(20,14)VALIDATE(DIGIT)SIZE
(2):CS::IF CS="" THEN 200 ELSE C=
VAL(CS)
470 IF C<1 OR C>17 THEN CALL SOUND(50,2
20,0)::GOTO 460
480 CALL CLEAR::L=1::FOR Z=1 TO R::
GOSUB 810::IF C=AC THEN DISPLAY
AT(L,1):STR$(Z);":DS";":VS":
L=L+2
490 IF L/20<>INT(L/18)AND Z<>R THEN 440
ELSE DISPLAY AT(20,1):"PRESS ENTER
FOR MORE":GOSUB 860
500 NEXT Z::GOTO 200
510 FOR Z=1 TO 17::T(Z)=0::NEXT Z::
FOR Z=1 TO R::GOSUB 810::V=VA
L(VS)::T(AC)=T(AC)+V::NEXT Z::
CALL CLEAR
520 FOR Z=1 TO 17::DISPLAY AT(Z+2,1):
NS(Z):TAB(21):STR$(T(Z))::NEXT Z::
DISPLAY AT(24,1):"PRESS ENTER"::
GOSUB 860
530 DISPLAY AT(1,1):"ENTER PRINTER DEVI
CE":DV$:ACCEPT AT(2,1)SIZE(-28):
DV$::OPEN #2:DV$
540 DISPLAY AT(4,1):"ENTER YOUR CHOICE:
1) TOTALS 2) ALL RECORDS I
N CATEGORY":IF K=13 THEN
GOSUB 860
550 IF K=13 THEN RECORDS CLOSE #2
::GOTO 200 ELSE IF K<49 OR K>51 TH
EN CALL SOUND(50,220,0)::GOTO 550
560 DISPLAY AT(15,1):"ENTER TODAY'S DATE
":DT$:ACCEPT AT(16,1)SIZE(-28):
DT$::DISPLAY AT(18,1):"ENTER REPO
RT TITLE":RT$:RT$:PRI
NT #2::DEDUCTION FILER REPORT
::TAB(40):RT$:DT$:ON K-48
GOTO 580,610,670
580 PRINT #2:"TOTALS REPORT":"CATEGOR
Y"
590 FOR Z=1 TO R::GOSUB 810::T(AC)=
T(AC)+VAL(VS)::NEXT Z::FOR Z=1 T
O 17::PRINT #2:NS(Z):TAB(26);"$";
T(Z)::NEXT Z
600 CLOSE #2::GOTO 200
610 CALL CLEAR::K=53::GOSUB 820::
DISPLAY AT(20,1):"WHICH CATEGORY?"
620 ACCEPT AT(20,17)VALIDATE(DIGIT)SIZE
(3):CS::IF CS="" THEN CLOSE #2::
GOTO 200
630 C=VAL(CS)::IF C<1 OR C>17 THEN CAL
L SOUND(50,220,0)::GOTO 620
640 PRINT #2:"CATEGORY RECORDS FOR ";NS
(C):"AMOUNT"
650 FOR Z=1 TO R::IF C=ASC(AS(Z))-100
THEN GOSUB 810::PRINT #2:STR$(Z)
:":DS:TAB(36);":VS
660 NEXT Z::CLOSE #2::GOTO 200
670 PRINT #2:"ALL CATEGORIES":"RECO
RD CATEGORY ITEM DESCRIPTION";TAB
(50);"AMOUNT":GOSUB 810::PRINT
#2:STR$(Z);":":TAB(9);AC:TAB(19);DS
:TAB(50);":VS::NEXT Z::CLOSE
#2::GOTO 200
690 DISPLAY AT(1,7):MS(6):"1) CS1 - C
ASSETTE"
700 GOSUB 860::IF K<49 OR K>51 THEN 7
00 ELSE ON K-48 GOTO 710,720,730

```

Continued

SENSATIONAL SOFTWARE GIVEAWAY

Proof Of Purchase Necessary

1 YOUR CHOICE OF THIS ISSUE'S PROGRAMS AT A REMARKABLE PRICE—



ONLY \$3.95!

To participate in our monthly Software Giveaway, you need to be a bona-fide purchaser of **Home Computer Magazine** and fill out the questionnaire on the reverse side.

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 \$3.95 (\$5.95 in Canada & Mexico). 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.

*FREE SOFTWARE!

2 When Subscribing To **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 months** of **ON TAPE** or **ON DISK**. Subscribe for 2 years and receive **4 months**. And with a 3-year subscription we'll give you **6 full months** 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 30% Off The Single-Copy Price Of The Magazine!

SAVE EVEN MORE!

3 AND ENJOY THE CONVENIENCE OF A PROGRAM SUBSCRIPTION

By subscribing to **ON TAPE™** or **ON DISK™** you will save money off the single-copy price **and** receive the same high-quality programs published monthly in the magazine—delivered right to your door **each and every month!** 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 _____ St _____ Zip _____

☐ Check or Money Order Enclosed

MUST BE IN U.S. FUNDS DRAWN ON A U.S. BANK

Bill my ☐ VISA ☐ MasterCard

Account No.

Signature _____

Exp. Date _____

Enclose payment or credit card information and mail entire page with completed form to:

HOME COMPUTER MAGAZINE
 P. O. Box 5537, Eugene, OR 97405
 -OR-

Use our TOLL-FREE LINE for Visa/MasterCard only:

1-800-828-2212 In Oregon, Alaska, Hawaii Tel.(503)485-8796

THIS SECTION MUST BE COMPLETED

INDICATE YOUR CHOICE OF MEDIA: ☐ ON TAPE™ ☐ ON DISK™

INDICATE WHICH COMPUTER MEDIA IS FOR: (pick one)

☐ Apple II family ☐ C-64 ☐ VIC-20 ☐ IBM PC ☐ PCjr ☐ TI 99/4A

☐ Please send a raincheck for the **FREE** software.

1 YES! Send me all the programs in this issue: **Vol. _____ No. _____** which have versions for my selected machine. I have indicated my choice of media and have enclosed \$3.95 (\$5.95 in Canada & Mexico). 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)
☐ 1-yr (12 issues) \$25 ☐ 2-yr (24 issues) \$45 ☐ 3-yr (36 issues) \$63
PLUS 2 FREE PLUS 4 FREE PLUS 6 FREE
 ON TAPE or ON DISK ON TAPE or ON DISK ON TAPE or ON DISK
 Canada add \$7; Foreign Surface add \$18 for 1-yr magazine subscription. Free software not available outside USA.

3 YES! I want to save time and money. Please enter my subscription to **ON TAPE** or **ON DISK** for the term listed below:

☐ 10 MONTHS—ONLY \$38 ☐ 12 MONTHS—ONLY \$45
 Canada add \$15 for software subscription.
 Software subscription not available in other countries at this time.

Please allow 6-8 weeks for your first issue—Magazine & Media shipped separately.

SGSO/9-84

Satisfaction Guaranteed—or the unfilled portion of your subscription will be refunded, less the cost of any premiums you have received.

3 FANTASTIC OFFERS

HOME COMPUTERTM magazine

QUESTIONNAIRE

Complete and mail to: Home Computer Magazine • P.O. Box 5537 • Eugene, Oregon 97405

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 Ile ☐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 COMPUTERTM magazine



Please Print

Name _____

Address _____

City _____ State _____ Zip _____

☐ Check or Money Order Enclosed Total _____

MUST BE IN US FUNDS DRAWN ON A US BANK

Bill my ☐ VISA ☐ MasterCard Date Expires _____

Account No. _____

Tel. No. _____ Signature _____

Enclose payment or credit card information & mail with completed form to:

Home Computer Magazine

P.O. Box 5537 • Eugene, OR 97405

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:

1-800-828-2212

In Oregon, Alaska, Hawaii Tel. (503) 485-8796

ITEMS	PRICE
Home Computer Magazine Back Issues (Circle Issues Desired)	\$3.95 each — U.S.
Vol. 4 No. 1 Vol. 4 No. 2	\$4.50 each — Canada
August, 1984 September, 1984	\$5.50 each — Foreign Surface
	\$7.50 each — Foreign Air
ON DISK & ON TAPE Back Issues (Circle Issues Desired)	\$5.95 each — U.S.
Vol. 4 No. 1 Vol. 4 No. 2	\$7.95 each — Canada
August, 1984 September, 1984	\$8.95 each — Foreign Surface
	\$9.95 each — Foreign Air
SAVE EVEN MORE — Order Combined Sets (Circle Magazine & Media Sets Desired)	\$7.90 each set — U.S.
Vol. 4 No. 1 Vol. 4 No. 2	Foreign add \$3.00 each set
August, 1984 September, 1984	

Indicate your choice of media: ☐ ON TAPETM ☐ ON DISKTM

Indicate which computer media is for: (check one)

☐ Apple ☐ C-64 ☐ IBM PC ☐ IBM PCjr ☐ TI-99/4A

For more information see inside front cover.

Give A GIFT SUBSCRIPTION To HOME COMPUTERTM magazine

Please Print

PLEASE ENTER A 1-YEAR GIFT-SUBSCRIPTION FOR:

Name _____

Address _____

City _____ State _____ Zip _____

FROM: (Gift-giver must complete this section)

Name _____

Address _____

City _____ State _____ Zip _____

☐ Check if you would like a gift card sent in your name.

☐ Enclosed is \$25 for each 1-year gift-subscription

Canada add \$7; Foreign Surface add \$18 for 1-yr magazine subscription only. Free software not available outside USA.

☐ Check or Money Order Enclosed Total _____

MUST BE IN US FUNDS DRAWN ON A US BANK

Bill my ☐ VISA ☐ MasterCard Date Expires _____

Account No. _____

Tel. No. _____ Signature _____

Enclose payment or credit card information & mail with completed form to:

Emerald Valley Publishing Co.

P.O. Box 5537 • Eugene, OR 97405

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:

1-800-828-2212

In Oregon, Alaska, Hawaii Tel. (503) 485-8796

The Perfect Gift



For Any Occasion!



WHO RECEIVES THE FREE SOFTWARE?

☐ Gift-Giver ☐ Gift-Recipient

Indicate your choice of media: ☐ ON TAPETM ☐ ON DISKTM

Indicate which computer media is for: (check one)

☐ Apple ☐ C-64 ☐ IBM PC ☐ IBM PCjr ☐ TI-99/4A ☐ Rain-check

Select your 2 FREE software issues:

☐ Sept. 1984 ☐ Oct. 1984 ☐ Nov. 1984 ☐ Dec. 1984 ☐ Jan. 1985

Enclose payment or credit card information & mail with completed form to:

Home Computer Magazine

P.O. Box 5537 • Eugene, OR 97405

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:

1-800-828-2212

In Oregon, Alaska, Hawaii Tel. (503) 485-8796

Allow 6-8 weeks for your first issue.

For more information see page 7.

Save \$ \$ \$ On BACK ISSUES of



**SPECIAL CLOSE-OUT PRICES NOW
IN EFFECT FOR TI-99/4A USERS!**

Please Print

Name _____

Address _____

City _____ State _____ Zip _____

☐ Check or Money Order Enclosed

MUST BE IN US FUNDS DRAWN ON A US BANK

Bill my ☐ VISA ☐ MasterCard Date Expires _____

Account No. _____

Tel. No. _____ Signature _____

Enclose payment or credit card information & mail with completed form to:

Emerald Valley Publishing Co.

P.O. Box 5537 • Eugene, OR 97405

Or use our TOLL-FREE Order Line for VISA/MasterCard orders only:

1-800-828-2212

In Oregon, Alaska, Hawaii Tel. (503) 485-8796



*Due to the limited availability of some issues, it may not be possible to send you all your choices. Therefore, please indicate your choice of alternate issues in the space provided below to expedite your order.

ITEMS	PRICE	
99'er Home Computer Magazine Back Issues (Circle Issues Desired)	Any 3 For Only	Any 6 For Only
Vol. 1 No. 6 Nov. '82 Dec. '82	\$5.95	\$10.95 U.S.
Jan. '83 Feb. '83 Mar. '83	\$7.50	\$12.50 Canada
Apr. '83 May '83 June '83	\$8.95	\$13.95 Foreign Surface
July '83 Aug. '83 Sept. '83	\$9.95	\$14.95 Foreign Air
Oct. '83 Nov. '83	All prices include postage.	
Finder Binder (no tapes)	\$9.95	In U.S. Only

Total _____

* Please list alternate choices here:

Mo. Yr. Mo. Yr. Mo. Yr.

For more information see inside back cover.

GUIDE TO **HOME COMPUTER**TM magazine READER SERVICES

See Rear Bind-In Card

See Page 7

See Inside Front Cover

See Rear Bind-In Card

See Rear Bind-In Card

See Inside Front Cover

See Inside Back Cover

See Page 93

HOME COMPUTERTM magazine

Subscriptions

Gift-Subscriptions

Back Issues



ON TAPETM



ON DISKTM



This Issue's Software

Program Subscriptions

Back Issues



Back Issues



Blank-Media Service

LIMITED
SUPPLY

EXCLUSIVELY FOR TI-99/4A USERS

LIMITED
SUPPLY

Special **CLOSE-OUT** Prices on All BACK ISSUES of



Exclusively covering the TI-99/4A home computer. Complete with ready-to-type-in-and-run program listings! The original 99'er Magazine and 99'er Home Computer Magazine were the forerunners of the current Home Computer Magazine.



ISSUE #1 ISSUE #2 ISSUE #3 ISSUE #4

ISSUE #6 (Partial Contents)

• How To Produce Sound Effects • Debugging a Game Program • How to Start a User's Group • Verbose: A speech Vocabulary Expansion Aid • Color Mapping • Dynamic Manipulation of Screen Character Graphics • The Beginner's Guide to Cassette Operation With the Home Computer • Pre-School Block Letters and Data Compaction • Picking the Ponies in TI BASIC • Battle Star Space Game • 3-D Animation on the Home Computer • Programming Tips • Who is LOGO for? • Tower of Hanoi in TI LOGO • A Review of the TLesson-Development Software • An Interview with a Game Designer • Learning Assembly Language with a Magic Crayon • and much, much more.

NOVEMBER 1982 (Partial Contents)

• Chatting with Your Micro: Languages for the Home Computer • A Review of the Smith Corona TP-1 Daisy Wheel Printer • The Micro Jaws Arcade Game • A Knight's Tour in TI BASIC • LOGO Has Style • ASPIC: A Language for Children • A P-System Beginners Tutorial • An Interview with a P-System Pioneer • A Mini-Memory Screen Dump to the Home Computer Printer • Up Scope!—An exciting Undersea Combat Game • Strategy for Munch Man • A Brief Encounter with a TI Hand-Held Computer • 99'er Shopping Bus • A Pocket Battleship • Sub-Programs in Extended BASIC • Arcade & Adventure Game Reviews • and much, much more.

DECEMBER 1982 (Partial Contents)

• Text Scribe: A Text Editor for the Home Computer • A Christmas Computer Carol • Managing a Mailing List the Futura Way • Parsec: The Arcade Game • Plotting With the Home Computer—Pixel by Pixel • Preventing the Situation—On No! Memory Full • A Colorful Tour of TI-Fest: The Home Computer Show • Santa's Workshop: The Making of a Home Computer • The Turtle Arcade: Movies & Video Games in LOGO • Controlling a BASIC Termite • The 99'er Gold Rush—An Arcade/Adventure in the Home • 99'er Digest of News & Happenings in the TI World • Plus Games, Reviews, and much, much more.

JANUARY 1983 (Partial Contents)

• Computer Assisted Instruction for the Handicapped

• p-System Basics • Debugging in LOGO • The Dow-4 Gazelle Flight Simulator • Note Whiz and Pitch Master Musical Game Reviews • Learning With the PLATO Computer Library • Strategies for Adventure Gaming • Death Drones • Using the Line-By-Line Assembler • Close Encounters of the Simon Kind • Electrical Engineering Education Program • Interview With an Arcade Game Designer • TI Invaders • Programming With Pascal • Cyber-Dice • News and Happenings in the Home Computer World • Arcade Game Reviews • The Thief Adventure Game • Programming Tips • and much, much more.

FEBRUARY 1983 (Partial Contents)

• Texas Instruments at the Winter Consumer Electronics Show • Home Computer Printers on Review • How to Create Math Daisies in LOGO • Vectors in LOGO • ASPIC: A Language for Teachers • The Joys of Adventuring—Part 2 • Programming Pointers with Chuck-A-Luck—Part 4 • Interview With the Voice of Parsec • Why You Need a Printer for Your Home Computer • Lifeline to Titan Space Game • Night Blockade Battleship Game • Tower of Hanoi Pocket Program • Computer Gaming Software Reviews • News of Late Developments in the World of Home Computers • and much, much more.

MARCH 1983 (Partial Contents)

• An Introduction to the TI-99/2 Basic Computer • The Hex-bus and the 4/A Connection • Making Your Own Say and Spell Game • Disabling Children: Learn and Grow • Super Cataloger—A Review of a Disk Library Utility Program • TI's New CC-40 Compact Computer • Robots and Their Social Impact • Twenty Questions With Robot Redford • The Gravity of LOGO • Joystick Jockey—An Overview of Remote Controllers • Parsec Strategy • Converting Extended BASIC to Assembly Language • Matrix Muncher • Mini Memory Disassembler Utility • Pulling the Shade on Sprites • Letters on LOGO • Tiny Tutorials • Games programs, reviews, and much, much more.

APRIL 1983 (Partial Contents)

• Computer Assisted Savings Planning to Build Your Nest Egg • TextCipher Writes and Decodes Secret Messages • Crossbytes—Computer Vocabulary Crossword Puzzle • Cutting Corners on Your Food Budget Using Coupons • Introducing Financial Planning with Multiplan • The Design Philosophy of the Compact Computer • LOGO Takes On the Popular Fifteen Puzzle • Super Language—Programming Sprites in Mini Memory • Colorful Word World—Reading Readiness for Pre-schoolers • Gameware Buffet's

A Maze-ing Boa Alley Game • Giant and Dwarfs Entrapment Game • Game Reviews • Programming Tips • Money Saving Hints • and much, much more.

MAY 1983 (Partial Contents)

• A Consumer's Guide to Word Processing • Word Processing Market Basket • A Generalized Filing Program for VLPs • The Multipian Medium Balances Your Checkbook and Budget • Activity Accountant Helps School Secretaries with Extracurricular Activities • Maximizing Your Mini Memory's 4K of RAM • Exploring Enhanced BASIC on the Compact Computer • The LOGO Tortoise Debates the BASIC Hare • A Pocket Program to Organize Data with Linked Lists • Mentally Handicapped Learners Team Up with the TI-99/4A • The Wonders of Diskette Storage • Beeline—a Multi-Screen Strategy Game • Lost Ruins—an Archeological Dig Game • 3-D Illusions with Sprites in Depth • Game reviews, Group Grapevine, and much, much more.

JUNE 1983 (Partial Contents)

• Children and Computers Make the 99/4H Connection • Tune Your Guitar with Our TI Tuning Fork • Talk to Your Computer—Voice Technology Is Here • Gameware Buffet's Eat or Be Eaten Aardvark Game • Protect Your Station in the Space Zapper Game • What Multipian Can and Can't Do • Understanding Inputs and Outputs in Drive For Diskettes—Part 2 • Calculate Loan Schedules on the CC-40 • Go on a LOGO Vacation • Letters on LOGO • A Review of Upper Room Software's Programs for Special Learners • Construct an RS232C Joystick Interface • Group Grapevine • Shopping Bus • A Natural Language Interface for the Professional • Game Reviews • and much, much more.

JULY 1983 (Partial Contents)

• The Evolution of Home Computer Graphics Comes Alive in Graphics Groups Up • Five Data Organizers in Never Out of Sorts • TI & 99'er at the Consumer Electronics Show • WarGames: The Movie and the Book • Editing with Multiplan • The LOGO Logician Presents To Model Is to Learn • LOGO Mosaic Designs Fill the Screen • Your Speech Synthesizer as a Spelling and Foreign Language Teacher • Software for Your Low-cost Printer Port • Gameware Buffet's Treasure Island and the Colorful Switch-A-Row • A Book Review of Learn BASIC for CC-40 Users • 3-D Animation with the TMS 9918A Video Chip • Games Reviews • Group Grapevine • and much, much more.

AUGUST 1983 (Partial Contents)

• The Home Computer Goes To Work • Bit One, Part Two at the Fashion Factory • Better Business Bar

Graphs in Graphic Persuasion • An Ensemble of Assemblers • Cashflow Helps Money Management • Keystrokes for Thrifty Folks—a Review of Typewriter • Game Reviews of Cavern Quest and Starprobe '99 • Counting Fun for Preschoolers • Peripheral Vision 99 • Mean Machines and Small Potatoes • Multipian Medium Groups Cells into Rectangles • Turtle Text, a LOGO Word Processor • Group Grapevine • Hello, Little Brother—CC-40 Speaks to 99/4A • Plato's Progress Looks at Student Assessment • Pocket Sunrise • Public Investigator • Gameware Buffet's Jungle Jim, Success Formula, and much, much more!

SEPTEMBER 1983 (Partial Contents)

• Adding on to Your Home Computer • 99'er Directory of Commercially Available Software, Accessories and Peripherals • Peripheral Vision 99 Hardware Reviews • Pocket Sunrise, Part Two in Extended BASIC • Byte Lightning—Mini Memory Plays a Game • TI-WRITER Tutorial • LOGO's Functions, Sets and Turtles • The CC-40 and 99/4A Take the Data, and RUN • PLATO's Progress Looks at Geometry Courseware and the Shape of Things to Come • Gameware Buffet's Challenge of Camelot and The Fly • Game Reviews of Nit Wit and Crime and Punishment • Group Grapevine • 99'er Hall of Fame • 99'er Digest Update on New Products, and much, much more!

OCTOBER 1983 (Partial Contents)

• Adventures and Fantasy with Your Home Computer • Do-It-Yourself: A P-System Pricer/Processor Emulator, a 4-Bit Microprocessor • Have No Fear: Assembly Language Won't Bite, Part 1 • Make Your Mark • Les Izmore and Debug Computing Cartoon • Once Upon a Tortoise Shell—A LOGO Adventure • Turtle Strut • Number Nibbler for Children • Lots of Plots on Your CC-40 and HX1000 Four-Color Printer/Plotter • Multipian Medium—Bartender • A Grisly Adventure Bear Hunt • Escape From Wizard's Keep in Extended BASIC • Game Reviews • Shopping Bus • and much, much more.

NOVEMBER 1983 (Partial Contents)

• Education with your Home Computer • Five Creative Learning Activities for Children • Let's Build America • Have No Fear: Assembly Language Won't Bite, Part 2 • Squeezing the Most Out of TI BASIC • TI-WRITER Tutorial • LOGO Lexicon • Interview with Dale Osborn • TI-WRITER At Home in the Office • PLATO's Progress • Les Izmore and Debug • The Multipian Medium • Computer Arrested Instruction—99'er Interviews the Kids • Gameware Buffet: Taco Man and Robo Chase • Game Reviews of Jai! Break and Arithmetic • Hall of Fame • 99'er Digest • and much, much more.

CLOSE-OUT SPECIAL

Hurry, Supplies Are Limited!

Any **3** back issues for only **\$5.95** postpaid in the U.S.

Any **6** back issues for only **\$10.95** postpaid in the U.S.



NEVER BEFORE OFFERED!

The Best Programs*

Available On Disk or On Tape
From Each Issue Above.

only **\$3.95** per issue, postpaid in U.S.

—NOT FOUND ON ORDER FORM—Write on order form or on separate sheet issues desired and choice of tape or disk.

*Requires corresponding magazine
back issues for program documentation.

* SUPER BONUS *

This offer not found on center bind-in order card.

Please write "Super Bonus" on card when ordering and indicate your 12 choices.

Any **12** for only **\$21.90 PLUS a**
FREE Simon's Saucer Flipper/Snapper™



• A quality, ready-to-run game on cassette tape or diskette.

• A complete, easy-to-use programming lesson on a deck of colorful flip cards.

• A durable and attractive collector's case for your software library.



All prices include postage in the U.S. To Order, Use Order Card In Center Of Magazine.

ALL PROGRAMS IN THIS MAGAZINE



ONLY \$3.95 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 ON DISK™ or ON TAPE™ to newsstand purchasers or subscribers of this magazine.

For only \$3.95 (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.