Sprite Graphics And Sound Synthesis On The Commodore 64

# COMPUTE!

$2.50
September 1982
Issue 28
Vol. 4, No. 9
63379 £1.85 in UK

The Journal For Progressive Computing™

## The Sounds Of TRON

**Using Apple and Atari In The Production Of A Computer Fantasy**


**A Turtle PILOT Compiler In BASIC For Personal Computers**
Part 1: The Apple Version

**Automatic Disk Boot For Commodore Computers**

**Joystick Tutorial For The VIC-20**

**An Atari Rainbow**
Three GTIA Graphics Articles

**Graphic Display For The Radio Shack Color Computer**
Extending The Utility Of The Energy Monitor Program

0 74470 63379 09

**REVIEWS: Space Games For VIC And Atari, PET/CBM Data Base Manager And More.**

# THE LEADING EDGE IN PRINTERS

## ONE GREAT LINE. ONE GREAT WARRANTY.

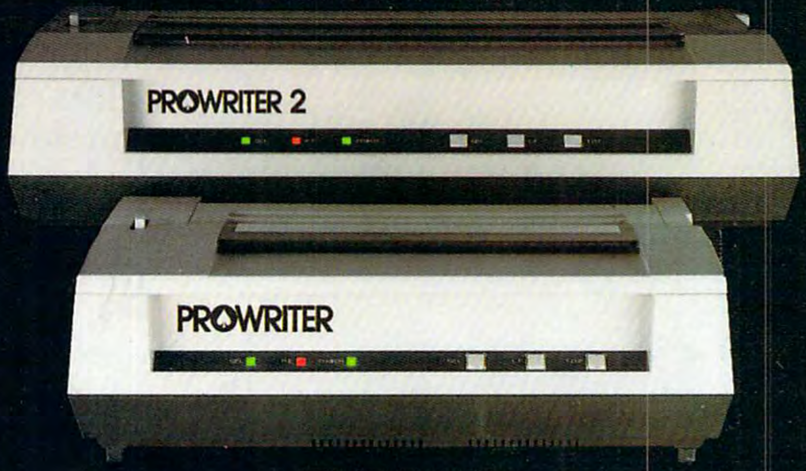
Finally, there's one full family of printers that covers every business or word processing application—all from C. Itoh, a company known for packing more product into less price; and all distributed exclusively by Leading Edge, a company known for searching out and providing that very thing. Which means that one call to one source can get you any printer, any time you need it, for any purpose. All backed by a full years' warranty from Leading Edge. (Try *that* on any other line of printers.)

## THE PRO'S.

**The Prowriters: business printers—and more.** The "more" is a dot-matrix process with more dots. It gives you denser, correspondence quality copy (as opposed to business quality copy, which looks like a bad job of spray-painting).

**Prowriter:** 120 cps. 80 columns dot matrix compressable to 136. 10" carriage. Parallel or serial interface.

**Prowriter 2:** Same as Prowriter, except 15" carriage allows full 136 columns in normal print mode. Parallel or serial interface.




## THE STAR.

**The Starwriter F-10.** In short (or more precisely, in a sleek 6" high, 30-pound unit), it gives you more of just about everything—except bulk and noise—than any other printer in its price range. It's a 40 cps letter-quality daisy-wheel with a bunch of built-in functions to simplify and speed up word processing. It plugs into almost any micro on the market, serial or parallel.


## THE MASTER.

**The Printmaster F-10.** Does all the same good stuff as the Starwriter except, at 55 cps, the Master does it faster.


*Distributed Exclusively by Leading Edge Products, Inc., 225 Turnpike Street, Canton, Massachusetts 02021.*
*Call: toll-free 1-800-343-6833; or in Massachusetts call collect (617) 828-8150. Telex 951-624.*

# IF YOU'RE WAITING FOR THE PRICE OF WORD PROCESSORS TO FALL WITHIN REASON,


Everyone expected it would happen sooner or later... with WordPro PLUS™ it already has! Now all the marvelous benefits of expensive and advanced word processing systems are available on Commodore computers, America's largest selling computer line. WordPro PLUS, when combined with the new 80 column CBM 8032, creates a word processing system comparable to virtually any other top quality word processor available—but at savings of thousands of dollars!

New, low cost computer technology is now available at a fraction of what you would expect to pay. This technology allowed Commodore to introduce the new and revolutionary CBM 8032 Computer.

WordPro PLUS turns this new CBM 8032 Computer into a sophisticated, time saving word processing tool. With WordPro PLUS, documents are displayed on the computer's screen. Editing and last minute revisions are simple and easy. No more lengthy re-typing sessions. Letters and documents are easily re-called from memory storage for editing or printing with final drafts printed perfectly at over five hundred words per minute!

Our nationwide team of professional dealers will show you how your office will benefit by using WordPro PLUS. At a price far less than you realize.

**Invest in your office's future...**
**Invest in WordPro PLUS...**
**Call us today for the name of the WordPro PLUS dealer nearest you.**

**Professional Software Inc.**
51 Fremont Street
Needham, MA 02194
(617) 444-5224
TELEX: 95 1579

TM WordPro is a Registered Trademark of Professional Software, Inc. WordPro was written by Steve Punter.
All specifications subject to change without notice.

# We could tell you we make
# We don't have to.


**All for the Apple and Atari on diskette.**

*You keep Brøderbund at the top of the best seller lists, so you must know what it takes to make a great game. We think our APPLE and ATARI games are uniquely challenging with action and graphics second to none. But you keep asking for Brøderbund games because they're just plain fun.*

*Enjoy the magic of our growing collection. Lead a daring rescue mission in CHOPLIFTER. Or survive the maze and monsters in slithery SERPENTINE and LABYRINTH. Blast through the future with STAR BLAZER or conquer the ultimate pinball game with DAVID'S MIDNIGHT MAGIC. Take up train robbing in TRACK ATTACK or tense your mind and reflexes in DUELING DIGITS.*

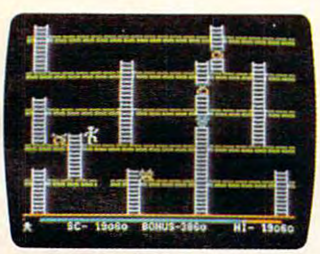
**Brøderbund**

*Brøderbund products are available at your retailer or by writing to:*
*Brøderbund Software, Inc., 1938 Fourth Street, San Rafael, CA 94901 • Tel: (415) 456-6424*

# the finest computer games. You keep telling us.

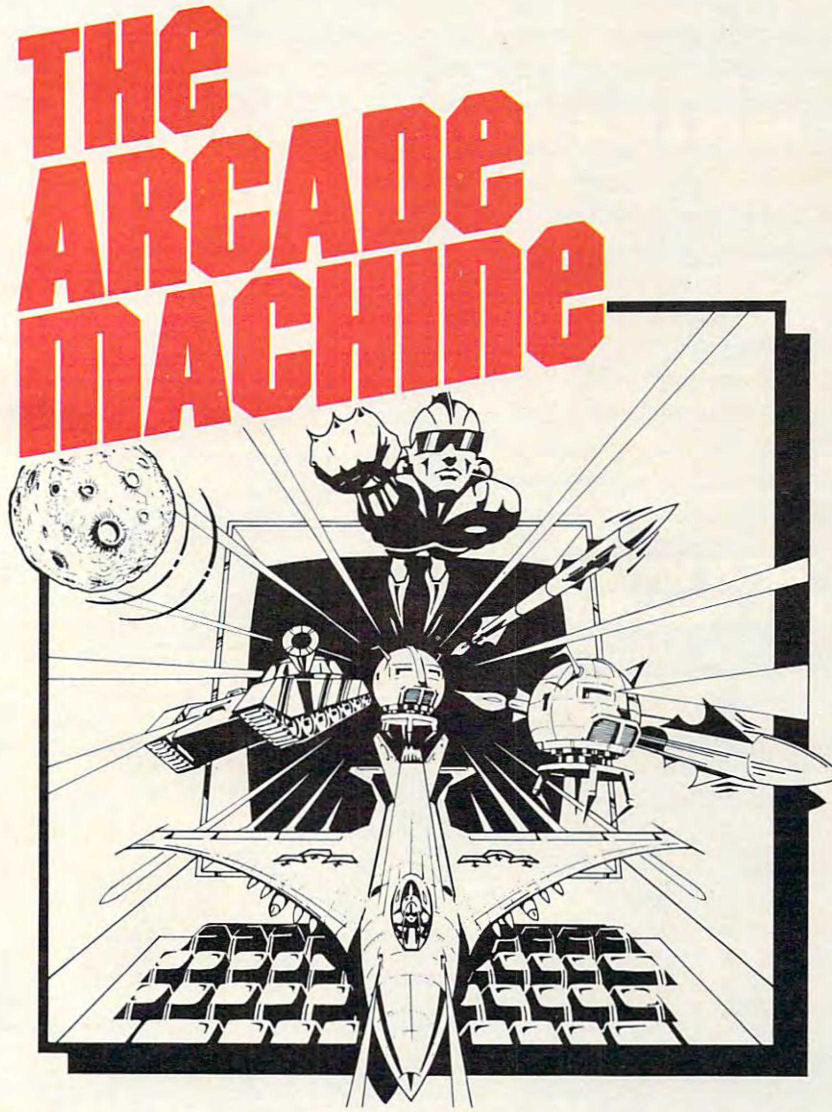

**SEAFOX**
*Sub against the convoy*
*Apple & Atari disk*

**DEADLY SECRETS**
*Hi-res adventure*
*Apple disk*




**APPLE PANIC**
*The arcade classic*
*Apple disk, Atari cassette & disk*


**STELLAR SHUTTLE**
*Monsters and meteors*
*Atari cassette & disk*




*PUT US OUT OF BUSINESS! THE ARCADE MACHINE lets you design and produce your own computer games, without any programming knowledge! Send us your best game and enter the BRØDERBUND ARCADE MACHINE CONTEST. We'll be giving away thousands of dollars worth of hardware and software in prizes. If you have a creative touch and an artistic eye, you too can be an arcade designer. Write to Brøderbund for contest details or visit your participating retail store! (Available on Apple disk.)*

**Software**

*All Brøderbund games are fully guaranteed. If they ever fail to boot, return the original disk to Brøderbund for a free replacement. If you have physically damaged the disk please include five dollars for replacement.*

# WHY THE MICROSOFT RAMCARD™ MAKES OUR SOFTCARD™ AN EVEN BETTER IDEA.

Memory — you never seem to have quite enough of it.

But if you're one of the thousands of Apple owners using the SoftCard, there's an economical new way to expand your memory dramatically.

### 16K ON A PLUG-IN CARD.

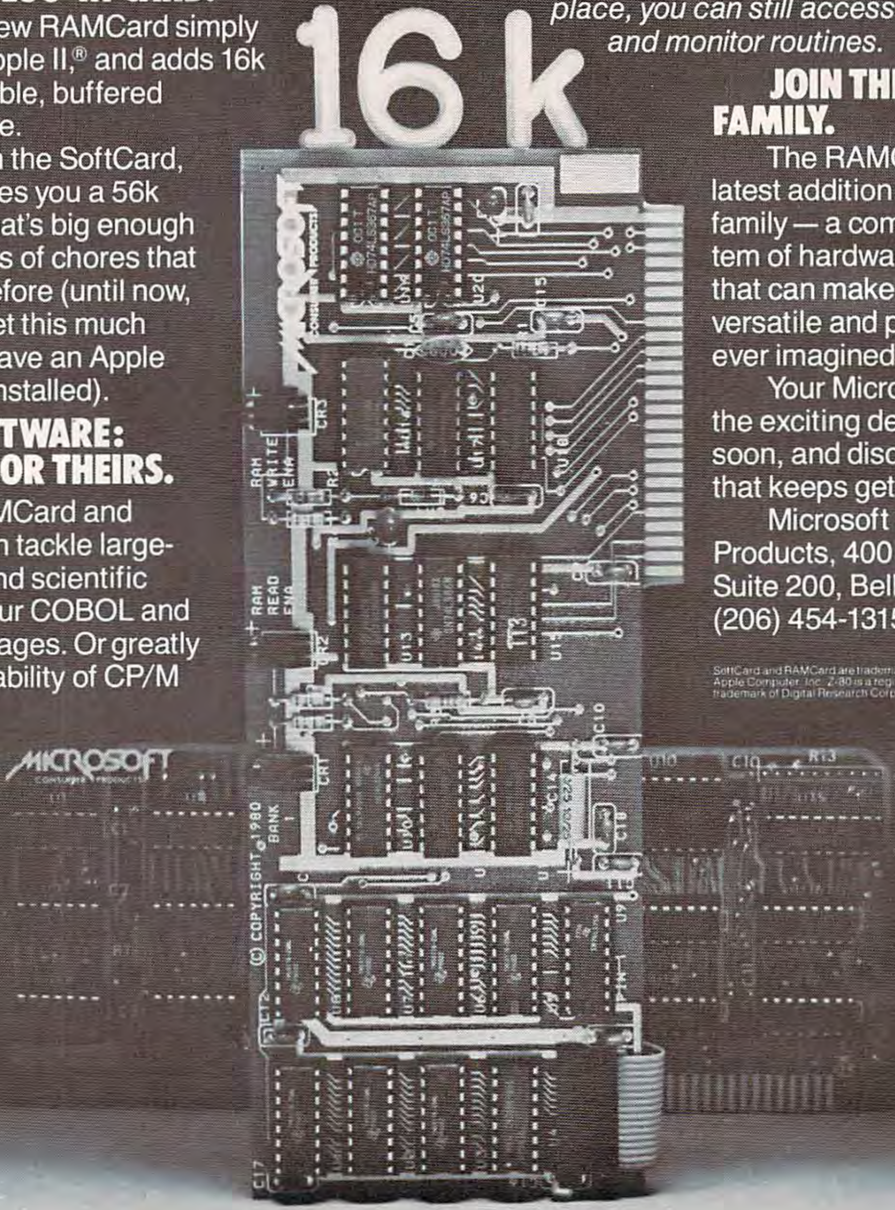
Microsoft's new RAMCard simply plugs into your Apple II,® and adds 16k bytes of dependable, buffered read/write storage.

Together with the SoftCard, the RAMCard gives you a 56k CP/M® system that's big enough to take on all kinds of chores that would never fit before (until now, the only way to get this much memory was to have an Apple Language Card installed).

### GREAT SOFTWARE: YOURS, OURS, OR THEIRS.

With the RAMCard and SoftCard, you can tackle large-scale business and scientific computing with our COBOL and FORTRAN languages. Or greatly increase the capability of CP/M applications like the Peachtree Software accounting systems. VisiCalc™ and other Apple software packages can take advantage of RAMCard too.

And RAMCard gives you the extra capacity to develop advanced programs of your own, using the SoftCard and CP/M. *Even with the RAMCard in place, you can still access your ROM BASIC and monitor routines.*




### JOIN THE SOFTCARD FAMILY.

The RAMCard is just the latest addition to the SoftCard family — a comprehensive system of hardware and software that can make your Apple more versatile and powerful than you ever imagined.

Your Microsoft dealer has all the exciting details. Visit him soon, and discover a great idea that keeps getting better.

Microsoft Consumer Products, 400 108th Ave. N.E., Suite 200, Bellevue, WA 98004. (206) 454-1315.



MICROSOFT

## Table of Contents

**September 1982 Vol. 4, No. 9**



**AP** = Apple, **AT** = Atari, **P** = PET/CBM, **V** = VIC-20, **O** = OSI, **C** = Radio Shack Color Computer, * = All or several of the above.

**COMPUTE! The Journal for Progressive Computing** (USPS: 537250) is published 12 times each year by Small System Services, Inc., P.O. Box 5406, Greensboro, NC 27403 USA. Phone: (919)275-9809. Editorial Offices are located at 625 Fulton Street, Greensboro, NC 27403. Domestic Subscriptions: 12 issues, $20.00. Send subscription orders or change of address (P.O. form 3579) to Circulation Dept., **COMPUTE!** Magazine, P.O. Box 5406, Greensboro, NC 27403. Second class postage paid at Greensboro, NC 27403 and additional mailing offices.

ISSN 0194-357X.

# The Editor's notes...

Robert Lock, Publisher/Editor-In-Chief

## A Very Personal Aside: 100,000 Here We Come!

Those of you who remember when we broke through 10,000 paid circulation will be excited to know that we are rapidly approaching the 100,000 mark. Our October press run will exceed 100,000 for the first time in our history, and we forecast breaking 100,000 paid in November.

Practically speaking, that 5,000 magazine growth from 95,000 to 100,000 is no different than that same amount of growth from 90,000 to 95,000. Emotionally it's glorious, and we're rather beside ourselves with excitement. Thank you all for your continued input and support. If you should happen to call in early September and hear a bit of background uproar, you'll understand why.

## The Commodore 64

As you read this, the first production line Commodore 64's should be appearing in your local stores. We know they exist. Our Features Editor, Tom Halfhill, spent an entire day working with one at Commodore's Valley Forge, PA headquarters to produce the article in this issue. Rumor has it that outside software vendors have been working hard on an impressive array of support software for this new arrival from Commodore.

## The Mass Market Micros

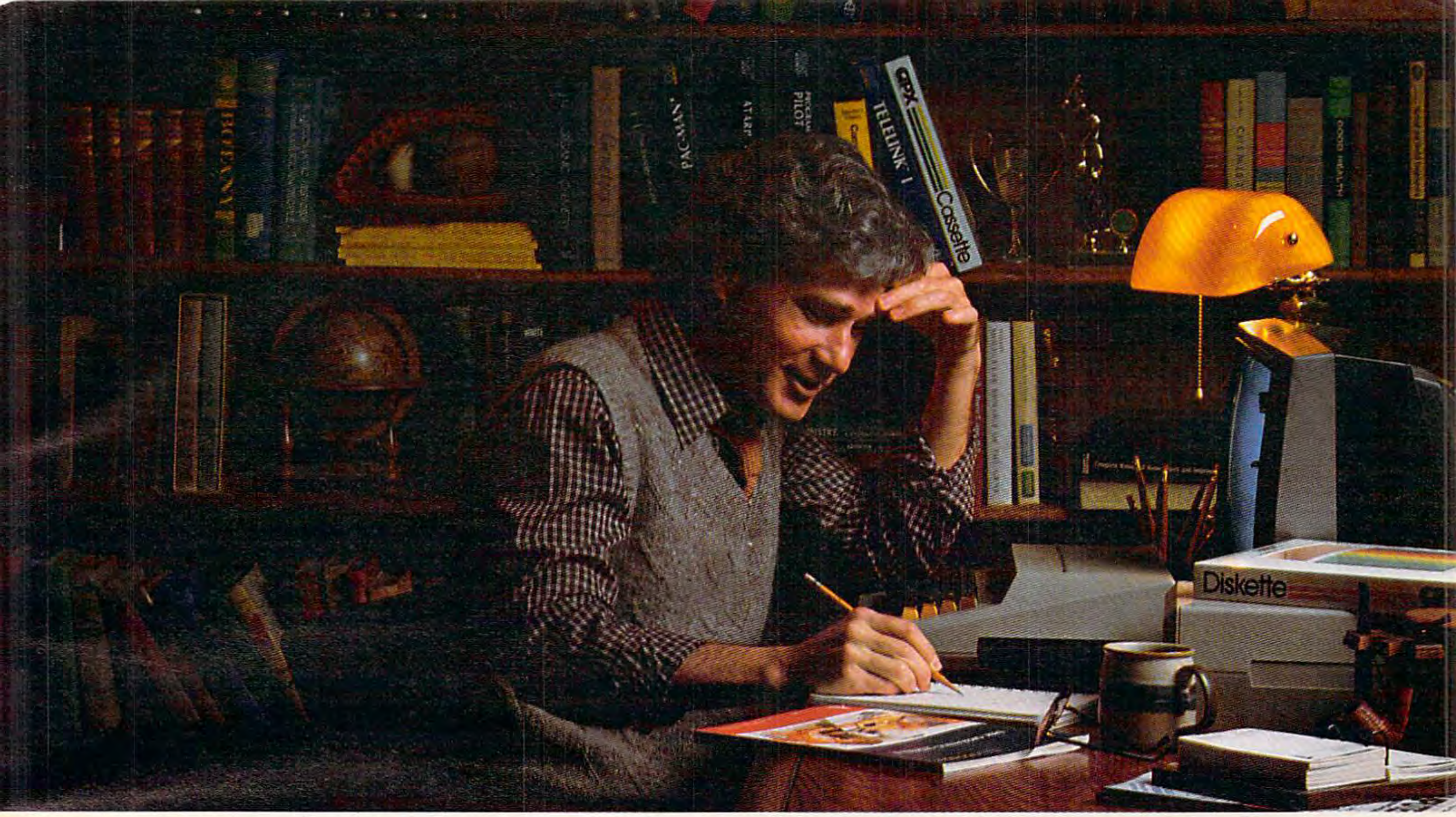
The competition continues: Atari has dropped the base price of the Atari 400 to $299.95 and entered a major marketing agreement with Sears. K mart has announced the doubling of the number of stores carrying the Commodore VIC-20. You'll find VIC's in 1100 K mart locations now. K mart is selling the Texas Instruments personal computer as well. Commodore also is selling VIC-20's through Montgomery Ward stores. Radio Shack/Tandy has decided to drop its traditional approach to the consumer market, and expand the potential customer base for their Color Computer. They will be selling a version of it (different color case) through department stores, etc.

Our sources indicate that between the two of them, Atari and Commodore are currently shipping over 60,000 VIC-20's and Atari 400's a month. By the time you factor in the Atari 800, the Commodore 64, Texas Instruments, Sinclair, and the Radio Shack Color Computer, you can conservatively estimate 200,000 consumer computers a month by October. Imagine... there were some who thought we'd go the way of the CB radio.

## The October Issue: Games

Michael Day combines telecommunications with games for an exciting column on telegaming; Bill Wilkinson articulates machine language gaming on the Atari; Tom Halfhill interviews industry leaders on the future of games. You'll find all of this and much more in our very special October theme issue, including some exciting games ready to type into your computer.

Our July issue featured an action game and accompanying article entitled "Gold Rush!" This game should not be confused with an arcade-graphics game for both the Apple and Atari of the same name produced and marketed by Sentient Software of Aspen, Colorado. No comparison or confusion was intended regarding the products of Sentient Software, and readers should be aware that these two games are entirely different products. In any future use of this article and action game, we will refer to it as "Gold Miner." ©




# MORE THAN EVER, ATARI HOME COMPUTERS ARE SPEAKING YOUR LANGUAGE.

With more program languages than ever to choose from, you now have more opportunities than ever to utilize the amazing capabilities of the ATARI 800™ Home Computer.

Whether you're a beginning programmer, or at the forefront of the art, you'll find an ATARI programming language that can make your task quicker and simpler than ever before. Just look at what ATARI has to offer:

**ATARI Microsoft BASIC**—Now we offer the industry standard, the most powerful Microsoft BASIC yet. With simple commands, it allows you to take advantage of unique ATARI hardware features such as our well-known player/missile graphics. For ease of programming, it includes n-dimensional arrays for numerics and string handling. And importantly, conversion procedures are simple.

**ATARI Macro Assembler**—Faster and more powerful than any ATARI language before, the ATARI Macro Assembler also allows you to access more memory space. And it's excellent for I/O interface and manipulation of such features as: player/missile graphics, sound registers and peripherals. In addition, the macro processor and "include" file library features speed-up program development considerably.

**Fig-FORTH***—For specialized programming needs, such as educational or game applications, ATARI Fig-FORTH is uniquely effective. Fig-FORTH combines power and simplicity in an efficient 10K size, with characteristics of an interpreter and the speed of machine language code.

**ATARI BASIC**—An affordable and easy to use BASIC that requires only 8K of memory. It allows you to take advantage of the spectacular ATARI graphics and sound capabilities. And its immediate mode error messages greatly simplify debugging.

**ATARI Assembler Editor**—An excellent tool to assist the assembler-programmer in creating, editing and debugging assembly programs.

**PILOT**—ATARI PILOT is an exceptional learning language, with built-in "turtle" graphics to let you create spectacular designs and pictures with very short programs. Simple one or two-letter commands allow you to create a dialogue with the computer. And a single "match" command can perform complex text evaluation and pattern-matching instantly.

**ATARI Pascal***—An excellent high-level language for teaching structured programming, and for developing and maintaining programs. In addition to offering all the features of the ISO Pascal standard, ATARI Pascal offers unique extensions that allow you to take advantage of ATARI graphics and sound capabilities.

ATARI is constantly developing new ways to help you get more out of your ATARI 800 Home Computer. So watch for more innovative and exciting programming languages from ATARI in the future.

For more information, write to ATARI, Inc., Dept. C4Z, P.O. Box 16525, Denver, CO 80216.

ATARI
A Warner Communications Company


*Available from the ATARI Program Exchange

**ATARI HOME COMPUTERS**
We've Brought The Computer Age Home.™

| | |
|---|---|
| Publisher/Editor-In-Chief | Robert C. Lock |
| Publisher's Assistant | Alice S. Wolfe |
| Senior Editor | Richard Mansfield |
| Managing Editor | Kathleen E. Martinek |
| Features Editor | Tom R. Halfhill |
| Editorial Assistant | Charles Brannon |
| Administrative Assistant | Vicki Jennings |
| Copy Assistants | Juanita Lewis<br>Mary Parker |
| Associate Editors | Jim Butterfield, Toronto, Canada<br>Harvey Herman, Greensboro, NC<br>Fred D'Ignazio, c/o COMPUTE!, P.O. Box 5406 Greensboro, NC 27403<br>David Thornburg P.O. Box 1317, Los Altos, CA 94022 |
| Contributing Editors | Marvin DeJong<br>Bill Wilkinson<br>Gene Zumchak |
| Art Director/ Production Manager | Georgia Papadopoulos |
| Artists | De Potter<br>Kate Taylor |
| Typesetting | Terry Cash |
| Illustrator | Harry Blair |
| Production Assistant | Dai Rees |
| Advertising Sales | Harry Blair |
| Advertising Coordinator | Alice S. Wolfe |
| Operations/Customer Service Manager | Carol Lock |
| Coordinator | Fran Lyons |
| Assistants | Christine Gordon<br>Dorothy Bogan |
| Shipping & Receiving | Jim Coward |
| Accounting Manager | W. Jerry Day |
| Bookkeeper | Ellen Day |
| Accounting Assistant | Linda Roquemore |
| Advertising Accounts Assistant | Bonnie Valentino<br>Ruth Granger |


**Subscription Information**

**COMPUTE! Circulation Dept.**
**P.O. Box 5406**
**Greensboro, NC 27403**

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

**COMPUTE! Subscription Rates**
**(12 Issue Year):**

| | | |
|---|---|---|
| US | (one yr.) | $20 |
| | (two yrs.) | $36 |
| | (three yrs.) | $54 |
| Canada and Foreign Surface Mail | | $25 |
| Air | | |
| Europe, US Possessions | | $38 |
| Middle East, Central America and North Africa | | $48 |
| South America, South Africa, Far East and Australia | | $88 |


Small System Services, Inc. publishes:

**COMPUTE!**
The Journal For Progressive Computing

**COMPUTE! Books**

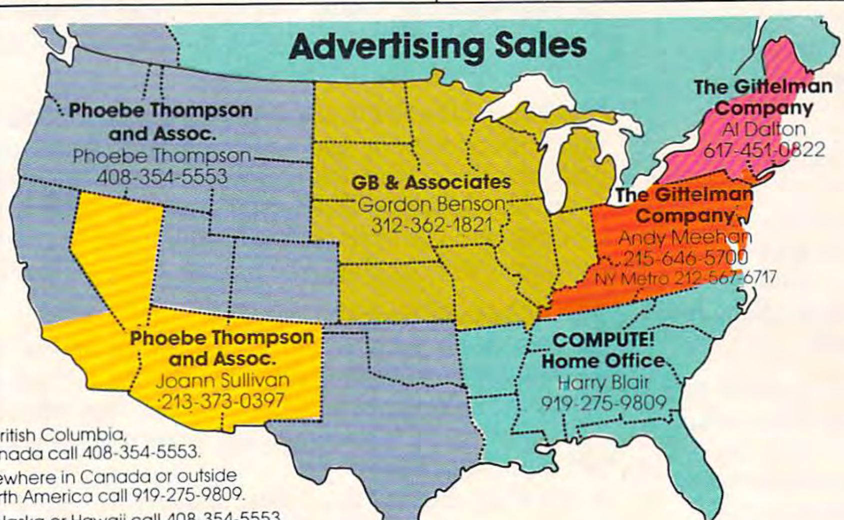
**Corporate office:**
625 Fulton Street,
Greensboro, NC 27403 USA

**Mailing address: COMPUTE!**
Post Office Box 5406
Greensboro, NC 27403 USA

**Telephone: 919-275-9809**

Robert C. Lock, President
W. Jerry Day, Vice-President and Comptroller
Kathleen E. Martinek, Assistant To The President
Sonja Whitesell, Executive Assistant




In British Columbia, Canada call 408-354-5553.
Elsewhere in Canada or outside North America call 919-275-9809.
In Alaska or Hawaii call 408-354-5553.

**Phoebe Thompson and Associates**
101 Church Street, Suite 13
Los Gatos, CA 95030
PHOEBE THOMPSON

**Phoebe Thompson and Associates**
2556 Via Tejon
Palos Verdes Estates, CA 90274
JOANN SULLIVAN

**GB & Associates**
P.O. Box 335
Libertyville, IL 60048
GORDON BENSON

**COMPUTE! Home Office**
625 Fulton Street
Greensboro, NC 27403
HARRY BLAIR

**The Gittelman Company**
Statler Office Building
Suite 518
20 Providence Street
Boston, MA 02116
AL DALTON

**The Gittelman Company**
Summit Office Center
7266 Summit Avenue
Fort Washington, PA 19034
ANDY MEEHAN

**Address all advertising materials to:**
Alice S. Wolfe
Advertising Production Coordinator
**COMPUTE!** Magazine
625 Fulton Street
Greensboro, NC 27403 USA

**Coming In The Special October Games Issue Of COMPUTE!**

**National Public Radio: Computer Programs Via FM**

**Future Games**

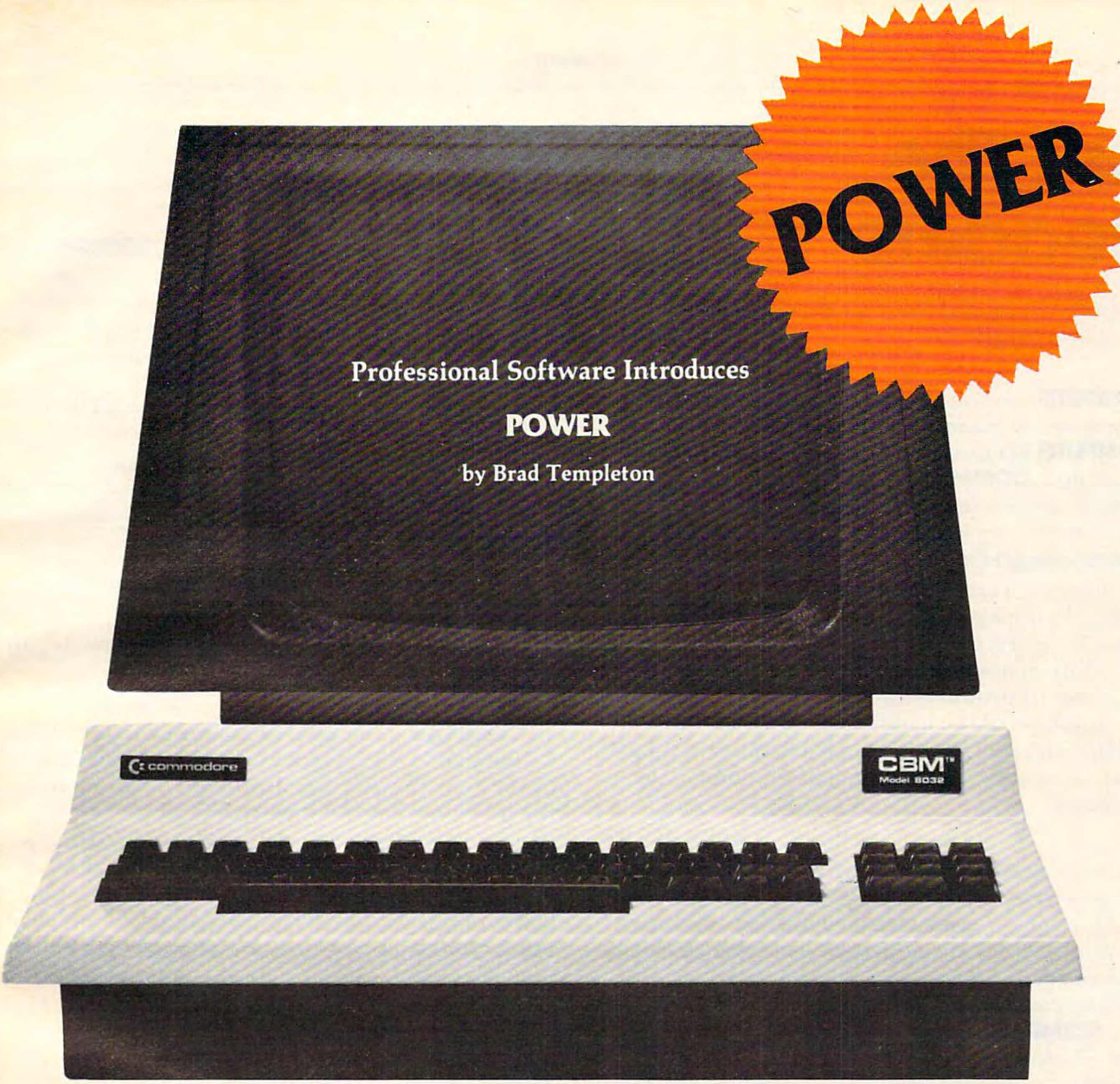
**Superchase**

**Space Math**

**Meteor Storm**

**Laser Barrage**

**and much more...**




# ADD POWER™ TO YOUR COMMODORE COMPUTER

**$89.95**

POWER produces a dramatic improvement in the ease of editing BASIC on Commodore's computers. POWER is a programmer's utility package (in a 4K ROM) that contains a series of new commands and utilities which are added to the Screen Editor and the BASIC Interpreter. Designed for the CBM BASIC user, POWER contains special editing, programming, and software debugging tools not found in any other microcomputer BASIC. POWER is easy to use and is sold complete with a full operator's manual written by Jim Butterfield.

POWER's special keyboard 'instant action' features and additional commands make up for, and go beyond the limitations of CBM BASIC. The added features include auto line numbering, tracing, single stepping through programs, line renumbering, and definition of keys as BASIC keywords. POWER even includes new "stick-on" keycap labels. The cursor movement keys are enhanced by the addition of auto-repeat and text searching functions are added to help ease program modification. Cursor UP and cursor DOWN produce **previous** and next lines of source code. COMPLETE BASIC program listings in memory can be displayed on the screen and scrolled in either direction. POWER is a must for every serious CBM user.

**Call us today, for the name of the Professional Software dealer nearest you.**

**Professional Software Inc.**
51 Fremont Street
Needham, MA 02194
Tel: (617) 444-5224 Telex #951579

TM POWER is a Registered Trademark of Professional Software, Inc.
All specifications subject to change without notice.

# Ask The Readers

Robert Lock, Richard Mansfield, And Readers

**COMPUTE!** *welcomes questions, comments, or solutions to issues raised in this column. Write to: Ask The Readers,* **COMPUTE!** Magazine, P.O. Box 5406, Greensboro, NC 27403. **COMPUTE!** *reserves the right to edit or abridge published letters.*

### Handicapped Computerists

Do there exist any large-screen CRT's for those with sight impairment? Or magnifying devices to enlarge a given CRT?

More generally, has anyone constructed a data base of products and ideas that concern the problems of the handicapped? I'd like to be able to reference and sort such a database with a home computer. Has anything been done in this direction?

Kevin Sinclair

*Since most home computers attach to ordinary televisions, a large-screen TV might be one solution. Also, check with local television dealerships to see if they have a relatively inexpensive, free-standing magnifier now being marketed. Each month new clubs, newsletters, telecommunications "bulletin boards," and special interest data bases emerge. Also,* **COMPUTE!** *publishes a bi-monthly column, "Micros With The Handicapped," and you might want to write to its author, columnist Susan Semancik, c/o* **COMPUTE!**.

### Lowercase On Comprint

I have a PET with a Comprint 912 printer. The printer is capable of printing lowercase, but I have not been able to find anyone who knows how to write a program which would send lowercase to it from the PET.

M. Souza

*If your printer responds to standard ASCII code, you should be able to communicate with it by using the suggestions in the article "PET ASCII To ASCII" (***COMPUTE!***, April 1982, p. 126). This is a short machine language program which makes the necessary conversion between PET's way of coding and ASCII. Here's a sample BASIC program which would use this conversion routine. Don't forget the comma in line 30.*

```
10 OPEN4,4
20 INPUTA$
30 SYS634,A$
40 PRINT#4,A$
50 CLOSE4
```

### PET To Epson

In the "Ask The Readers" column (**COMPUTE!**, June 1982), Hank Roth asked how to make his Epson printer print single line feeds instead of double line feeds. You can achieve this by ending each PRINT# line in your program with: CHR$(13);

(Don't forget the ending semicolon.) This can be added in two ways:

1. **100 PRINT#1,"(text here)"CHR$(13);**

or

2. **100 PRINT#1,"(text here)";**
   **110 PRINT#1,CHR$(13);**

The first method requires the least added memory, but adds line length.

Norman Girard

### Computers In Medicine

In **COMPUTE!**, May 1982, #24, one of your readers, L. Thomas, wrote in to ask where to obtain a copy of the book *Computers In Medicine*. Unfortunately the title is not exact. I have written three books *Computers In Medicine* – an Introduction ($16), *Computers In Laboratory Medicine* ($28.00), and *Microcomputer Programs In Medicine* ($55.00). The books may be obtained from Medical Communications, Suite 10E, 860 Fifth Ave., New York. I hope that this information is helpful.

Derek Enlander, M.D.

### A New VIC Champion

In **COMPUTE!**, June 1982, #25, "Ask The Readers," Mary Payne said she had the record for best score on Vixel's "Fire" game of seven seconds. Well, I put the fire out in six seconds, with 750 gallons remaining. So who's got the record now? I also think I have the high score on Vixel's "Race" game of 6421.

Mike DeLuca

### Apple Baseball Scorekeeper

My problem is this: I have seven teams in a baseball game and I keep the win-loss record of each team in a text file on my Apple II Plus. I wrote a program that updates the files for me, but could not find a way to make the computer print the seven teams out in order by winningest to losingest record. What I am asking for is a routine that will sort the win-loss records of the teams, and print them out in the proper order. An example: Milwaukee currently has a 6 win-4 loss record, Boston is 6-4 also, New York is 5-5, and Pittsburgh is 3-7. I need

# Your VIC 20 never had it so good!


VIC 20 owners rejoice! HES presents a complete range of software from our exciting series of games to our professional group of utility and language programs.

Our new cartridge programs include: HES MON, an indispensible monitor for assembly language programmers; HES WRITER, a word processing program; Turtle Graphics, a fun and easy way to learn computer programming; and VIC FORTH, a powerful language that is many times faster than BASIC, yet easier to use than assembly language.

HES is committed to offering high-quality, well-documented computer programs on a continual basis. Look for our cartridge and cassette based software at your local dealer.

HES

Human Engineered Software
71 Park Lane
Brisbane, California 94005
Telephone 415-468-4110

VIC 20 is a registered TM of Commodore

a routine that will sort these four, usually seven, teams and print them in the order: Milwaukee, Boston, New York, Pittsburgh.

Greg Palen

*This program will read the seven teams (you only mentioned four, so three others were arbitrarily chosen) into the arrays N$,WINS, and LOSSES which respectively hold the team names, wins, and losses. The subroutine at 370 prints the list of the teams in table form, and includes the percentage, which is calculated with:*

**WINS/(WINS + LOSSES)**

*Note line 420, which rounds the variable A to three decimal points.*

*The sort, a simple bubble sort, uses the percentage to order the list. Teams with a higher percentage will appear before teams with a lower percentage. Since three arrays are moved around in the sort, it can be slow with large lists. You can change the number of teams by adding or deleting DATA items, and changing the variable NUM, the number of teams, at line 100.*

```
100 NUM=7:REM NUMBER OF TEAMS IN DATA STATEMEN
    TS
110 DIM N$(NUM):REM NAME OF TEAMS
120 DIM WINS(NUM),LOSSES(NUM)
130 REM DATA STATEMENTS ARE: TEAM NAME, WINS, ~
    LOSSES
140 DATA MILWAUKEE,31,29,NEW YORK,29,29,BOSTON
    ,37,23,PITTSBURGH,27,30
150 DATA CLEVELAND,29,29,DETROIT,35,22,BALTIMO
    RE,31,28
160 FOR I=1 TO NUM
170  READ N$(I),WINS(I),LOSSES(I)
180 NEXT I
190 GOSUB 370:REM PRINTOUT
200 PRINT:PRINT "BEFORE SORT"
205 GETA$:IFA$=""THEN205
210 REM BUBBLE SORT ON PERCENTAGE
220 EXCHANGED=0
230 FOR I=1 TO NUM-1
240  A=WINS(I)/(WINS(I)+LOSSES(I))
250  B=WINS(I+1)/(WINS(I+1)+LOSSES(I+1))
260  IF A>=B THEN 310
270  TEMP$=N$(I):N$(I)=N$(I+1):N$(I+1)=TEMP$
280  TW=WINS(I):WINS(I)=WINS(I+1):WINS(I+1)=TW

290  TL=LOSSES(I):LOSSES(I)=LOSSES(I+1):LOSSES
    (I+1)=TL
300  EXCHANGED=1:REM NOTE EXCHANGE
310 NEXT I
320 IF EXCHANGED=1 THEN 220
330 REM NO EXCHANGES FOR ENTIRE LIST, SO SORT ~
    COMPLETE
340 GOSUB 370
350 PRINT:PRINT"SORTED"
360 END
370 PRINT CHR$(147);:REM USE HOME ON APPLE II+

380 PRINT "TEAM";TAB(15);"WINS";TAB(20);"LOSSE
    S";TAB(30);"PERCENTAGE":PRINT
390 FOR I=1 TO NUM
400  PRINT N$(I);TAB(15);WINS(I);TAB(20);LOSSE
    S(I);TAB(30);
410  A=WINS(I)/(WINS(I)+LOSSES(I))
420  A=INT(A*1000+.5)/1000:PRINT A:REM ROUND T
    O 3 DIGITS
430  PRINT
440 NEXT I
450 RETURN
```

## Atari Microsoft Cartridge

Is there a possibility that Microsoft BASIC will be released as a ROM cartridge? It would be nice not to have to buy a disk drive and not to have to boot up for each use.

Eric Gallion

*We know of no work being done on such a cartridge at this time. Atari says they know of no plans for one.*

## VIC User Groups

In the June 1982, #25 issue of **COMPUTE!**, a question from Fred S. Dart was printed in the "Ask The Readers" column. Mr. Dart asked whether any VIC users groups existed and how he could get in touch with them.

I have recently found out that there are now several in existence. I know of four which are primarily VIC users groups and one which is a PET users group, but now deals with the VIC as well.

Here is a list of these five users groups and their addresses:

Paradox Group
39-41 North Road
London N7 9DP
England

TBH VIC-NIC's
P.O. Box 981
Salem, NH 03079

VIC-20 Computer Club
c/o Ray Thigpen
4071 Edgewater Dr.
Orlando, FL 32804

The VIC-20 User's Group
c/o Roberto Morales, Jr.
655 Hernandez St.
Miramar, Puerto Rico 00907

Toronto Pet Users Group
c/o Chris Bennett
381 Lawrence Ave. W.
Toronto, Ontario
Canada M5M 1B9

Many other Commodore users groups seem to be taking on the VIC also.

Michael Kleinert ©

A Monthly Column

# Computers And Society

David D. Thornburg
Associate Editor

### Chicago, Chicago ...

Each June I make my pilgrimage to the ancestral homeland for the purpose of attending the summer Consumer Electronics Show in Chicago. While I am in town, I also look up a few friends to see how the midwest is faring. To tell the truth, it is easy to fall into the trap of thinking that the San Francisco Bay Area is the intellectual capital of the universe. But a day spent walking the floors of the Chicago Art Institute, or talking with people like Sam Savage, quickly dispels that myth.

Sam is a mathematician/computer scientist/folk singer/inventor/entrepreneur whose recent fame has sprung from his company Shmuzzles, Inc. Shmuzzles, for those who haven't seen them, are jigsaw puzzles in which each identically shaped piece is in the shape of a salamander. (The connection between his design and the interlocking salamanders of artist M.C. Escher is covered in Dr. Savage's talk "Gerbils, Escher, Schlock," but I digress.)

Patterns that interlock perfectly to cover a plane (such as bathroom tiles) are called tesselating figures. The fact that Shmuzzles are both sophisticated mathematically and successful commercially is delightful.

While the projects Sam and I have cooking will be covered at a later time, I did want to share a short piece he has written on the social impact of technology:

***Hey Dad I'm Getting A Disk Read Error***

*I became acutely aware of the electronic revolution over a year ago when I was having dinner at a friend's house. We were sipping wine on the patio when his ten year old son called from inside, "Hey Dad, I'm getting a disk read error" – the kind of computer lingo I was used to hearing from thirty year old technicians.*

*Where will it lead? Does history provide any parallels? With the development of the textile mills of the industrial revolution I am sure many people lamented the loss of the spinning wheel. Some, no doubt, observing the dwindling number of young women engaged in the spinning of wool, predicted that within a generation the entire country would be left naked.*

*So it is with the electronic revolution. Some are appalled that the calculator has diminished our ability to do arithmetic in our heads and predict a nation of numbskulls. Others rejoice that they can now do arithmetic to eight decimal places of accuracy on their forty dollar wrist watches. Some see today's video games as the final dissolution of civilization. Others see their five year olds mastering typewriter keyboards in order to peck out RUN SPACEINVADERS. Some are outraged that a large Chevy costing $3000 has been replaced by a small one costing $6,000 and complain that our standard of living will soon be reduced to that of Neandrethal man. Others are amazed that computers costing $300,000 have been replaced by those costing $3,000 and feel a sense of power and control over their lives unknown to previous generations.*

*Whatever the case, man's electronic extensions of himself are still only in their infancy.*

*Sam L. Savage 8 June 1982*

*Post Script – Upon presenting the above to my word processor's spelling checker, it politely informed me that I should have written "Neanderthal".*

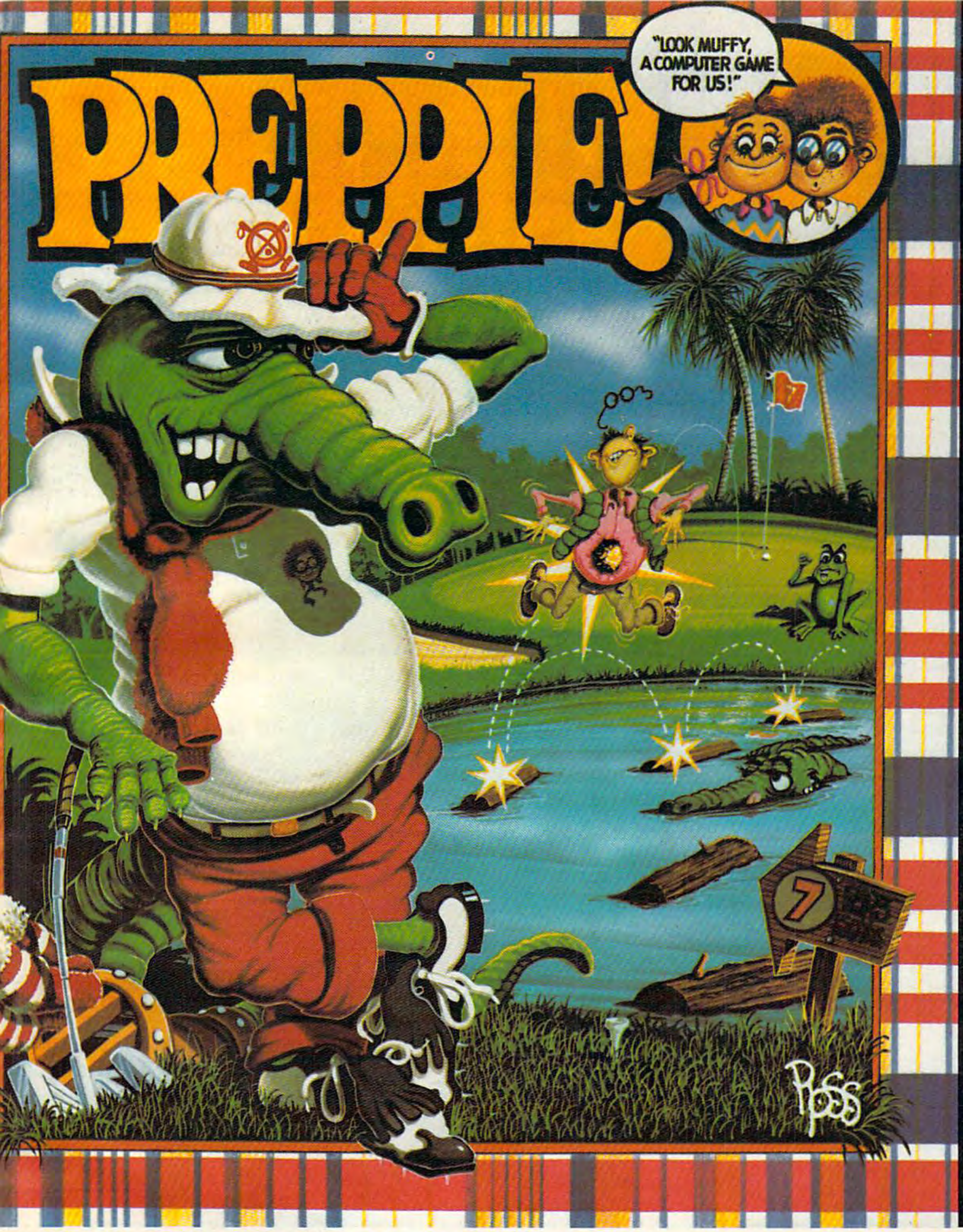
While children with access to computers are still in the minority, the trend is clear. A survey by a major merchandizing magazine showed that 5% of a national sample of families already had a personal computer in the house, and that an additional 5% were planning on purchasing one in 1982! What technology is it that will be moving into people's homes this year? For insight on that topic the CES is a tremendous resource.

You may recall that my report from the January CES in Las Vegas discussed the Ultimax from Commodore – a $150 computer to be introduced this year. Commodore made two changes between shows. They changed the name to the Max Machine and raised the price of the 2.5K RAM version of the machine to $179.95. Given the mass market channels (such as Toys R Us) through which the VIC-20 is presently being sold (how does a worldwide volume of 40,000 units per month grab you?), I would not be surprised to see the Max discounted to the original $150 price shortly after its October release.

I was also curious to see what Mr. Sinclair had up his sleeve. With his present computers being made by Timex at its plant in Dundee, Scotland, clearly he could devote his energies to the next product. He has.

### The New Spectrum

The new Sinclair computer is called the ZX Spectrum – a color computer with 16 or 48K of RAM. This computer is already being sold in the UK for




# Good Golly! What A GAME!

The original arcade game **PREPPIE!** will give you hours of toe-tickling fun. **You** take the controls, moving your young prepster about on the golf course. And what a golf course! It's unlike any you have ever played on before. There are dangers **everywhere,** and only you can keep the little Ivy League from a fate worse then Penn State!

**PREPPIE!** is written in state-of-the-art machine language, is joystick compatible and fully utilizes the Atari's sound and graphics capabilities. Quite frankly, it will give you the most fun you can have on an Atari microcomputer.

**PREPPIE!** is available at fine computer, book, and hobby stores everywhere.

ATARI 16K TAPE — $29.95
ATARI 32K DISK — $29.95

An exclusive game from

ai Adventure INTERNATIONAL

A DIVISION OF SCOTT ADAMS, INC.

To order, see your local dealer. If he does not have the program you want, then call 1-800-327-7172 (orders only) or write for our free catalog.

**Published by ADVENTURE INTERNATIONAL**
a division of Scott Adams, Inc.
BOX 3435 • LONGWOOD, FL 32750 • (305) 862-6917

"My Atari never did things like this before!"
*—Holister Townsend Wolfe*

"I had so much fun I almost blew my doughnuts."
*—Theodore Boston III*

"I haven't had this much fun since Buffy and I went to Princeton for the weekend."
*—Martha Vineyard*

under $350. The US price for the 16K version was not announced, but will probably be $199.95. Unlike the ZX-81, the Spectrum uses a nice mechanical keyboard with well-spaced medium travel keys. It sports a high resolution graphics mode with eight colors. Given the memory size, display capabilities, and mechanical (rather than membrane) keyboard, the Spectrum is a strong response to the Commodore Max.

Even if the Spectrum costs more than Max, the Sinclair distribution channels might more than make up for this deficiency. For example, a one-shot campaign for the ZX-81 was undertaken by American Express. The immediate response was about 2,000 orders per day. Not bad for a product with the ZX-81's limitations. With Timex selling through its 100,000 retail outlets, it will only be a matter of time before the ZX Spectrum will be available at your corner drugstore.

Perhaps Sinclair's *next* product will be a pocket TV to use as a display for these computers.

If nothing else, the battle between Sinclair and Commodore should dispel the myth of the so-called Japanese invasion – at least in this price range. At a higher price range, the NEC PC-6000 is quite impressive. This 16 to 48K RAM computer has three graphics modes as well as a 32 character by 16 line text mode. A printer interface and joystick ports are standard, as is a built-in 16K BASIC and a screen editor. While the price was not announced, I would guess that this product will retail for under $450. A graphics tablet (about $200) was also demonstrated for this computer.

Speaking of input devices, this CES was unique in the number of companies that were selling advanced joysticks and trak balls. WICO, a principal manufacturer of arcade game controllers, has adapted its devices for home use. For a modest $70 expense, your home Atari games can now use the same trak ball controllers found at the local pizza parlor arcade. Cynex, of Hillsdale, NJ, introduced GAME MATE II, a pair of radio remote control joysticks that let you control your computer without wires from up to 20 or 30 feet away. Aside from reducing the clutter of cables, this product opens the door to some very interesting applications for handicapped computer users.

And if anyone still needed proof that computers were mass-market items, what better example could one have than the fact that entertainment giants such as Thorn EMI (from England) and 20th Century Fox are now in the computer software business?

Yes, it promises to be an interesting year for that additional 5% of the US population that will also hear their children say, "Hey, I'm getting a disk read error!" ©


THEY CAME OUT OF THE BLUE OF THE WESTERN SKY
and it's up to you to stop them!
REAR GUARD
FEATURING ARCADE ACTION GRAPHICS AND SOUND
You know those alien residents who look like blue jello and smell like a cow pasture on a warm day? Well, they're up in arms about something and have declared war! Their weapons are Cyborg ships with their intelligent parts suspended inside. These Cyborgs pursue like crazed bats to try and kamikaze your crew pods. You're armed with an unlimited supply of Guided Energy Darts, but even when you blast 'em, their doomsday missiles come right at you! Good luck. (You'll need it!)
REAR GUARD features many extras which are usually found only on coin-operated video arcade machines. These include advanced play levels, running high score (with high score "clear" option), two player option, brilliant horizontal scrolling, and great sound effects.
Adventure International
APPLE VERSION
by John Anderson
48K DISK $29.95
ATARI VERSION
by Neil Larimer
Assisted by Sparky Starks
WE ARE OVERRUN
...ALL IS LOST
16K TAPE $19.95 24K DISK $24.95
TRS-80 VERSION
by Wayne Westmoreland
and Terry Gilman
16K TAPE $19.95 32K DISK $24.95
TO ORDER!
Adventure INTERNATIONAL
A DIVISION OF SCOTT ADAMS, INC.
BOX 3435, LONGWOOD FL 32750
(305) 862-6917 (QUESTIONS)
ORDER FROM YOUR FAVORITE DEALER
or CALL TOLL FREE (800) 327-7172
(ORDERS ONLY PLEASE)
120

*The Atari and Apple produce sounds of greater purity than some synthesizers, according to Frank Serafine, one of the creators of the* TRON *sound track. He used these computers to create some of the movie's most memorable sound effects.*

# The Sounds Of TRON

Tom R. Halfhill
Features Editor

*His eyes locked in the grim intensity of combat, the video warrior takes aim at his opponent, slowly winds up like a baseball pitcher, and violently unleashes a glowing discus. Whooshing and screeching, the disc arcs over the arena toward its frantically dodging target – who has dodged too late. The disc demolishes the enemy gladiator in a split-second burst of light and sound.*

*The victorious warrior reaches up to retrieve his deadly disc, which zooms back to him like a boomerang. Standing tall in his glowing armor, TRON has triumphed again over the forces of the sinister Master Control Program.*

This early scene from Walt Disney Productions' summer release, *TRON*, is typical of the film's pioneering use of animated computer graphics. But while *TRON*'s stunning visual effects have attracted the most attention from critics and audiences, fewer people are aware that the film's sound effects break new ground, too. The sounds, like the visuals, also are the product of high-technology computerization.

### Apple And Atari Sounds

What is even more interesting, at least for personal computer enthusiasts, is the equipment used to fashion those sound effects: many were generated with an off-the-shelf Atari 800 and an Apple II.

Not only that, but an Atari 800 running a commonly available data base manager program was used to store, categorize, index, and instantly retrieve all the special sound effects collected for *TRON*. How many was that? "Oh, my gosh – thousands, just thousands of sound effects," says one of their key creators, Frank Serafine, of Serafine FX Music/Sound Design.

SFX – you can see its credits roll by at the end of *TRON*, if you watch closely – is a Los Angeles-based sound design studio hired by Disney to collect and create most of the special sounds heard in *TRON*. Equipped with the most advanced audio components and sound-dedicated computers available, SFX previously had done sound effects for *Star Trek: The Motion Picture* and *The Fog*. But *TRON* was the studio's most innovative and involved project by far – SFX labored for a year and three months shaping the film's sounds.

### Electronic Sound Assembly

Serafine says the job would have taken even longer had he used the established method, improved little since the 1930s, of cutting and splicing bits and pieces of the soundtrack on mechanical film editors. Faced with a staggering task, Serafine turned to computers.

With his broad array of audio and computer equipment, including the Atari and Apple, Serafine designed a process he calls "Electronic Sound Assembly." Used for the first time in *TRON*, it does for the creation of sound what word processing does for writing: it allows the manipulation and fine-tuning of the work on a video screen. Serafine can digitalize sound effects, feed them into a Fairlight CMI (Computer Musical Instrument), plot the waveforms on a monitor and tinker with them almost endlessly. In fact, he can actually alter the sound directly on the screen with a light pen.

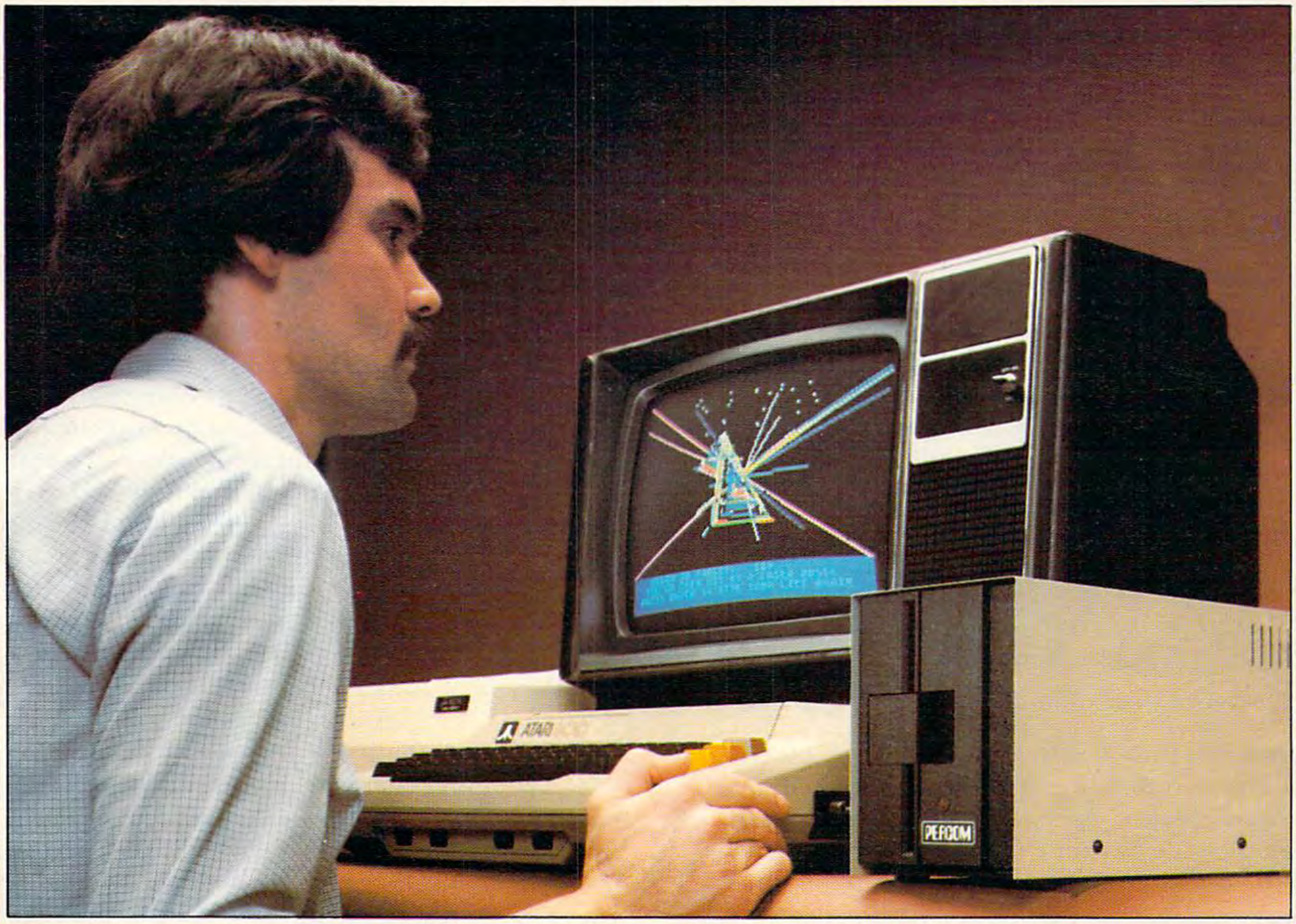
The results are sounds honed to an unbelievable degree of detail. Serafine says he was inspired by the sound-layering techniques pioneered by the Beatles in the 1960s with considerably less sophisticated equipment. "They achieved a subliminal effect, something which made you want to listen to their music over and over again to hear every sound. That's what I tried to do with the sound effects in *TRON*."

Consider the roar of the "light cycles," the futuristic motorcycles on which the gladiators of *TRON* duel while racing along the circuit grids of the computer in which they are trapped. To make the light cycles seem real, Serafine assembled and combined more than 50 different sounds. When a cycle makes a 90-degree turn to cut off an opponent, for example, the sound effect is a combination of video game tones generated on the Atari and a recording of a buzz saw.

Likewise, the sound of the sleek tanks which prowl the circuit canyons in *TRON* are a compilation of dozens of noises made with the Atari, all layered together. Serafine first tried recording a real army tank, but was disappointed by the clanking rattle. "I wanted something that sounded more turbine-like, more computer-controlled," he explains.

### Screaming Monkeys

Other notable sound effects in *TRON* generated by




# Introducing the PERCOM Alternative to ATARI Disk Storage

Your Atari 800 is the finest home computer on the market. Now you can own a floppy disk system that measures up — an RFD mini-disk storage system from Percom.

At Percom we've been making disk storage systems since 1977.

Our designs are proven, our quality is well known. And we back our dealers with service know-how. Expect more from Percom. You won't be disappointed.

- Operate in either single- or double-density storage mode using Atari DOS 2.OS. In double-density you can store almost 184 Kbytes (formatted) on one side of a 40-track diskette.
- Connect your Percom RFD first-drive system directly to your computer or connect into your system through your Atari 810 Disk Drive.
- Add an RFD first-drive system with its versatile four-drive controller, then connect as many as three more low-cost RFD add-on drives.
- Write application programs that can query and set up your system to operate a different type drive at each cable position — that can even change configuration as the program executes.
- Get quality and state-of-the-art capability at competitive prices. Percom first-drive RFD systems are priced from only $799, first add-on drive is only $459. Cables included.

**Watch for announcement of a new, powerful, easy-to-use disk-operating system for your Percom-equipped Atari 800 computer.**

Minimum system requirements — are an Atari 800 computer with 24-Kbytes of RAM and compatible video display system; Atari's disk-operating system (ver 2.OS) and owner's manual; and, for add-on drives (if used) an optional disk drives interconnecting cable available from Percom.

**For the best thing next to your computer, see your Atari dealer about a Percom RFD floppy disk storage system. For the name of your nearest dealer, call Percom toll-free 1-800-527-1222.**

Yes . . . I'd like to know more about Percom RFD disk drives for my Atari 800 Computer. Rush me free literature.
Send to: PERCOM DATA COMPANY, Inc., Dept. 18-C01
11220 Pagemill Road, Dallas, Texas 75243

name
street
city state
zip phone number
I plan to add a hard disk system to my Atari: ☐ yes ☐ no.
MAIL TODAY!

PERCOM

PRICES AND SPECIFICATIONS SUBJECT TO CHANGE WITHOUT NOTICE.

**PERCOM DATA COMPANY, INC.**
11220 PAGEMILL RD. • DALLAS, TX 75243 • (214) 340-7081

*Serafine in his studio, flanked by his Atari and Apple.*

the Atari include numerous video game bleeps and zaps, the sound of the "grid bugs" which arise from the circuits below the "solar sailer," the shock prods wielded by the Master Control Program's guards, and the collisions of tanks with walls (combined with recordings of real military explosions).

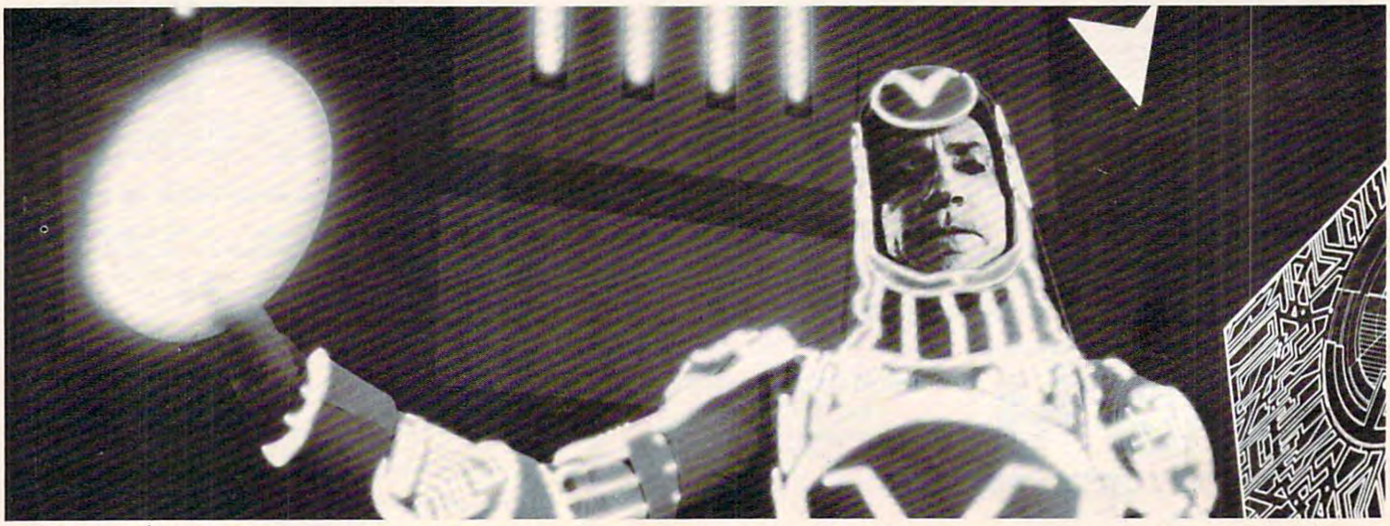
The Apple II was used with plug-in sound cards from Mountain Hardware and an Alpha Centauri keyboard. These add-ons made the Apple capable of a wider range of sound effects than the Atari, says Serafine, but also made it harder to use. Still, the added capability was an advantage when programming certain video game sounds and the "bonging" noise of a thrown discus.

The discus sound is a good example of how much work went into each effect. The Apple's "bonging" sound was overdubbed with recordings of a bullwhip and of monkeys screaming at the San Diego Zoo.

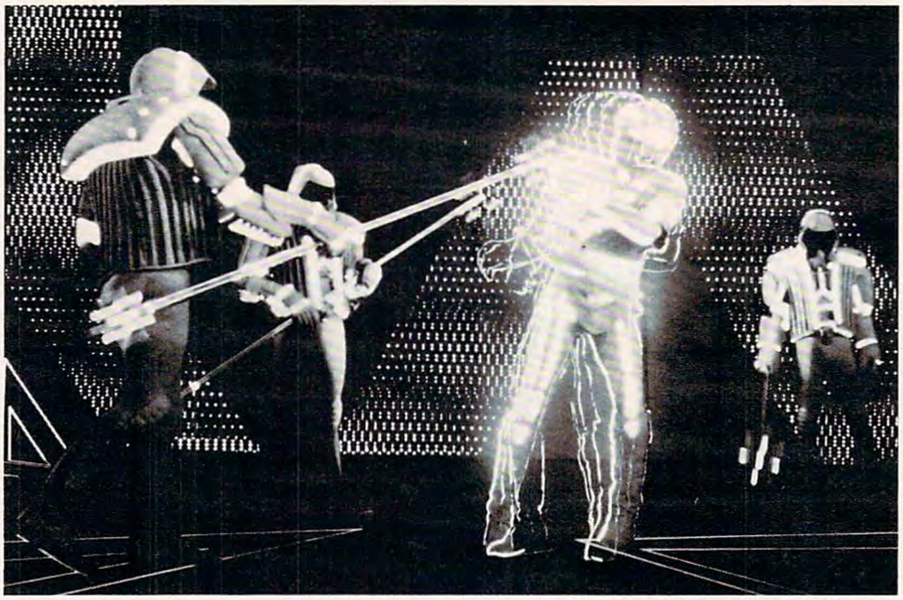
"The director [Steven Lisberger] demanded a concept for each sound effect," explains Serafine. "I couldn't just go around doodling with sound effects. We had to sit around with the director in discussion sessions to talk over the concept of each sound effect. Like, for the disc-throwing sound, we came up with the concept that they had to sound beautiful, yet sad – sad because something so beautiful can at the same time kill. So overlaying the monkey screams lets you know that, although this flying disc is really beautiful, you also know you'd hate to be hit by it."

It was originally Disney's idea to involve personal computers in the sound production. Serafine's background is in audio and multi-media presentations, not computers. He was designing planetarium shows in Colorado in 1976 when he first attracted Disney's attention. Disney hired him to put together a multi-media presentation for the grand opening of Space Mountain at Disneyland in 1977.

A few years later, after Serafine founded SFX, Disney hired him to create the sound effects for *TRON* and sent him to "Silicon Valley" in California, the home of America's microcomputer industry. Disney figured it was the ideal place to find video game sound effects for a movie whose central theme was to be video games. Serafine met with representatives from Apple and Atari, who set him



*Sark, assistant to the arch-villain Master Control Program, introduces the killer disc.*



© MCMLXXXII Walt Disney Productions.

*Crack Programmer Flynn finds himself at the mercy of distinctly user-unfriendly guard programs.*

up with computer systems. Atari also lent him an Atari Sound Development Disk, a well-guarded, powerful utility package rarely entrusted to anyone outside Atari itself.

Most of the Atari sounds were not actually programmed by Serafine, though. He got help from Ed Rotberg, an ace Atari programmer who has since left Atari to form his own company, and from another expert – Laurent Basset, who is 17. Basset is the son of one of Serafine's friends.

"He's a whiz kid," says Serafine. "This kid was actually able to do anything I wanted done on the machine. I would dream of a sound or a concept, and he would come back to me the next day with the finished programs."

All the thousands of sound effects created on the computers or collected on tape were cataloged on the Atari with *FileManager 800*, a data base program by Synapse Software. Serafine says it saved his studio hours of tedious filing. The record for any sound effect, listing its characteristics, source, and location on tape, could be retrieved in 1.5 seconds.

### A Clean, Pure Sound

Serafine, who had only limited previous experience with personal microcomputers, was also impressed with the sound quality of the machines. "The amazing thing to me is the purity of the sound that comes out of the Atari, and also the Apple. Their sound chips produce an extremely clean, pure sound which is even superior to some synthesizers I've worked with. We had no trouble using those sounds in the movie."

In fact, he thinks the sound capabilities of the computers are underused, partly because the proper tools are not available. "I think Atari and Apple will look at the success of the personal computers [in *TRON*] and develop better sound development disks as a result. For example, Atari has a *Music Composer* which gives you all the tools you need to compose music except really good sound. You can't access all the good sounds in the machine with the *Music Composer*. Someone at Atari ought to combine the *Music Composer* with features of the Sound Development disk, and they'd really have something. Maybe if they read this...."

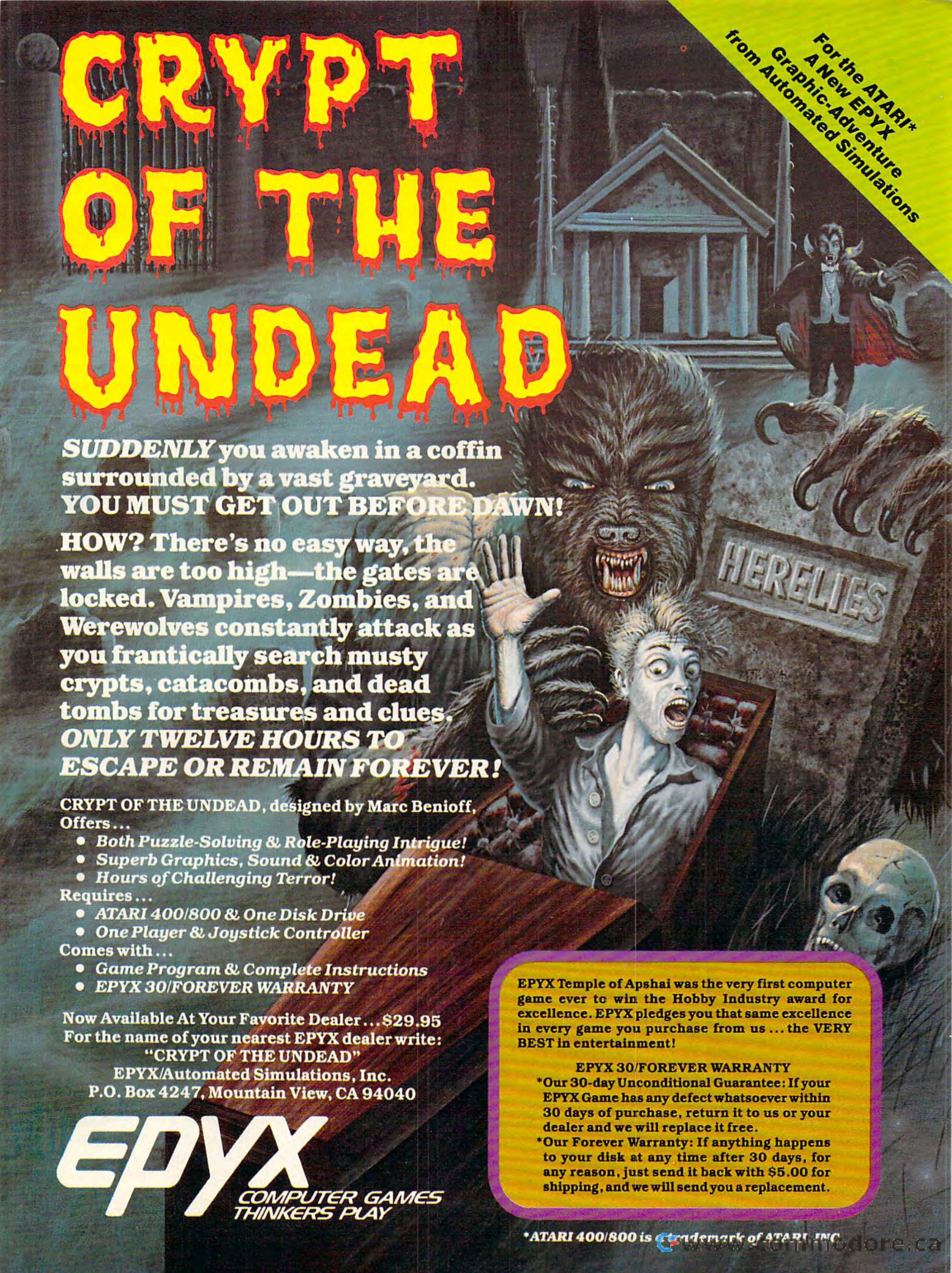
Serafine sees a future for personal computers in other productions involving video game-like effects. In fact, his current project is a *Pac-man* commercial for Seven-Up.

Unfortunately, most people take sound effects for granted, he says. The sounds in *TRON* come and go so fast that almost nobody realizes the amount of labor involved. "Several people spend a week of intense work to create something," he sighs, "and it lasts only one second."

**Three Atari sound effects by Laurent Basset, who helped program many of the Atari sounds in *TRON*.**

```
10 ? CHR$(125):POKE 710,80:POKE 755,0:? "
THUNDER & RAIN"
14 FOR TIME=0 TO 1
15 B=INT(255*RND(0)+50):X=RND(0)*200
20 FOR PITCH=1 TO B:SOUND 0,PITCH,8,15:NEXT PITCH
25 FOR T=1 TO X:NEXT T:NEXT TIME:SOUND 0,0,0,0
27 FOR RAIN=0 TO 15 STEP 0.2:SOUND
1,0,0,RAIN:FOR W=1 TO 20:NEXT W:NEXT RAIN:FOR
W=1 TO 2000:NEXT W:SOUND 1,0,0,0
40 FOR X=1 TO 100:NEXT X
50 ? CHR$(125):POKE 710,80:POKE 755,0:? "
HEART BEAT"
60 FOR Y=1 TO 5:FOR W=1 TO 40:SOUND
0,12,3,15:NEXT W:FOR W=1 TO 150:SOUND
0,0,0,0:NEXT W:NEXT Y:SOUND 0,0,0,0
70 FOR X=1 TO 200:NEXT X
80 ? CHR$(125):POKE 710,80:POKE 755,0:? "
STEAM LOCOMOTIVE"
90 X=0.1
100 FOR W=1 TO 150
110 FOR LOUD=15 TO 4 STEP -X:SOUND
0,15,0,LOUD:NEXT LOUD
120 X=X+0.01:IF X>0.7 THEN X=0.7
130 NEXT W:SOUND 0,0,0,0
135 FOR S=1 TO 2
140 SOUND 1,40,10,10:SOUND 2,10,10,8:SOUND
3,90,10,10
145 FOR X=1 TO 200:NEXT X
147 SOUND 1,0,0,0:SOUND 2,0,0,0:SOUND 3,0,0,0
148 FOR X=1 TO 60:NEXT X
150 NEXT S
```

©


For the ATARI*
A New EPYX
Graphic-Adventure
from Automated Simulations
CRYPT OF THE UNDEAD
SUDDENLY you awaken in a coffin surrounded by a vast graveyard.
YOU MUST GET OUT BEFORE DAWN!
HOW? There's no easy way, the walls are too high—the gates are locked. Vampires, Zombies, and Werewolves constantly attack as you frantically search musty crypts, catacombs, and dead tombs for treasures and clues.
ONLY TWELVE HOURS TO ESCAPE OR REMAIN FOREVER!
HERELIES
CRYPT OF THE UNDEAD, designed by Marc Benioff, Offers...
• Both Puzzle-Solving & Role-Playing Intrigue!
• Superb Graphics, Sound & Color Animation!
• Hours of Challenging Terror!
Requires...
• ATARI 400/800 & One Disk Drive
• One Player & Joystick Controller
Comes with...
• Game Program & Complete Instructions
• EPYX 30/FOREVER WARRANTY
Now Available At Your Favorite Dealer...$29.95
For the name of your nearest EPYX dealer write:
"CRYPT OF THE UNDEAD"
EPYX/Automated Simulations, Inc.
P.O. Box 4247, Mountain View, CA 94040
EPYX
COMPUTER GAMES THINKERS PLAY
EPYX Temple of Apshai was the very first computer game ever to win the Hobby Industry award for excellence. EPYX pledges you that same excellence in every game you purchase from us ... the VERY BEST in entertainment!
EPYX 30/FOREVER WARRANTY
*Our 30-day Unconditional Guarantee: If your EPYX Game has any defect whatsoever within 30 days of purchase, return it to us or your dealer and we will replace it free.
*Our Forever Warranty: If anything happens to your disk at any time after 30 days, for any reason, just send it back with $5.00 for shipping, and we will send you a replacement.
*ATARI 400/800 is a trademark of ATARI, INC
www.commodore.ca

## The Beginner's Page

# How Computers Think

Richard Mansfield
Senior Editor

Computers don't yet think in the broadest sense. Some believe that computers never will match the human mind in overall mental ability. Others argue that artificial intelligence is inevitable and might come to pass within the next ten to fifteen years. Before we can consider this question, however, we should first look briefly at the mechanics of computer thinking. Your personal computer does "think" in a way. It can remember, it can play a good game of chess, it can make certain kinds of decisions, and it performs math calculations far faster than we do.

How does your computer do the "thinking" that makes it a worthy chess opponent? For the answer, we have to get down to the simplest level. If we could enter the silicon chips inside the machine, we would see a vast pattern of intersections and pathways, like Los Angeles seen at night from a jet. Which path the electricity follows determines what happens. The computer controls this electric traffic with *gates*. There are great numbers of gates (the intersections), and they are the heart of the computer's decision-making process. Each gate makes a yes or no decision, like a traffic cop who signals "stop" or "go," to the electricity trying to flow past it.

How does this "gate deciding" take place in physical terms? The symbol of a gate looks like this:

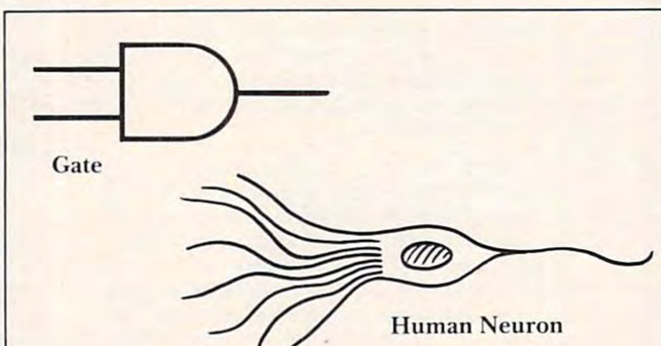



Figure 1.

with two lines (wires) coming into the gate and one line going out. The incoming lines are the facts, the basis for the judgment, and the line out announces the decision. A gate is a transistor (or diode). It functions like an automatic traffic light which turns green only after two cars have pulled up to the intersection. Beneath the pavement, in each lane, is a sensor which can tell if a car is sitting on it. The light stays red until both sensors are switched on.

If there is only one car in either lane, or no cars, the light stays red. With only these weight sensors connected by wires to the gate, a true decision can be made. The wire coming out of the gate is "turned on" when both incoming wires are on. This "out" wire is connected to the traffic light. In this way, electricity flowing through wires and gates can *decide* things, can think.

A gate which says "yes" when both of its incoming wires are on is called an AND gate. Both one wire AND the other must be on for the AND gate to say "yes." It is easy to see that this gate could have many uses. How would your computer decide whether to put a capital or a lowercase "C" on your TV screen when you press "C" on the keyboard?

### Using Gates In BASIC

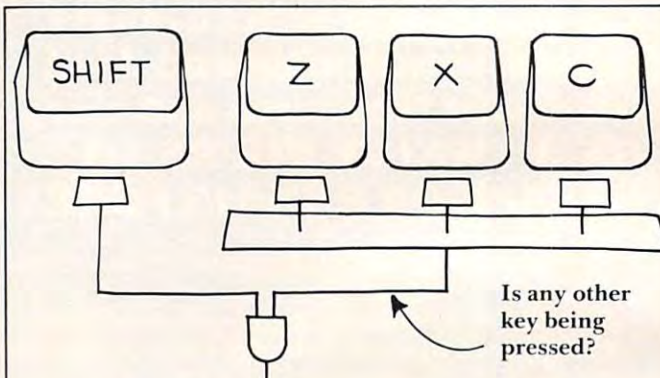



Figure 2.

There are multitudes of AND gates inside your computer. One of them could always be checking to see if a signal is coming in from the shift key at the same time as another signal is coming from a regular key. If *both* of these wires are on at the same time, the shift decision is made, so the wire coming out of the "Shift AND Gate" turns on and the computer displays an uppercase "C."

Less common, but also important, is the OR gate. It looks roughly the same as AND with two wires in and one out, but it says "yes" if *either* incoming wire is on. It might be useful for night traffic. If there is a car in either lane, the light switches to green.

There are times when the best way to solve a

# Good News for Kids...


Adventure on a Boat


Robby the Robot Catcher


Sky Rescue

**The SubLOGIC line of children's software (ages 4-12) is available for the Atari © 400™ and 800™ computers.**

## Adventure on a Boat

Go fishing for points in the waters of Fantasy where anything can happen. 32K

## Robby the Robot Catcher

Baby robots are falling from the skies. Catch them if you can. 32K

## Sky Rescue

Use your helicopter to rescue the people of Irata from the Mad Bomber. 32K

**These and other children's games are on cassette for $19.95 or on disk for $24.95.**

***See your dealer...***

or for direct orders, add $1.50 and specify UPS or first class mail. Illinois residents add 5% sales tax. Visa and MasterCard accepted.

subLOGIC

Communications Corp.
713 Edgebrook Drive
Champaign, IL 61820
(217) 359-8482
Telex: 206995

programming problem in BASIC is to set up your own gates. If your computer's BASIC includes the words AND and OR, you can use them to experiment with this concept of gates. You could try:

```
10 INPUT A: INPUT B
20 IF A = 10 AND B = 5 THEN PRINT" AND ": GOT
   O 10
30 IF A = 10 OR B = 5 THEN PRINT" OR ": GOTO
   10
40 PRINT" NEITHER AND NOR OR ": GOTO 10
```

or you could get more complicated by, for example, checking to see if both numbers are less than 100 and B is 1/2 as large as A:

```
20 IF A < 100 AND A/2 = B THEN PRINT "CONDITI
   ONS SATISFIED"
```

The computer's "mind" is built of AND and OR gates, lots of them, arranged in various ways, to form the pathways along which an electric impulse flows. This spark races through the computer, darting now left, now right, at the gates. Things work similarly in the human brain, but there are significant differences.

### A Thousand Ideas About Orange

In our brain, the wires leading into a gate are nerve fibers, the gates themselves are neurons, and the wire coming out of the gate (with the answer) is called an axon. Perhaps the most important difference is that a computer normally has two wires bringing facts into a gate for a decision. Some neurons have up to 100,000 incoming wires. A single decision can be influenced by that many "facts." Fortunately for us, the facts don't all need to be "true" to switch an AND neuron. The outgoing axon can say "yes" based on a percentage of the incoming votes.

Another significant difference contributing to the sophistication of human thought is that we can think many things at the same time. When you imagine an orange, your brain activates up to perhaps 1,000 separate thinking processes simultaneously. It provides pictures (of both the color and the fruit and possibly throws in an "associated" idea or two like a photo of an orangutan ape), taste memories, and hundreds of other pieces of information it has wired to the word *orange*. And, at the same time, it is selecting which of these thoughts to ignore, which to connect, and which have the highest value at the moment.

### Computer Gates Can Switch A Million Times Each Second

Your computer, by contrast, can process only one "thought" at a time. This sort of thinking (A leads to B which leads to C) lends itself very well to math problems, but more or less eliminates poetry, common sense, or any thinking involving creativity or flexibility. At present, computers are excellent at solving problems where there is only one right answer, but weak with ambiguous tasks (including mastery of languages with their shadings and nuances).

Underlying these differences are some interesting contrasts between your brain and your computer's chips. Some estimates place the total number of gates in the brain at 10,000,000,000. Personal computers have far less density. The brain's memory may have 12,500,000,000,000,000,000 bytes (memory cells). We consider ourselves lucky if our computer has 48,000 bytes. This imbalance in size and complexity is somewhat offset by the great speed of the computer. A computer's gate can swing open or slam shut one million times every second. Our gates can open or close only 100 times per second. (But 1,000 of them can be operating at once, so it's not as bad as it sounds.)

At the vanguard of computer science, however, the gap is closing. The newest chips are coming close to the density of the human brain. Advances are being made in creating "parallel processors," computers which can, like us, handle many thoughts simultaneously. Gates with multiple incoming wires are being built. The latest computer speed records leave us hopelessly in the dust: some working computers can switch a gate in a nanosecond (one billionth of a second).

Will computers ever become intelligent in our sense of the term? Will a new life form, *silicon sapiens*, dominate Earth? Nobody knows. There is one thing to consider though: our mental machinery isn't getting any bigger or faster. We have had our present brain capacity and speed since prehistoric times. The modern electronic computer first appeared in the 1940's, it has rapidly improved since, and we cannot now say that there is any known limit to the ultimate power of a computer mind. ©

# POOL 1.5™

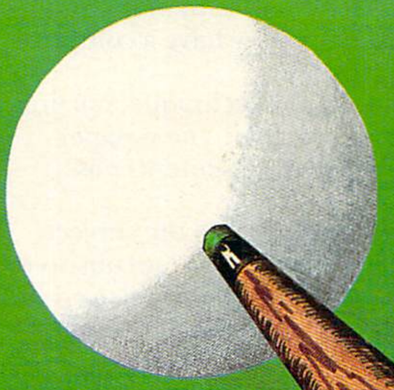
# ATARI™ OWNERS,

NOW YOU TOO CAN PLAY POOL 1.5 ON YOUR ATARI 48K DISK SYSTEM.

ALSO AVAILABLE FOR THE APPLE II™ 48K DISK SYSTEM.

**New! POOL 400 in Cartridge form for your 16K Atari 400/800 $39.95, requires joystick.**

PRICE: $34.95



- **High resolution graphics-Life like motion.**
- **Instant replay on any shot.**
- **Superfine aiming.**
- **Pool room sound effects**
- **Choice of 4 popular games**

**See your local computer dealer or order directly from IDSI. Specify ATARI or APPLE.**

IDSI
P.O. BOX 1658
LAS CRUCES, NM
88004
Ph. (505) 522-7373

ATARI is the registered trademark of ATARI Inc.
APPLE II is the registered trademark of APPLE Computer Inc.
POOL 1.5 is trademark of IDSI.

*This program (for Atari and Microsoft BASICs) will aid in improving and maintaining speed reading skills.*

# Peripheral Vision Exerciser

Ron Kushnier
Richboro, PA

We have all seen or heard about people who can read a lengthy novel in an hour. Their eyes scan the pages faster than most of us can read a sentence. How do they do it? How much comprehension is really taking place?

Those who have taken speed reading courses have said that it is a matter of extreme concentration and practice, mostly practice. Their reading speed depends on the complexity of the reading material – a novel goes much faster than a technical article. Use of the technique must be constant. Once a reader strays from the method, he quickly forgets it.

The major technique used in speed reading is the concept of the word picture. When we see a word like *the*, we do not see it as " t...h...e ," but rather as an entire word picture. If we extend this idea and add another word, like *on the*, most of us can still see the entire picture rather than separate words. If we add *on the hill*, we now have a complete thought in one word picture.

Speed reading builds on this technique, starting with single words and progressing to newspaper columns, and finally to full-length sentences as found in standard texts.

A recent TV program devoted to the subject showed several speed reading tools. These consisted of charts of random letters appropriately spaced to exercise the peripheral reading vision of the student. I immediately thought, "Wow! What a terrific application for a computer!" Not only could I produce such a chart, but I could also animate it.

Those who are interested in trying speed reading may find this Peripheral Vision Exerciser of interest.

The program generates lists of random letters formatted in such a way that the field of view is gradually expanded. The eyes take in more letters at one time in this way.

The program consists of two levels: beginner and advanced. Concentrate on the screen and try to read the groups of letters with one eye fixation, that is, without moving your eyes *across* the screen. Press the SPACE key to call up another set of letters. As you progress, press the "M" key (more separation) to make things a little harder. To stop, or to go into another mode, just hit the STOP key, then hit RUN.

**Program 1. Microsoft Version**

```
100 REM **PERIPHERAL VISION EXERCISER**
110 REM ** BY RON KUSHNIER **
120 INPUT"{CLEAR}ARE THESE CAPITAL LETTERS(Y/N
    )";A$
130 IFLEFT$(A$,1)="Y"THENPOKE59468,14:     F1=1
    :GOTO150
140 IFLEFT$(A$,1)<>"N"THEN120
150 INPUT"{CLEAR}DO YOU WANT LEVEL ONE(1) OR T
    WO(2)";A
160 IFA=1THENF=1:GOTO190
170 IFA=2THENF=0:GOTO190
180 GOTO150
190 INPUT"{CLEAR}STARTING SEPARATION( >=2 )  "
    ;B
200 PRINT"{CLEAR}"
210 IFF1=1THENA$="ABCDEFGHIJKLMNOPQRSTUVWXYZ":
    GOTO230
220 A$="ABCDEFGHIJKLMNOPQRSTUVWXYZ"
230 X=1
240 DEF FNL(X)=INT(26*RND(X)+1)
250 FOR I =1 TO 10
260 C$=MID$(A$,FNL(X),1)
270 IF F=1THEN300
280 C$=C$+MID$(A$,FNL(X),1)
290 C$=C$+MID$(A$,FNL(X),1)
300 D$=MID$(A$,FNL(X),1)
310 IF F=1THEN330
320 D$=D$+MID$(A$,FNL(X),1)
330 E$=MID$(A$,FNL(X),1)
340 IFF=1THEN360
350 E$=E$+MID$(A$,FNL(X),1)
360 IF F=1THEN PRINT TAB(18- B)D$; TAB(18)C$;T
    AB(18+B)E$;CHR$(13):GOTO380
370 PRINT TAB(18- B)D$; TAB(18)C$;TAB(19+B)E$;
    CHR$(13)
380 NEXT
390 POKE158,0
400 GETA$:IFA$=""THEN400
410 IF A$="M"THENIFB<18THENB=B+1:GOTO200
420 GOTO200
```

**Program 2. Atari Version**

```
100 REM *** PERIPHERAL VISION EXERCISER
    **
110 OPEN #1,4,0,"K:":GRAPHICS 0:POKE 752
    ,1
150 ? CHR$(125);"Do you want level one (
    1) or two (2)":GET #1,A:A=A-48
155 IF A<1 OR A>2 THEN 150
160 F=1:IF A=1 THEN F=0
190 ? CHR$(125);"Starting separation (2-
    18)";:INPUT B
195 IF B>18 THEN B=18
200 ? CHR$(125);:POKE 559,0
250 POSITION 18-B,10:V=F*2:GOSUB 500
260 POSITION 18,10:V=F*3:GOSUB 500
270 POSITION 18+B,10:V=F*2:GOSUB 500
280 POKE 559,34
290 GET #1,A
300 IF A=ASC("M") THEN IF B<18 THEN B=B+
    1:GOTO 200
310 GOTO 200
500 FOR I=1 TO V
510 ? CHR$(65+INT(26*RND(0)));
520 NEXT I
530 RETURN
```

©

*Though written for the PET, this in-depth exploration is useful to any computerist whose BASIC includes the DEF FN command.*

# User-defined Functions: Defined

Myron D. Miller
Indianola, PA

Have you ever written a program in which a certain formula had to be used quite often? Perhaps the fundamental expression was always the same, but a variable would need to change. Some formulas can get quite lengthy, and repetitious entry can become tedious. There is an alternative besides subroutines for this situation: the *user-defined function*. Let's take a look at the user-defined function.

I will be inventing some names in the article to help explain the operation of the function. Any term which is of my creation will be followed by "(my jargon)" when introduced. The first invented term is the abbreviation *UDF* (my jargon) for *U*ser-*D*efined *F*unction. Also, there are a lot of short and simple examples included in the article. Sit close to your PET and try the examples when you come across them. I think you will learn more by seeing the UDF in action.

### What Is A UDF?

A UDF is an arithmetic function that is defined by the user. Once defined, the UDF is implemented just like a normal, resident BASIC function, and can be used at any point within the program. A UDF is somewhat like a variable, but rather than being a variable for data, it is a variable for an arithmetic formula. Like a variable, a UDF can be defined, called up, and redefined as often as desired. While similar to a variable, a UDF is not a variable, and you should avoid thinking of it as a variable. A UDF is for number crunching operations only. As such, string variables, string functions, and other non-arithmetic operations must not appear in a UDF. The string symbol ($) must not appear in the UDF's syntax. Forget about strings when working with UDFs.

When should you use a UDF? Whenever it is convenient and practical to do so. Generally, they are used for complex formulas which are repeated many times throughout the program. However, they can be as simple as you like. It is chiefly a question of practicality. If a formula appears only once in a program, a UDF is clearly impractical. If it appears many times, it may then become practical to implement a UDF. Another feature of the UDF is that one of the variables of the formula is made available for independent substitutions. Thus, values can be plugged into the UDF without changing variables found in the program. I think you will find the UDF to be an interesting as well as powerful programming tool.

### UDF Syntax

Let's take a look at a UDF (see Figure 1 and Example 1). First, the UDF has two parts: 1. the definition statement – DEF FN...–. 2. the *execution statement* (my jargon) – FN...–. The definition statement tells the computer: "Remember this formula, we will need it later." This is similar to a LET for variables. The execution statement tells the computer: "Use that formula that I told you to remember." This is similar to calling up a variable for use. Figure 1 shows the required syntax incorporated into both statements of the UDF. Example 1 shows a simplified use of a UDF. (Note: the examples given in this article will emphasize simplicity rather than practicality.) We will take a detailed look at each segment of the UDF's syntax.

BASIC Keywords. DEF FN (*define function*) is the keyword that states: remember this function. It must have a line number; that is, you cannot use a

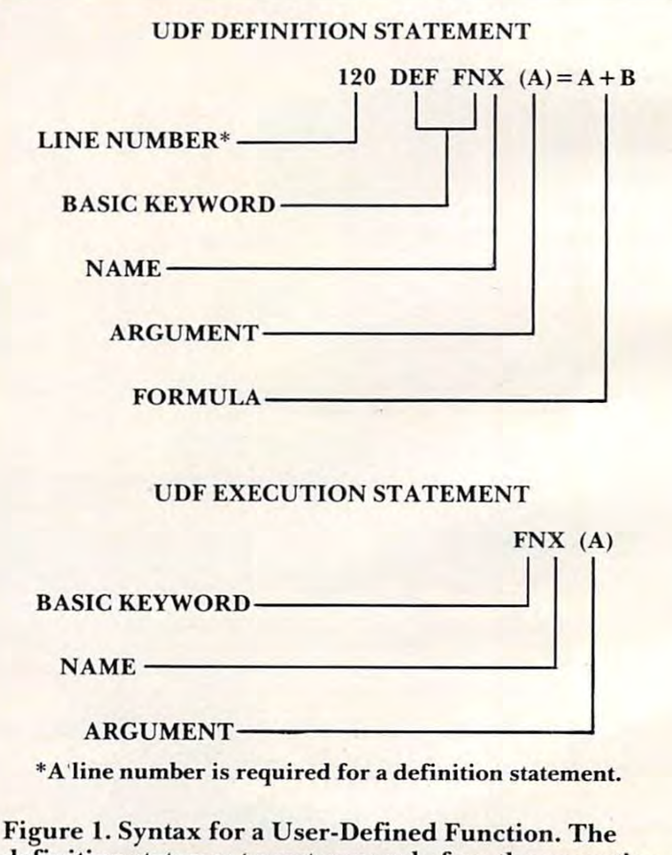



**Figure 1. Syntax for a User-Defined Function. The definition statement must appear before the execution statement in the program.**


# Make your fun and games computer get serious
## For only $79.95

VIC-20[1] The Wonder Computer of the 1980's. Less than $300. One heckuva lot of fun. But if you're using your VIC-20 strictly for recreation and fun, we at Micro-Systems think you're not getting your money's worth. Because we've designed the Micro-Systems VIE Cartridge which allows you to interface VIC to all existing Commodore IEEE peripherals (8050, 4040, 2031, and printers) for only **$79.95**. In addition, we've designed the V-232 RS-232 Interface board to allow the VIC-20 to communicate with various serial devices such as printers, modems, and other systems—**NOT JUST A BUFFER DRIVER**—for only **$45.00.** Also available are a full range of Ram Expansion Cartridges in increments of 8K, 16K, and 24K of ram expansion memory at very competitive prices. If you need to access more than one cartridge in your VIC-20 expansion port, we have the solution. The V-Expander is now available with 6 additional expansion ports DIP selectable.

And if you want to keep the fun in your computer. Micro-Systems has developed five all new video games **(VICTOROIDS, MOBILE ATTACK, CRACK, GUARDIAN, and QUARK)** Available in Cartridge; Cassette (requires 8K expansion); and disk (requires IEEE interface above and 8K expansion); all compatible with VIC-20.

### Available from Micro-Systems

| | | | |
|---|---|---|---|
| VIE Cartridge (IEEE-488) | $ 79.95 | V-Expander (6-slot) | $ 75.00 |
| VIC-20 Color Computer | $259.00 | V-232 RS-232 Interface | $ 45.00 |
| VIC-1540 Single Disk Drive | $499.00 | Joy Stock (Arcade Quality) | $ 29.95 |
| VIC-1515 Graphic Printer | $325.00 | VTE Terminal Emulator | |
| VIC Modem | $ 89.95 | Tape | $ 8.00 |
| VIC-1210 3K Expansion | $ 35.00 | Disk | $ 15.00 |
| V8K Ram Expansion | $ 49.95 | GAMES Cartridge | $ 29.95 |
| V16K Ram Expansion | $ 95.00 | Cassette | $ 19.95 |
| V24K Ram Expansion | $149.00 | Disk | $ 19.95 |

**DEALER INQUIRIES INVITED**

**Yes, Please send me:**

| QTY. | MODEL# | NAME | PRICE |
|---|---|---|---|
| | | | |
| | | | |
| | | | |
| | | | |
| TOTAL | (In Texas, add 5% sales tax) | | |

Your Name ____________________

Address ____________________

City __________ State __________ Zip __________

payment method: Check ☐ Card-Exp. Date __________

Master Card # ____________________

Visa # ____________________

American Express # ____________________

Signature ____________________

Micro-Systems • 11105 Shady Trail #103 • Dallas, Texas 75229 • (214) 484-7836

[1]Registered Trademark of Commodore Int.

**LISTING**

**110 A=3:B=5**
**120 DEF FNX (A)=A+B**
**130 PRINT FNX (A)**

**COMPUTER'S ACTIONS**

**110 A=3:B=5**
**120 Define a function named *X*.**
**FNX (A)=A+B**
**130 Execute the function named *X*.**
**FNX (A)=A+B**
**FNX (A)=3+5**
**FNX (A)=8 Print on screen.**

**RESULT – COMPUTER DISPLAYS:**
**8**
**READY.**

**Example 1. A simple usage of a User-Defined Function. Note: do not worry about the arguments for now. They will be explained later in the article.**

definition statement in the direct (calculator) mode. The definition statement must appear in the program before the execution statement. FN is the keyword that states: use the remembered function. The execution statement must appear after the definition statement. It can be used in direct mode, but only after the definition statement has been *run* in a program – *run*, not just listed.

### Rules For BASIC Keywords

1. DEF FN must have a line number.
2. DEF FN must appear before FN in the program.
3. FN may be used in direct mode if a DEF FN has been run in a program.

*Names:* A UDF must have a name. I chose the name "X" for the UDF in Figure 1 and Example 1. A program can have many different UDFs in it; we are not limited to one. Thus, the name identifies which particular UDF we wish to use. Even if the program contains only one UDF, it still requires a name. Also, both the definition statement and the execution statement must use the same name for any specific UDF.

UDF names follow the same rules as variable names. Thus we can use: a single letter – A,B,C...Z; or two letters – AA, AB, AC...AZ, BA, BB, BC...ZZ; or a letter followed by a number – A0, A1, A2...A9, B0, B1, B2...Z9. Like a variable, a UDF can use a longer name, and it will be plagued with the same problems that crop up when long names are used with variables. *First* – the entire definition statement (line number, keyword, name, argument, and formula) must fit on one 80 character line, just like any other program line. Thus, the more space that is used by a name, the less space left for a formula. *Second* – long names are used in the listing only – *not in the program run.* During the run, the computer looks at only the first two characters of the function or variable name. Thus, if two UDFs have long names that begin with the same first two characters (e.g., *RA*DIUS and *RA*DIANS), the first UDF that was entered will be redefined and lost when the second UDF is defined. Hence, no matter which of the two UDFs is called up, only the second UDF will be executed. To further complicate matters, the listing will appear to be correct because long

**LISTING**

**210 A=2:B=3:C=7:D=9**
**220 DEF FNRADIUS (Z)=A+B**
**230 PRINT FNRADIUS (Z)**
**240 DEF FNRADIANS (Z)=C+D**
**250 PRINT FNRADIANS (Z)**
**260 PRINT FNRADIUS (Z)**
**270 PRINT FNRABBIT (Z)**

**COMPUTER'S ACTIONS**

**210 A=2:B=3:C=7:D=9**
**220 Define a function named *RA*.**
**FNRA (Z)=A+B**
**230 Execute the function named *RA*.**
**FNRA (Z)=A+B**
**FNRA (Z)=2+3**
**FNRA (Z)=5 Print on screen.**
**240 Define a function named *RA*.**
**FNRA (Z)=C+D**
**250 Execute the function named *RA*.**
**FNRA (Z)=C+D**
**FNRA (Z)=7+9**
**FNRA (Z)=16 Print on screen.**
**260 Execute the function named *RA*.**
**Same action as line 250.**
**270 Execute the function named *RA*.**
**Same action as line 250.**

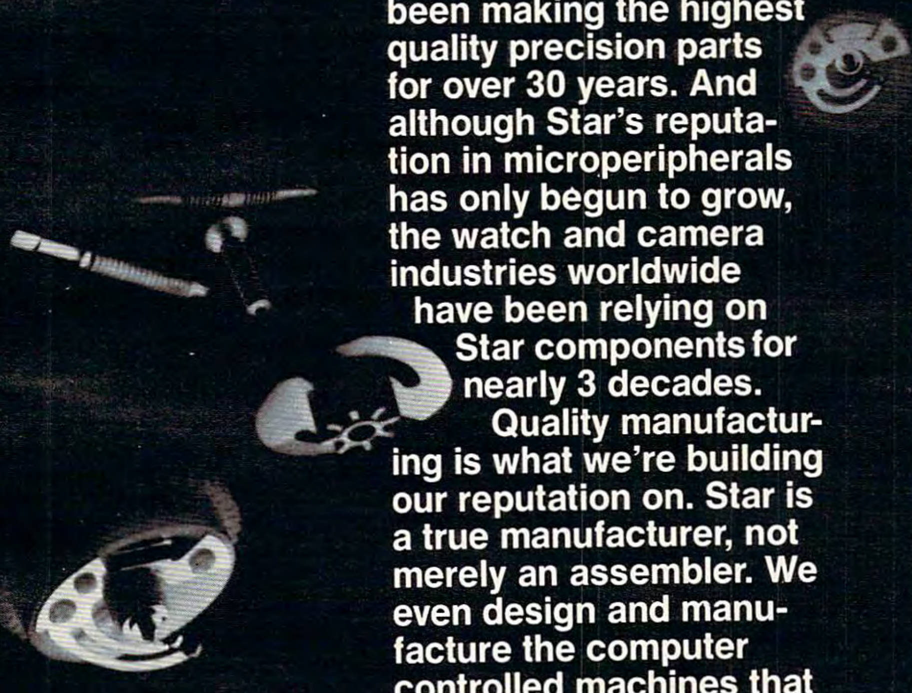
**RESULT – COMPUTER DISPLAYS:**
**5**
**16**
**16**
**16**
**READY.**

**Example 2. This program demonstrates the problem of using two UDFs with names that begin with the same first two characters. FNRADIUS is redefined and lost in line 240. Thus, FNRADIUS and FNRABBIT execute the same as FNRADIANS. This is because the computer operates with only the first two characters of the name – FNRA – during the run. The long names are used in the listing only. Again, for now, don't worry about the arguments.**


STAR.
WE'RE MAKING A

# NAME FOR OURSELVES.

## Building a reputation for quality printers of astronomical value, from components on up.



Making a name for ourselves in printers is only natural when we've been making the highest quality precision parts for over 30 years. And although Star's reputation in microperipherals has only begun to grow, the watch and camera industries worldwide have been relying on Star components for nearly 3 decades.

Quality manufacturing is what we're building our reputation on. Star is a true manufacturer, not merely an assembler. We even design and manufacture the computer controlled machines that make our parts. With unlimited production capabilities, Star's factories can operate 365 days a year, 24 hours a day, producing more than 100 million parts a month. It is this kind of productivity and total quality control that assures you of getting the very finest printer for the very lowest price.

And the best product for less is what you can depend on getting from Star. Star offers you a full line of reliable printers, printers with divergent technologies to meet your specific needs. And there's more. You get a low cost, high quality printer with a wide spectrum of interfaces to choose from—backed by Star's superior service network, there if you need it.

Star. We're going to be the quality printer you can count on, from components on up. And that's no pie in the sky promise.

**star**
**micronics • inc**

**MAKING A NAME FOR OURSELVES**
200 PARK AVENUE, NY, NY 10166

names are retained in the listing. (See Example 2.) *Third* – BASIC keywords cannot be nested (contained within) a two or more character name. Thus, the following names will cause a syntax error: *GO*, *INT*EGRATE, SUB*TO*TAL, KI*LOG*RAM, *FRE*QUENCY, FACT*OR*, SUBTRACTI*ON*, S*TAN*DARD. (See Example 3.)

**LISTING**

```
310 A=2:B=3
320 DEF FNSUBTOTAL (Z)=A+B
330 PRINT FNSUBTOTAL (Z)
```

**COMPUTER'S ACTIONS**

**310 A=2:B=3**

**320 Define a function named *SU*. TO ???? Computer thinks: Why is this TO here? I didn't see a FOR statement or a GO. I do not understand this. Print: ?SYNTAX ERROR IN 320. Terminate program execution.**

**330 Not executed because of SYNTAX ERROR IN 320. Will also cause a syntax error if 320 is corrected.**

**RESULT – COMPUTER DISPLAYS:**

```
?SYNTAX ERROR IN 320
READY.
```

**Example 3. This program demonstrates what happens when a BASIC keyword is nested in a UDF name. During the program run, line 320 tells the computer to define a function named *SU*. Then, before the argument and the formula are given, the line tells the computer to perform a TO. The computer does not understand this instruction, and lets you know with a syntax error message. Try changing the *O* in SUBTOTAL to an *A* (SUBTATAL) in line 320 only, and run it. You will still have a syntax error, but now located in line 330. Do the same in line 330 and the program should run with no problems. You should now get:**

```
5
READY.
```

In my opinion, long function or variable names have no socially redeeming value other than to keep aspirin manufacturers busy. Avoid them! If you need documentation, use REM statements (REMarks). Don't forget that some two-character names can get you into trouble also (IF, GO, TO, ON, FN).

The name of the UDF serves only to identify the function. The name does not relate to or affect any variables in any manner. Thus, a program may contain UDFs and variables with the same name, and such variables may be used in the UDF. The function will operate normally because there is no interaction between UDF names and variable names. Also, the % (integer) sign and the $ (string) sign must not appear in the UDF name, or an error message (SYNTAX for %, TYPE MISMATCH for $) will result. A UDF is an arithmetic function, not a variable. So, integer and string signs have no meaning and are forbidden in the UDF's name. The best advice on UDF and variable names is: keep it simple. You can't go wrong with a single letter or a letter followed by a number (A3, X9, etc.) for a name.

### Rules For Names

1. A UDF must have a name.
2. Both the definition statement and the execution statement must have the same name.
3. UDF names follow the same rules as variable names.
4. BASIC keywords, %, and $ must not appear in the name.
5. Names for different UDFs must not begin with the same first two characters.
6. UDF names do not relate to or affect any variable.
7. (Recommendation) Limit the names to one or two characters.

*Arguments:* The arguments for UDFs behave somewhat differently from the other BASIC functions. The *CBM User Manual* and the *PET/CBM Personal Computer Guide* call the arguments dummy variables or dummy assignments. They are not that simple! The arguments are extremely useful, and contribute a great deal to the power and flexi-

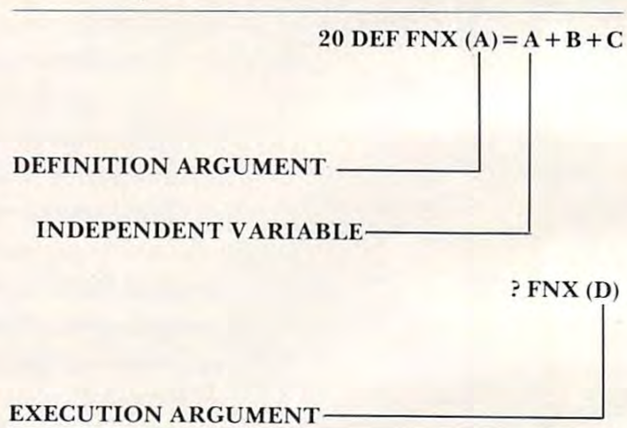



**Figure 2. The UDF's arguments. The definition argument selects a variable in the formula to be the independent variable. The value of the independent variable is determined by the execution argument. The independent variable is a separate entity from program variable of the same name. In the above function, the value of *D* will be plugged into the independent variable during the execution of the function. Thus, the equivalent formula for the above function is FNX = D + B + C.**

bility of the UDF. Let's see how they operate.

Both the definition statement and the execution statement must have an argument contained in the parenthesis of each statement. Each statement must contain only one argument. For simplicity, let's refer to the definition statement's argument as the *definition argument* (my jargon), and the execution statement's argument as the *execution argument* (my jargon). See Figure 2.

The definition argument selects *one* variable from the UDF's formula to be the *independent variable* (my jargon). The independent variable is no longer a program variable and can exist only within the formula of the UDF. It is a new entity and, as such, it neither affects nor is affected by a program variable with the same name. Its operation is independent of the program variable. Another way of looking at it is that the definition argument selects a location within the formula for the independent variable. The location is marked by the variable name in the definition argument. The independent variable may appear in the formula as often as required, but there can be only one variable name in the definition argument.

The execution argument provides a value to be "plugged," or substituted, into the independent variable during the function execution. See Example 4. The other variables of the formula (those not specified in the definition argument) will operate with their current values as assigned in the program. So, the definition argument selects an independent variable, and the execution argument provides a value for it.

**LISTING**

```
410 A=2:B=3:C=7
420 DEF FNX (A)=A+B
430 PRINT FNX (C)
440 PRINT A
```

**COMPUTER'S ACTIONS**

**410 A=2:B=3:C=7**

**420 Define a function named *X*.**
**FNX=I.V.+B**

**430 Execute the function named *X*.**
**FNX=C+B**
**FNX=7+3**
**FNX=10 Print on screen.**

**440 A=2 Print 2 on screen.**

**RESULT – COMPUTER DISPLAYS:**

```
10
2
READY.
```

**Example 4. This program demonstrates the operation of the UDF's arguments. The argument in line 420 selects the variable for the independent variable. The definition argument actually selects a *location* in the formula for the independent variable. The location is indicated by underscoring in the comments to the right (I.V.). The execution argument in line 430 provides the value for the independent variable. Notice that *B*, *C*, and even *A* (line 440) do not change values.**

**LISTING**

```
510 A=2:B=3:C=7:D%=9:E=-25
520 DEF FNX (A)=A+B+C
530 PRINT FNX (A)
540 PRINT FNX (D%)
550 PRINT FNX (E)
560 PRINT FNX (K)
570 PRINT FNX (150)
580 PRINT FNX (SQR(D%↑3) )
```

**COMPUTER'S ACTIONS**

**510 A=2:B=3:C=7:D%=9:E=-25**

**520 Define a function named *X*.**
**FNX=I.V.+B+C**

**530 Execute the function named *X*.**
**FNX=A+B+C**
**FNX=2+3+7**
**FNX=12 Print on screen.**

**540 Execute the function named *X*.**
**FNX=D%+B+C**
**FNX=9+3+7**
**FNX=19 Print on screen.**

**550 Execute the function named *X*.**
**FNX=E+B+C**
**FNX=-25+3+7**
**FNX=-15 Print on screen.**

**560 Execute the function named *X*.**
**FNX=K+B+C**
**FNX=0+3+7**
**FNX=10 Print on screen.**

**570 Execute the function named *X*.**
**FNX=150+B+C**
**FNX=150+3+7**
**FNX=160 Print on screen.**

**580 Execute the function named *X*.**
**FNX=SQR(D%↑3)+B+C**
**FNX= 27 +3+7**
**FNX=37 Print on screen.**

**RESULT – COMPUTER DISPLAYS:**

```
12
19
-15
10
160
37
READY.
```

**Example 5. This program shows a variety of execution arguments in action. In line 560, *K* is an unassigned program variable. Therefore, *K*=0.**

The definition argument must specify *one* floating point variable only. String ($), integer (%), and subscripted (from arrays) variables must not appear in the definition argument. The same is true for BASIC keywords, numerical values, and arithmetic expressions. The execution argument may use floating point, integer, and subscripted variables. However, string variables must not be used in the execution argument. Direct numerical values, arithmetic expressions, or a program variable with the same name as the independent variable may also be used in the execution argument. The value of such items will be substituted in the same manner as the variables.

A UDF may be executed as often as desired in the program, and each time a different execution argument can be used. Thus, a variety of values can be plugged into the function's independent variable by changing nothing other than the execution argument. If an unassigned variable is entered as an execution argument, zero will be substituted into the independent variable. Any variable equals zero until the program LETS, READS, INPUTS, or otherwise assigns a value to it. Example 5 shows the various possibilities for execution arguments.

If it is necessary to change the assignment of the independent variable to another variable within the UDF's formula, then the UDF must be redefined using the desired variable as a definition argument. (See Example 6.) If the original UDF is also required, then a new UDF, with a different name, should be defined for the change. Also, it is possible not to have an independent variable. If the definition argument contains a variable which is not used in the formula, there will be no independent variable in the formula. The function will operate on the current values of the variables found in the formula. No substitutions will take

**LISTING**

```
610 A=2:B=24:C=8
620 DEF FNX (A)=B/A
630 PRINT FNX (C)
640 PRINT FNX (12)
650 DEF FNX (B)=B/A
660 PRINT FNX (C)
670 PRINT FNX (12)
```

**COMPUTER'S ACTIONS**

```
610 A=2:B=24:C=8
620 Define a function named X.
    FNX=B/I.V.
630 Execute the function named X.
    FNX=B/C
    FNX=24/8
    FNX=3 Print on screen.
640 Execute the function named X.
    FNX=B/12
    FNX=24/12
    FNX=2 Print on screen.
650 Define a function named X.
    FNX=I.V./A
660 Execute the function named X.
    FNX=C/A
    FNX=8/2
    FNX=4 Print on screen.
670 Execute the function named X.
    FNX=12/A
    FNX=12/2
    FNX=6 Print on screen.
```

**RESULT – COMPUTER DISPLAYS:**

```
3
2
4
6
READY.
```

**Example 6. This program demonstrates the reassignment of the independent variable to another variable in the formula. Note that the only item changed is the definition argument (line 650). Yet, look at the difference in results.**

**LISTING**

```
710 A=2:B=3:C=7
720 DEF FNX (Z)=A+B+C
730 PRINT FNX (C)
740 PRINT FNX (100)
```

**COMPUTER'S ACTIONS**

```
710 A=2:B=3:C=7
720 Define a function named X.
    FNX=A+B+C
730 Execute the function named X.
    FNX=A+B+C
    FNX=2+3+7
    FNX=12 Print on screen.
740 Execute the function named X.
    FNX=A+B+C
    FNX=2+3+7
    FNX=12 Print on screen.
```

**RESULT – COMPUTER DISPLAYS:**

```
12
12
READY.
```

**Example 7. This program demonstrates a UDF without an independent variable. The definition argument, *Z*, is not found in the formula. Therefore, there is no independent variable, and the function does not use the execution argument. The same result will occur, regardless of value of the execution argument. Note: both arguments must still be included in their respective statements, or a syntax error will result.**

place in any variable. The execution argument, although still required, will have no effect on the function, and hence, is a dummy assignment. (See Example 7.)

Examples 4, 5, 6, and 7 demonstrate the powerful effects that the arguments have on the UDF. Note the differences in the results of the examples caused by manipulating the arguments. Obviously, if you are to effectively use UDFs, you must have a clear understanding of the operation of the arguments. Try devising some UDFs of your own on paper; see if you can predict the results for specific arguments, and then try them on your computer.

### Rules For Arguments

**Definition Argument**

1. There must be one, and only one, argument in the definition statement.

2. The argument must be contained in parentheses.

3. The argument must be a floating point variable only; no %, $, or subscripted variables and no numbers, BASIC keywords, or expressions are allowed in the definition argument.

4. The definition argument selects *one* variable from the formula to be the independent variable.

5. To change the independent variable selection within the formula, the UDF must be redefined.

6. If the argument does not appear in the formula, there will be no independent variable.

**Execution Argument**

1. There must be one, and only one, argument in the execution statement.

2. The argument must be contained in parentheses.

3. The argument can be a floating point, integer, or subscripted variable. Numbers and arithmetic expressions are allowed. No string variables.

4. The value of the execution argument is substituted into the independent variable.

5. A new argument may be used each time the function is executed.

6. If the argument is an unassigned program variable, 0 will be substituted into the independent variable.

7. If the formula does not contain an independent variable, the execution argument, though still required, will have no effect on the formula.

*Formulas:* What can we use in the formula? A general rule: if it has something to do with math, it can appear in the UDF's formula. Let's see what we can use.

*Variables:* floating point, integer (%), and subscripted variables may be used in the formula. String ($) variables must not appear in the formula.

*Numbers:* any numerical value within the normal range of the computer can be used. Also the symbol π (pi) may be used.

*BASIC Commands:* (CLR, LIST, LOAD, ... etc.) may not appear in the formula.

*BASIC Statements:* (DEF FN, DIM, FOR/NEXT, ... etc.) may not appear in the formula.

*String Functions:* (ASC, CHR$, LEFT$, ... etc.) and String Concatenation (+) may not be used in the formula.

*Arithmetic Functions:* (ABS, ATN, COS, ... etc.) may be used in the formula.

*Arithmetic Operators:* (+, -, *, /, ↑, - negation) can be used.

*Boolean Operators:* (AND, OR, NOT) can be used in bit-oriented operations.

*Relational Operators:* (=, <, >, <=, >=, <>) can all be used in the formula.

*Exceptions:* There are some non-arithmetic functions that will work in the formula. All of these functions are of a "return a value" nature. The following is a list of these exceptions: ASC, LEN, VAL, FRE, PEEK, POS, ST, TI, USR. I suggest using these functions with caution. Thoroughly experiment with the functions before including them in a program.

A UDF can have only one formula. A variable, number, operator, or function can be used as often as needed within the formula, but the entire definition statement must fit on one 80-character line. If there is a bug (error) in the formula, it will not show up until the function is executed. The appropriate error message will be displayed, but it will be referenced to the line of the execution statement, not the definition statement. Thus, the following program would result in a ?DIVISION BY ZERO ERROR IN 30. Note the error is referenced to line 30, but the correction will have to be made in line 20.

```
10 A=2:B=3:C=7
20 DEF FNX (A)=A+(B/0)
30 PRINT FNX (C)
```

Keep this in mind; it will make your debugging effort much easier. Don't forget about the order of evaluation (multiplication is performed before addition, etc.) and the other rules applicable to formulas. UDF formulas behave just like any other formulas with the exception of the independent variable.

**Rules For Formulas**

1. A UDF can have only one formula.

2. The formula and the other elements of the definition statement must fit on one 80-character line.

3. Any variable except strings ($) may be used.

4. Any number and $\pi$ may be used.

5. Any arithmetic function or operator may be used except DEF FN.

6. Non-arithmetic functions may not be used. (See text for exceptions.)

7. Relationals and bit-oriented Boolean functions may be used.

8. A bug in the formula will show up in the execution statement line, not the definition.

9. To modify the structure of the formula, the UDF must be redefined.

10. Any variable or function may appear as often as needed in the formula.

## How Are UDFs Used?

UDFs are used in the same way that any other BASIC function in the program is. The first thing to realize is that no action takes place during the definition statement other than the function being stored in memory. The definition statement performs nothing; the function is performed in the execution statement. This is why errors in the formula show up at the first execution statement, rather than the definition statement. Don't try to use the function with a definition statement (e.g., J = DEF FNX (A) = A + B + C); it will not work. The UDF is always used, that is, *performed* (and the result used), with an execution statement. Also, bear in mind that in order to execute a UDF, you must first define it. Failure to do so results in an ?UNDEF'D FUNCTION ERROR. Watch out for GOSUBs, IF/THENs, GOTOs, and anything else that may block the definition statement.

You must do something with the execution statement. Try this:

```
10 A=2:B=3:C=5
20 DEF FNX (A)=A+B+C
30 PRINT FNX (A)
40 FNX (A)
```

The above program will give you a 10 (from line 30) followed by ?SYNTAX ERROR IN 40. Why? Line 30 proves that the UDF is OK. So, why the syntax error? The PET, like humans, resents pointless work. In the orderly world of machine logic, everything is done for a reason. Computers do not entertain idle thoughts, which is precisely what line 40 is: an idle thought. It does nothing with the result of the function, and the computer does not understand what it is supposed to do. Imagine a stranger approaching you on the street; his only words are "FNX (A)", and he then impatiently waits for your reply. You would probably deliver a syntax error message of greater magnitude than the computer's. Quite simply, you have to do something with the result of the UDF execution statement. What can you do with it? You can do anything except treat it as a variable and try to assign a value to it (e.g., let FNX (A) = 3, READ FNS (A), INPUT FNX (A) are all no-no's). Remember, the execution statement provides a result, or an output; it cannot be used as an input. Figure 3 shows some of the ways you can use a UDF in a program.

| USE IN PROGRAM | COMMENTS |
|---|---|
| DEF FNX (A) = A + B | Definition statement, can be used only to define a function. |
| PRINT FNX (3) | Print result on screen. |
| J = FNX (S) | Sets variable *J* equal to result. |
| IF FNX (D%) = Y THEN 500 | Testing result in an IF/THEN. |
| Z5 = FNX (F) AND 64 | ANDing result with 64. |
| FOR B = 1 TO FNX (Q) | Limit on a FOR/NEXT statement. |
| R2 = SQR(FNX(Q) ) | Argument for another arithmetic function. |
| ON FNX(P1) GOSUB 200,300, | Index for an ON GOSUB. |
| POKE32767 + FNX(T),99 | Address calculation for a screen POKE. |
| PRINT VTAB(FNX(G) )U | Argument for a TAB. |
| DEF FNY (D) = FNX (N) + D + E | Definition of another UDF. See nested UDFs in text. |

**Figure 3. (This is not a program.) Shown above are some of the ways that UDFs may be implemented in a program. The definition statement cannot perform the function; the execution statement must be used to implement the UDF. Some action must occur with the execution statement. And the execution statement must always act as an output of data. That is, the UDF cannot be a receiver of data like a variable, so you cannot LET FNX, INPUT FNX, READ FNX, or GET FNX.**

*Nested UDFs:* The execution statements of one or more *previously defined* UDFs may appear in the formula for a new UDF. See the last line of Figure 3. Why next UDFs? It is a very effective way to create formulas that exceed the 80-character limitation on a definition statement. Nesting allows very long formulas to be used in the program. Also, nesting is the programming convenience. If you have two formulas, the second containing the first (e.g., $A+B+C$ and $A+B+C+D+E$), the definition of the second UDF can contain the execution of the first UDF. This saves re-entry of the same formula. See Example 8.

**LISTING**

```
810 A=2:B=3:C=5:D=7:E=9:F=10:G=20
820 DEF FNX1 (A)=A+B+C
830 PRINT FNX1 (F)
840 DEF FNX2 (D)=FNX1 (F)+D+E
850 PRINT FNX2 (G)
860 DEF FNX3 (D)=FNX1 (D)+D+E
870 PRINT FNX3 (G)
```

**COMPUTER'S ACTIONS**

```
810 A=2:B=3:C=5:D=7:E=9:F=10:G=20
820 Define a function named X1.
    FNX1=I.V.+B+C
830 Execute the function named X1.
    FNX1=F+B+C
    FNX1=10+3+5
    FNX1=18 Print on screen.
840 Define a function named X2.
    FNX2=FNX1 (F)+I.V.+E
850 Execute the function named X2.
    FNX2=FNX1 (F)+G+E
    Execute the function named X1.
    FNX1=F+B+C
    FNX1=10+3+5
    FNX1=18 Send to FNX2.
    FNX2=18+G+E
    FNX2=18+20+9
    FNX2=47 Print on screen.
860 Define a function named X3.
    FNX3=FNX1 (I.V.)+I.V.+E
870 Execute the function named X3.
    FNX3=FNX1 (G)+G+E
    Execute the function named X1.
    FNX1=G+B+C
    FNX1=20+3+5
    FNX1=28 Send to FNX3.
    FNX3=28+G+E
    FNX3=28+20+9
    FNX3=57 Print on screen.
```

**RESULT – COMPUTER DISPLAYS:**

```
18
47
57
READY.
```

**Example 8. Nested UDFs. Note in line 860 that the execution argument of FNX1 is an independent variable for FNX3. Compare the execution of FNX3 with FNX2.**

UDF nesting is not without problems. If too many UDFs are nested, an ?OUT OF MEMORY ERROR will result. Try Example 9. Notice how many free bytes are left. How can you be out of memory? As each function is executed, data is entered on the *stack*.

The stack is a section of memory containing 256 consecutive locations. The processor uses the stack to store addresses and data when it is called to perform another function before completing the current operation. When the interrupting operation is completed, the processor removes the stored information from the stack and continues where it left off. If the processor is interrupted while working on an interrupt, then interrupted while working on that interrupt, etc., at some point the stack will be filled and the program will terminate with an ?OUT OF MEMORY ERROR.

When the stack is full, an ?OUT OF MEMORY ERROR results regardless of how much RAM is available. The program can be further compounded by GOSUBs, FOR/NEXTs, and any other operation

```
910 A=2:B=3:C=7
920 DEF FNXA (A)=A+B
925 DEF FNXB (A)=FNXA (A)+C
930 DEF FNXC (A)=FNXB (A)+C
935 DEF FNXD (A)=FNXC (A)+C
940 DEF FNXE (A)=FNXD (A)+C
945 DEF FNXF (A)=FNXE (A)+C
950 DEF FNXG (A)=FNXF (A)+C
955 DEF FNXH (A)=FNXG (A)+C
960 DEF FNXI (A)=FNXH (A)+C
965 DEF FNXJ (A)=FNXI (A)+C
970 DEF FNXK (A)=FNXJ (A)+C
975 PRINT FNXJ (A)
980 PRINT "FREE BYTES="FRE(0)
985 PRINT FNXK (A)
```

**RESULT – COMPUTER DISPLAYS:**

```
68
FREE BYTES=31318          Exact value will vary.
?OUT OF MEMORY ERROR IN 985
READY.
```

**Example 9. This program demonstrates consumption of the stack by excessive nesting of UDFs. The *68* results from line 975, thus executing FNXJ did not fill the stack. Line 980 proves that there is plenty of RAM left. (The exact value depends on memory size, and if there is other programming stored.) Executing FNXK in line 985 fills the stack, and the program terminates with the ?OUT OF MEMORY ERROR message.**

that uses the stack. If this happens, you have no alternative other than to reduce the complexity of the nested UDFs. You can always set a variable equal to a lower function, and then use it to pass the data to the higher UDFs (for Example 9, change line 970 to: 970 Q=FNXJ (A):DEF FNXK (A)=Q+C).

When nesting, a second UDF cannot be defined with the definition statement of the first (e.g., 10 DEF FNX2 (A)=DEF FNX1 (A)=A+B+C). An error will result when either function is executed. When a UDF is nested within a second UDF's formula, the first UDF must be defined before the second UDF is executed. That is, the computer must see the definition statement of the first UDF (as well as the second) before the execution statement of the second UDF can be performed. Failure to do so will result in an ?UNDEF'D FUNCTION ERROR when the second UDF is executed. Also, a UDF cannot redefine itself by nesting the original execution statement in the formula for the new UDF (e.g., 100 DEF FN*X1* (A)=FN*X1* (A)+B+C+D). In other words, the UDF in the formula must have a different name than the UDF that is being defined, or an ?OUT OF MEMORY ERROR will result.

Example 8 shows that the execution argument of the UDF in the formula can be an independent variable of the UDF being defined. This can lead to some powerful and interesting possibilities, and it can also lead to some real debugging problems if you are not careful. If you do not want other values substituted into the nested UDF, be sure that the execution argument is not an independent variable of the UDF being defined. The UDF FNX2 (in Example 8) has FNX1 nested in its formula. The execution argument of FNX1, *F*, is not an independent variable of FNX2.

When FNX2 is executed (line 850), FNX1 will still operate with *F* as the execution argument. FNX3 is the same as FNX2 except that the execution argument of FNX1 in the formula has been changed to *D*, the independent variable of FNX3. Compare the execution of FNX3 (line 870) with the execution of FNX2 (line 850). Quite a difference! Imagine the mess that could evolve out of nesting four or five complicated UDFs. You must be careful when selecting arguments for nested UDFs.

### UDFs In Immediate Mode

The PET is a very poweful calculator when used in the direct mode. Unfortunately, typing in a few dozen SQR, SIN, EXP, etc, can get to be a real drag. You may be able to use a UDF to eliminate some of that typing. Let's see how.

Let's assume that we have a series of calculations to perform. We don't have time to fool around with a program and, besides, the calculations vary, so a program is not practical. Throughout the series of calculations appears the equation:

$$C=\sqrt{A^2+B^2}$$

The good, old Pythagorean theorem. In BASIC the equation appears as:

**C=SQR(A↑2+B↑2)**

Now who in their right mind wants to type that in a dozen times or so? Let's create a UDF to calculate the equation. We cannot define a UDF in the direct mode; we would get an ?ILLEGAL DIRECT ERROR. No problem. We will define the function with a one line program:

**10 DEF FNC (A)=SQR(A↑2+B↑2)**

Now we must RUN the one line program. Nothing appears to happen, but an action did occur: the UDF was stored in the memory. OK, we can now use our UDF in the direct mode.

| | | |
|---|---|---|
| **Type in:** | **B=4:?FNC(3)** | **Press RETURN.** |
| **Result:** | **5 is displayed.** | |
| **Type in:** | **B=557:?FNC(332)** | **Press RETURN.** |
| **Result:** | **648.438895 is displayed.** | |

Anytime we need to use the above equation in our calculations, all we have to do is assign a value to *B* and execute the UDF. The value for *A* is entered in the execution argument.

Do you need cube roots in direct mode? If so, try this:

**20 DEF FNA (X)=EXP(LOG(X)/3)**

RUN the program, and try some direct mode cube roots.

| | | |
|---|---|---|
| **Type in:** | **?FNA(27)** | **Press RETURN.** |
| **Result:** | **3 is displayed.** | |

Try the same thing with 343; you should get seven. (The question mark, ?, is an abbreviated PRINT command.) If you replace the three in the above equation with a variable, say *Y*, you can use it to find any root. FNA will return the "Yth" root of *X*; that is, FNA will equal $\sqrt[Y]{X}$

The examples I have given are rather simple, but they demonstrate the idea. You can have definition statements up to 80 characters, and you can nest UDFs for direct mode as well as in the program mode. Thus, UDFs can transform your PET into a super-calculator for direct mode calculations. Just remember to define the UDFs with line numbers, and RUN the program before attempting to execute the UDFs. You should have to RUN the definition program only once, unless there is a power failure.

Where do you go from here? *Experiment!* Use UDFs in your programs. Your formulas don't have to be *Einsteinian* to qualify for UDFs. If you run into something that I didn't cover, or find a difference in operation with other ROMs, or, if you have a unique application, send it in to **COMPUTE!**.

If you do a lot of work with trig, you may find pages 62 ad 63 of the *CBM User Manual* (see references) of interest. Commodore lists about 20 different UDFs involved with trig. If you like games, I have one more technique for you. Games use a lot of random numbers, quite often with a variety of ranges within the same program. Here is a UDF that will give you a random integer in a range of 1 to *X*, where *X* is the maximum desired number.

```
10 DEF FNR (X)=INT(X*RND(1)+1)
```

To use the function in a program, simply execute the UDF with the maximum desired number placed in the execution argument. For example: PRINT FNR (500) will print a random integer between one and 500. In the same program you may have:

```
POKE32767+FNR (1000),42.
```

This would display an asterisk in a random location on the screen. You can use the function as often as required, and the range is a simple matter of choosing an execution argument.

**References**

*Butterfield, Jim. "Commodore ROM Systems: Terminology."* **COMPUTE!**, *July 1981, #14, pp. 146-148.*

*Donahue, Carroll S., and Janice K. Enger.* PET/CBM Personal Computer Guide. *First Edition. Berkeley: Osborne/McGraw-Hill, 1980.*

CBM User Manual. *First Edition. P/N 320856-3. Commodore Business Machines. Santa Clara, CA. 1979.*

©

*In this in-depth profile of* **COMPUTE!** *Associate Editor and Columnist Jim Butterfield, he discusses the future of computing, his background and hobbies, and his views on topics ranging from computers in education to the appeal of programming.*

# Meet Jim Butterfield

Gail Hook
Barris, Ontario

The Butterfield homestead is a modest brick house within walking distance of downtown Toronto. It is comfortably cluttered with books, plants, computers, and three cats. Even the attic is pressed into service as storage space for whatever books and computers Jim Butterfield cannot cram into his small office.

The office, in fact, resembles a crowded depot for a changing assortment of computers – including four Commodore PETs of varying screen sizes and ages, a VIC-20, an Atari 800, a KIM, a Rockwell AIM, and an Alpha, a European machine. Stacked next to the computers is a "disk tower" consisting of two Commodore double disk drives (a 4040 and an 8050), an Atari 810 drive, and an ancient Commodore 2023 printer perched on top. Bookshelves along one wall are overflowing, and every available inch of floor space is carpeted with piles of diskettes, papers, and still more books. Yet, amazingly, Butterfield always seems to know into which pile to dive for what he needs.

One of the three cats, the Siamese, possesses a similar instinct. With a feline knack for homing in on the center of warmth and attention, she often dozes atop whichever PET is on and humming.

The main occupant of the office – Butterfield – meshes with the environment, too. He speaks with a gravelly voice in the measured phrases of someone used to teaching or being quoted for publication. Middle-aged and greying, he brings to microcomputing an almost childlike curiosity and sense of delight, a fascination which led him first to an absorbing hobby and finally, in early 1981, to a new career as a freelance writer, consultant, and teacher. Today he is recognized as a premier expert on Commodore computers, as a prolific writer, and perhaps most of all as an unusually coherent voice in the seemingly impenetrable technical thicket of personal computing.

### A Change of Careers

Like most career changes, the switch surprised Butterfield as much as anybody. For 24 1/2 years he worked for Canadian National/Canadian Pacific Telecommunications. He quit solely because the company decided to move far away from central Toronto, and he would have spent so much time commuting there would have been none left for his hobby. For Butterfield, it was no contest. "When faced with that choice, I really had no choice and I quit."

Actually, it was while working for CN/CP in 1964 that Butterfield was first introduced to computers – although personal microcomputers were still undreamt-of in those days. Butterfield spent a year as a programmer of a rather specialized computer, a Collins C8401. FORTRAN and COBOL were coming into use at the time, but the Collins didn't use any such advanced languages. Programmers had to do almost everything in machine language. Butterfield soon moved into other areas of the company, but a little more than ten years later his interest was rekindled by a new invention – microcomputers.

"I decided to find out what this 'micro' stuff was all about and started watching the current magazines," he says. "I finally decided to purchase when I saw a completely pre-built machine called a KIM-1, which had a 6502 microchip in it. That turned out to be like a return to the past. Everything we had been doing a dozen years before on the large $1.5 million computer, we were doing again on this little $250 board – including making the same mistakes."

### KIM And The Start Of Social Computing

One machine led to another, and Butterfield began sharing his knowledge with other microcomputer users, as well as writing about his discoveries. He had gained some writing experience many years before in western Canada, where he was born, as a "continuity writer" for a couple of radio stations. (Butterfield smiles, "That means I spent about a year of my life writing commercials.")

As the users of early microcomputers began comparing notes, it wasn't long before a cult of sorts sprang up. Indeed, the emergence of microcomputers as a basis of social, and not merely technical, interaction is the facet of the field that Butterfield enjoys most. In the earliest days of "roll-your-own-computers," he notes, everyone had a different machine, which crimped the sharing of information. "Suddenly, along came the KIM. Everybody had the same computer. An amazing thing happened – and this is multiplied many times over in the Commodore line – people built a social life around microcomputers."

The thriving Toronto PET Users Group (TPUG) is a case in point. Butterfield had what he calls a "Machiavellian influence" on TPUG founder Lyman Duggan, whom Butterfield persuaded to hold the first meeting in his basement one summer evening. While Butterfield firmly rejects any organizing chores, he contributes a great deal as a friend of the club, speaking at monthly meetings

**"An amazing thing happened ... people built a social life around microcomputers."**

and sharing his expertise.

Butterfield admits, "It's getting harder to know what to talk about at those meetings. There are a number of people who have the ability to track down any part of the machine they want to go after, and who are quite skilled at machine language. As a result, my sympathy is with the beginner. I'd rather bore ten experts than lose the bulk of people, so I try to keep things fairly simple."

Butterfield's sympathy for beginners is well known and shows in his articles. His writing is informal and witty in spite of its technical content. "I try to write it as I would say it. I do a lot of presenting material to both kids and adults, and I try to keep the same style in my writing. Also, whenever I can, I slip in a simple example program. Then, even if the readers can't understand what I mean, they can run the programs."

### Light Consulting

Butterfield also indulges in what he calls "light consulting," principally for Commodore. In the spring he went on a western Canadian promotional tour for the VIC-20 computer. He's also frequently invited to shows, such as the PET Show in London he attended in June. He finds this part of his work "really great fun" because it provides opportunities for travel.

Lecturing and teaching, such as the machine language course he conducts each month for a special interest division of TPUG, provide him with feedback about problems and areas where people need more information. He has a reputation for being generous with his time, and his phone is open from 10 a.m. to 10 p.m. Monday to Friday. "If somebody phones me up and asks a question which shows they just haven't bothered trying it themselves, then I will sometimes be a little short, because it does seem like a waste of my time," he says. "But most people who call do so because they're stuck on something. It's just a question of getting another opinion. If I get a number of inquiries in a certain area, that's usually a signal that it's time for me to write an article about it. It's a very good way of keeping posted on what's bothering people at the moment."

Butterfield is equally generous with his software. He rarely sells any of his programs. "I would like to foster an environment where people pass out their software with reasonable generosity. I think that by showing a good example, I might sort of lead the way in that." Often he distributes his work on TPUG's library disk.

Still, Butterfield vehemently supports an author's copyright: "I believe very strongly that the person writing an original program has the right to do as he chooses with that program. If he chooses to sell it or to request that it not be copied except for a fee, then he has absolutely that right."

However, he feels that a person who takes money for software is obligated to support that program by upgrading it and furnishing the means to modify it, if necessary. "That's another good reason to give programs away. I really feel that most people who put down a lot of money for software feel that they are not buying a disk or cassette tape, but they are buying a service."

Interestingly, Butterfield believes the problem of software piracy might lessen, not grow, with the increasing business use of microcomputers. He laughs, "If an employee ran to the boss and said, 'Chief, I think you should give me a raise because I just saved you $500, I lifted a copy of a program,' I really don't think very many businesses would stick a cigar in my mouth and give me a promotion. They would more likely start keeping an eye on me."

Butterfield thinks that renting software eventually may be the best way to distribute it. A yearly fee could be charged for its use. In return, the user would receive continuing support on such things as upgrades, newsletters, information, warranty, and documentation.

### Something Unprecedented In Education

Given his multiple interests in computing, writing, teaching, and making life easier for beginners, it's only natural that Butterfield is a strong advocate of introducing children to computers early in school. "As I understand the writings of Seymour Papert [author of *Mindstorms: Children, Computers, and Powerful Ideas*], the earlier a child becomes exposed to computers, the better it is likely to be," he says. "I have seen no evidence to contradict this. It seems to me that more important than anything formalized we teach young people about computers is

# SOFTBOX® unlocks the world of proven business software to COMMODORE users.


Simply by plugging the Small Systems SOFTBOX into the IEEE-488 port and loading the CP/M disk, your Commodore PET/CBM will run under the world's most popular disk operating system. No internal connections or modifications are required.

A complete computer system in itself, the SOFTBOX uses the PET or CBM as its terminal while talking to any combination of Commodore and Corvus drives. A single command returns you to PET BASIC. The SOFTBOX utility disk gives you complete software control over your configuration.

**Terminal Emulation**

Applications packages designed to work with specific terminals (like ADM3A, TVI 912 or Hazeltine 1500) will need no modification for the 80-column CBM, since the SOFTBOX allows your computer to emulate any of these devices.

**Add Corvus Hard Disk**

Expand your CP/M file storage capacity with reliable Corvus drives. With the Corvus Mirror Option and a standard video cassette recorder, you have a fast, inexpensive disk backup. When you purchase the SOFTBOX and Corvus from Small Systems, you'll receive everything necessary to convert your Commodore to a hard disk-based CP/M system.

**Features**

- CP/M version 2.2 with menu driven system configuration.
- Z80 CPU running a 4MHz with no wait states.
- 64K RAM.
- Operates with any 2000, 3000, 4000 and 8000 series Commodore computer.
- Supports up to 8 Commodore 3040, 4040, or 8050 disk drives.
- Supports up to 4 Corvus hard disks of 6, 11, or 20 MBytes.
- RS232 serial interface with software-definable baud rates for use with a printer, modem or terminal.
- Diskette containing operating system with utilities.
- Comprehensive user manual.

A full range of CP/M languages and applications software on Commodore 8050 or 4040 format is available.

*CP/M is a registered trademark of Digital Research Corp.*

For more information on how you can use the world of proven business software on your PET and CBM, contact your local Commodore dealer. Or, for more information and the name of the dealer nearest you, call or write us today.

SSE (415) 964-8201
SMALL SYSTEMS ENGINEERING
222-B View Street • Mountain View, CA 94041

computer
CBM™
Model 8032
commodore

# Tomorrow's Technology Today

## Commodore Computer

| | |
|---|---|
| VIC 20 Personal Computer | $ 299.95 |
| VIC 1515 Graphic Printer | $ 395.00 |
| CBM 4016 | $ 995.00 |
| CBM 4032 | $1295.00 |
| CBM 8032 | $1495.00 |
| CBM SuperPet 9000 | $1995.00 |
| CBM 8050 Dual Disk Drive | $1795.00 |
| CBM 4040 Dual Disk Drive | $1295.00 |
| CBM 2031 Single Disk Drive | $ 695.00 |
| CBM 4022 Tractor Printer | $ 795.00 |
| CBM 8010 IEEE Modem | $ 279.00 |
| CBM Datasette | $ 74.95 |
| CBM-IEEE Interface Cable | $ 39.95 |
| IEEE-IEEE Interface Cable | $ 49.95 |

## NEC Spinwriter Printer

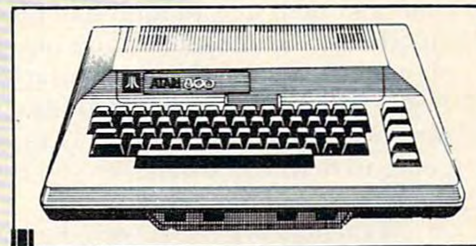
| | |
|---|---|
| NEC 7730 | $3085.00 |
| NEC 7710 | $3085.00 |
| NEC 7720 | $3610.00 |
| NEC 3510 | $2290.00 |
| NEC 3500RD | $1895.00 |

Tractor Feed Options are available

## Professional Software

WordPro™ — A Family of CBM Word Processing Programs

| | |
|---|---|
| WordPro 2 Plus | $ 199.95 |
| WordPro 3 Plus | $ 295.00 |
| WordPro 4 Plus | $ 450.00 |
| WordPro 5 Plus (for CBM 8096) | $ 450.00 |
| The Administrator (DataBase for CBM) | $ 650.00 |
| POWER™ (Programmer's Utility ROM) | $ 89.95 |
| InfoPro™ (DataBase for CBM) | $ 295.00 |



## Atari

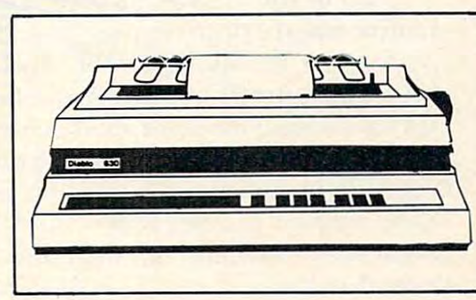
| | |
|---|---|
| Atari 400 16K | $ 399.00 |
| Atari 800 16K (incl. BASIC cartridge) | $ 899.00 |
| Atari 410 Recorder | $ 99.95 |
| Atari 810 Disk Drive | $ 599.95 |
| Atari 822 Thermal Printer | $ 299.95 |

## Amdek Monitors

| | |
|---|---|
| Video 100G (Limited Quantity) | $ 179.00 |
| Video 300G | $ 200.00 |
| Color 1 | $ 449.00 |
| Color 2 | $ 999.00 |

## Diablo 630 Printer

| | |
|---|---|
| Diablo 630 | $2710.00 |
| Tractor Option | $ 350.00 |



## Epson Printers

| | |
|---|---|
| MX-70 | $ 299.95 |
| MX-80 w/graphtrax | $ 645.00 |
| MX-80 FT | $ 745.00 |
| MX-100 FT | $ 945.00 |
| INTERFACE CARDS | |
| 8141 RS-232 Interface Board | $ 75.00 |
| 8145 RS-232 Interface Board w/2K Buffer | $ 149.00 |
| 8151 RS-232 Interface Board w/X/ON-X/OFF | $ 170.00 |
| 8161 IEEE Interface Board | $ 55.00 |
| 8131 Apple Interface Card | $ 85.00 |
| 8232 Apple Interface Cable | $ 35.00 |
| 8220 TRS-80 Cable | $ 35.00 |

## CMD Mupet

| | |
|---|---|
| MC-800A Mupet Controller (Multi-User Controller for CBM Computers) | $ 995.00 |
| CM-100 Channel Module | $ 250.00 |
| Printer Module | $ 350.00 |

## Qume

| | |
|---|---|
| Spring 9/45 | $2495.00 |
| Tractor Option | $ 210.00 |

WordPro, POWER and InfoPro are registered trademarks of Professional Software Inc.

**NEECO**
679 Highland Ave.
Needham, MA 02194

(617) 449-1760
TELEX: 951021

MON-FRI 9:00 - 5:30 E.S.T.
MASTERCARD *VISA

that we get them familiar with the concept, we get their fears allayed, we make sure that the usefulness of computers is understood at an early age. By the time a student gets to high school, computers are an oddity. There's something not quite natural about them – something manufactured and solemn. If you use computers in grade two or three you simply understand that they're around and they're going to help you whenever you feel like using them."

Teachers are faced with devising methods of guiding computer studies and providing resources for students, some of whom could soon outstrip them in programming ability. This can be an intimidating task, but student enthusiasm should make it stimulating and challenging as well. "We have in the microcomputer one of the most incredible forces that has ever happened in education," says Butterfield. "I'm not talking about games; games don't last very long. Students are begging for access to this logic device. It has no precedent. I don't know what specific educational objectives are precisely to be served. All I know is there must be something in the whole phenomenon, some need in the young mind that causes an intense urge to interface with the computer, to try things, to make the computer do things."

Part of the appeal, he believes, comes from the creative nature of programming. "Programming is creative not necessarily in the most visible sense. If you write yourself another *Space Invaders* it might end up looking like everybody else's. I sometimes like to compare programming, especially machine language programming, which is more exacting, to doing a jigsaw puzzle. Why would you sit there for two or three days and put in all this effort when you know that the end result will be a rather crummy-looking picture? The point is that you will have felt you have accomplished something, that you have brought together a number of skills, and even though it's the same as everyone else's, in a sense you have created it. It's the same thing with programming – you feel so good when it all comes together, when it all works."

### Expert Debugging

But what if it doesn't work? When you're the ranking expert, what do you do when you get stuck on a problem? "Well, when you reach a certain stage, and it really isn't all that hard to achieve, then you have control of all parts of the machine. Once you get to that point, and there are many people who have achieved that, you don't have to ask anybody. You can go in there and look for yourself. One of the messages that I try to deliver to people is, 'If I can do it, you can do it.' Because often there isn't anything in the problem that logically you can't look at."

### The Future Of Personal Computing

As personal computer enthusiasts grow wiser and more mature in the next few years, so will their machines, Butterfield predicts. Memory will be cheap, machines more powerful, and at the same time less expensive. The biggest single change will probably be a move toward better human interface. Full-screen editing, color, sound, and graphics will be almost universal and easier to use. Peripherals such as light pens, paddles or joysticks will simply plug in. Features such as upper/lowercase letters, now viewed as optional by some companies, will be standardized. There will be some moves toward better languages, but, Butterfield says, "BASIC appears to be indestructible at present."

More specifically, Butterfield offers some opinions on the future of microcomputer manufacturers: "I think we can say with some certainty that IBM will survive, not necessarily because of the merit of its products, but because IBM will gather around itself a massive amount of support. Radio Shack is very strong. Like IBM it will probably survive for reasons not directly associated with quality. This is not a reflection on its quality, but it has access to so many outlets of its own that it can support continuing sales. Atari has so far suffered from its games image.

"One of the most interesting phenomena could be Sinclair," he says. "Sinclair has introduced a series of small, not very powerful, but remarkably inexpensive computers. While people who are used to the speed of, say, a PET or a VIC would find some of the existing Sinclair computers very slow, we can't ignore the fact that Sinclair through Timex is going to sell an astonishing number of machines."

Butterfield foresees a very interesting battle between these less expensive machines, which are likely to be sold in every corner drug store, and the more powerful products. He notes that people tend to be loyal to a product line, and so far Sinclair's line has a clearly defined top end. Whether this situation will change as a result of demands from buyers of machines such as the ZX-81 who want to upgrade their systems remains a matter for speculation.

As computer prices drop, it is likely that people will begin to see a computer as an affordable tool for the family's financial management, entertainment, and education. Wider distribution of machines will affect society in several ways. Already, of course, people use home computers in a limited way for business, and more commonly for enjoyment and exercise of mental agility. "People test themselves against their computers by asking,

# PETSPEED® lets COMMODORE users do more...with less...with ease


Petspeed is not only going to change the way you think about your Pet Basic programs—it's going to change the way you use them.

It turns your painfully slow programs into fast, smooth-running, professional software.

Nothing can match the speed, compatibility and trouble-free performance of Petspeed.

**Optimizes Any Program**

While most compilers simply translate from one language to another, Petspeed analyzes your source program and eliminates unnecessary complexities. This speeds your processing time dramatically.

Programs are reduced to smaller components and reassembled into a far more efficient form.

For example, Petspeed:

- automatically uses faster integer arithmetic whenever possible
- automatically handles frequently occurring variables and arrays
- automatically calls all subroutines at maximum speed
- properly locates subroutines for optimum performance
- reduces the size of large programs so less memory is needed
- runs twice as fast as *any* other compiler

**Totally Compatible**

Petspeed will work with any combination of Commodore 4000 and 8000 series systems.

It will compile any Basic application.

And, Petspeed is already in daily use in major companies around the world, helping them work more efficiently and more profitably with their Commodore computers.

**Easy-To-Use**

You don't have to add compiler directives. Simply type in the program name, and in less than two minutes you'll see your program run significantly faster.

You'll see improvements no one can match.

**Built-in Security**

Petspeed is provided with a special security device, but you won't need a special runtime key for any compiled programs.

Instead, you can build in your own protection.

No one else can list your Petspeed code, so no one can tamper with your completed programs.

Your programs belong to you and you alone.

If you'd like to do more work in less time with your Commodore computer, contact your local Commodore dealer today. Or, call or write us for complete information and the name of the dealer nearest you.

SSE (415) 964-8201

SMALL SYSTEMS ENGINEERING
222 B View Street • Mountain View, CA 94041

'Can I make the computer do this task?' People also go to the computer for something resembling relaxation.

"I was talking to a microcomputer owner who is having difficulties in his business. He told me he goes home, speaks to no one, and works on his computer for an hour or so. Only when he shuts the machine off does he say 'hello' to everyone. He finds the computer a very great pacifier in some sense – perhaps he takes his energies out on it. He feels that he comes out of that environment more of a human being, and his family is very understanding of it."

Butterfield also feels that people armed with the facts rather than the myth of computers are better equipped to cope with society.

"The most important change that small computers have brought is they have restored to the individual a sense that he has control over the events around him. Not only can his computer calculate a mortgage as well as his bank can, but he has control in that he will not simply accept any nonsense the computer prints and mails to him. Essentially, it's related to the question of competence. If you can handle these little beasts, then in one sense, at least, you are more competent. You understand more about some of the things which are happening in the world around you. That in itself is probably one of the most profound things microcomputers do."

As we become more aware of a computer's true capabilities and limitations, we also may better assess the complex arguments about artificial intelligence. Butterfield defines it very simply: "A computer which adapts its behavior based on what it has learned from external sources is showing artificial intelligence." He cites a game called "Animals" as a simple example of a program which learns from the user. "Animals says it will guess any animal you can name. The first few times, you're going to name an animal it has never heard of. It will ask you for more information about the animal and put it in its list. Eventually you will run out of animals you know, and then it will know as much as you do."

Videotex is another computer-based system with great possibilities for the future – one which he fears will not reach its potential. "I wish I could see a stronger future for videotex. Things like Telidon, Prestel and so on have a conceptual problem for me. They seem to be predominantly one-way only communications systems, perhaps a little bit like television, only not as effective. You have a few people communicating to a lot of people. I don't view that as a good move, or even a typical move in this day where people are getting competence in their own hands. I think that if Telidon were more of a two-way interface, if more people could contribute, then you might have more of what I would call a lively medium."

### Rest And Diversion

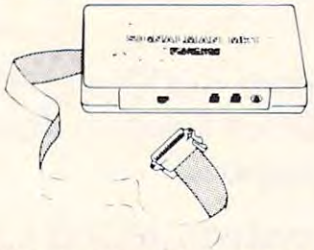
Now that Butterfield finds himself constantly occupied with computers, he must force himself to get away from them for relaxation. Prowling around whatever city he happens to be visiting is one of his favorite diversions. He adds, "I do play the piano quite badly. Occasionally I go and dig dandelions out of the garden if I have time. But there is a little bit of change in the order of things. Since my hobby has become my work, I can't do it all the time."

In many ways, Butterfield has achieved celebrity status. He is much sought after by the microcomputer community around the world and does enjoy the travel. Yet he remains very approachable. "It's really great fun. It's nice to be invited over to England. But simply if any part of it is intimidating to others – if I hear people say 'Well, that's all right for Jim Butterfield' – then I feel ... not good. Essentially, what I'm trying to say is, 'If I can do it, you can do it.' " ©

## SPECIALS on INTEGRATED CIRCUITS

| | | | | |
|---|---|---|---|---|
| 6502 | 7.45 | 10/6.95 | 50/6.55 | 100/6.15 |
| 6502A/6512A | 8.40 | 10/7.95 | 50/7.35 | 100/6.90 |
| 6520 PIA | 5.15 | 10/4.90 | 50/4.45 | 100/4.15 |
| 6522 VIA | 6.45 | 10/6.10 | 50/5.75 | 100/5.45 |
| 6532 | 7.90 | 10/7.40 | 50/7.00 | 100/6.60 |
| 2114-L200 | | 2.45 | 25/2.30 | 100/2.15 |
| 2114-L300 | | 2.25 | 25/2.10 | 100/2.00 |
| 2716 EPROM | | 4.90 | 5/4.50 | 10/4.00 |
| 2532 EPROM | | 8.90 | 5/8.45 | 10/7.90 |
| 6116 Hitachi 2K×8 CMOS RAM | | 8.90 | 5/8.45 | 10/7.90 |
| 4116-200 ns RAM | | | | 8 for 15 |
| Zero Insertion Force 24 pin Socket | | | | 2.00 |
| S-100 Wire Wrap Socket | | | | 2.40 |

**A P Products 15% OFF**
**A P Hobby-Blox 15% OFF**



## MODEM SPECIAL $89

SIGNALMAN Mk1 from Anchor Automation

DIRECT CONNECT Modem with RS232 Cable and Connector included. Fully compatible with all Bell 103 modems. 0 to 300 bps, full duplex, frequency shift keyed modulation, auto-select of ANSW/ORIG, direct connect, audible tone carrier detect indicator, self-contained battery powered.

| | |
|---|---|
| **Anchor ATARI Modem** | **89** |
| **Anchor PET/CBM Modem** | **169** |
| **COMPACK Intelligent Terminal Package** | **115** |

Combines intelligent RS232 port hardware from cgrs Microtech with EHS Intelligent Terminal software to allow you to connect any RS232 Modem to PET/CBM.

| | |
|---|---|
| **DC HAYES Smartmodem** | **230** |
| **RS232 MODEM – Accoustic** | **119** |

## We carry Apple II+ from Bell & Howell

| | |
|---|---|
| **16K RAM Card for Apple** | **65** |
| Apple LOGO | 150 |
| Video Recorder Interface | 545 |
| Super Serial Card | 149 |
| Thunderclock Plus | 119 |
| Z80 Softcard and CP/M | 295 |
| Parallel Printer Interface/Cable | 85 |
| Integer BASIC Card | 150 |
| Grappler Interface | 139 |
| Apple Paddle Pair | 29 |
| **T G Products Joystick for Apple** | **48** |
| T G Paddles | 32 |
| DC Hayes Micromodem II | 299 |
| Videx 80 Column Card | 259 |
| fullFORTH+ for Apple (fig-Forth) | 69 |
| Silentype Printer and Card | 310 |
| Graphics Tablet and Card | 645 |
| Apple PASCAL Language | 195 |
| Apple FORTRAN | 160 |
| We stock EDUWARE Software | |
| GENIS I Courseware Development System | 185 |
| Unicom Grade Reporting | 250 |
| Unicom School Inventory (resource mgmt) | 250 |
| Executive Breifing System with fonts | 225 |

## commodore

**We stock the complete Commodore Line. Come see us for Personal, Business, and Educational requirements.**

**Educational Discounts available.**

**Some used Commodore equipment available.**

### PETSCAN I $345 base price.

Allows you to connect up to 20 CBM/PET Computers to shared disk drives and printers. Completely transparent to the user. Perfect for schools or multiple word processing configurations. Base configuration supports 3 computers.

| | | | |
|---|---|---|---|
| VIC 20 | 239 | VIC Superslot | 24 |
| VIC 1515 Printer | 335 | VIC Super Alien | 24 |
| VIC 3K RAM | 32 | Meteor Run (UMI) | 39 |
| VIC 8K RAM | 53 | VIC Radar Ratrace | 24 |
| VIC 16K RAM | 99 | Amok (UMI) | 20 |
| VIC 1540 Disk Drive | 500 | Snakman | 13 |
| VIC Avengers (Invaders) | 24 | Rubik's Cube | 13 |
| VIC Jupiter Lander | 24 | Programmers Reference | 15 |
| Spiders of Mars (UMI) | 39 | Renaissance (UMI) | 39 |
| **VICTORY Software for VIC** | | | |
| Street Sweepers | 36 | Maze in 3-D | 12 |
| Night Rider | 11 | Cosmic Debris | 9 |
| Treasurers of Bat Cave | 17 | Grave Robbers Adventure | 11 |
| Games Pack I | 12 | Games Pack II | 12 |

| | |
|---|---|
| TNW 488/103 with DAA | 450 |
| Compute!'s First Book of PET /CBM | 11 |
| WordPro 3 Plus - 32K CBM, disk, printer | 195 |
| WordPro 4 Plus - 8032, disk, printer | 300 |
| VISICALC for PET, ATARI, or APPLE | 190 |
| **SM-KIT - PET ROM Utilities** | **40** |
| Programmers Toolkit - PET ROM Utilities | 35 |
| PET Spacemaker II ROM Switch | 36 |
| 2 Meter PET to IEEE or IEEE to IEEE Cable | 40 |
| Dust Cover for PET | 8 |
| IEEE-Parallel Printer Interface for PET | 110 |
| IEEE-RS232 Printer Interface for PET | 120 |
| The PET Revealed | 17 |
| Library of PET Subroutines | 12 |
| SADI Intelligent IEEE-RS232 or parallel | 235 |
| **Programming the PET/CBM (Compute!)** | **20** |
| Compute: First Book of VIC | 11 |
| **Best of Midnight Gazette** | **8** |
| **4 Part Harmony Music System for PET** | **60** |

| | |
|---|---|
| REVERSAL (Spracklen) 32K Apple | 28.00 |
| Data Manager (Lutus) 24K Apple | 40.00 |
| Data-Graph (Boyd) 48K Apple | 40.00 |
| Apple II User's Guide (Osborne) | 12.00 |
| Introduction to Pascal (Sybex) | 10.30 |
| Pascal Handbook (Sybex) | 12.00 |
| Musical Applications of Micros (Chamberlin) | 20.00 |
| User Guide to the Unix System | 13 |
| 6502 Assembly Language Subroutines | 11 |
| PET Fun and Games | 8 |

## DISK SPECIALS

| | | | |
|---|---|---|---|
| SCOTCH (3M) 5" | 10/2.45 | 50/2.35 | 100/2.30 |
| SCOTCH (3M) 8" | 10/2.60 | 50/2.45 | 100/2.40 |

### WE STOCK VERBATIM DISKS

Try the new Verbatim Head Cleaner Kits

| | | | |
|---|---|---|---|
| BASF 5" or 8" | 10/2.00 | 20/1.95 | 100/1.85 |
| Wabash 5" | 10/1.80 | 50/1.75 | 100/1.70 |
| Wabash 8" | 10/2.25 | 50/2.20 | 100/2.10 |

### WE STOCK MAXELL DISKS

| | | |
|---|---|---|
| Diskette Storage Pages | | 10 for 3.95 |
| Disk Library Cases | 8" - 2.85 | 5" - 2.15 |
| Disk Hub Rings | 8" - 50 @ 7.50 | 5" - 50 @ 6.00 |

**CASSETTES - AGFA PE-611 PREMIUM**

High output, low noise, 5 screw housings.

| | | | |
|---|---|---|---|
| C-10 | 10/.56 | 50/.50 | 100/.48 |
| C-30 | 10/.73 | 50/.68 | 100/.66 |

## SPECIALS

| | | |
|---|---|---|
| **EPSON MX-80 Printer with Graftrax+** | | **$** |
| **EPSON MX-80 F/T Printer with Graftrax+** | | |
| **EPSON MX-100 Printer with Graftrax+** | | |
| Zenith ZVM-121 Green Phosphor Monitor | | 109 |
| OKIDATA | | 465 |
| STARWRITER Daisy Wheel Printer F10 | | 1445 |
| D C Hayes Smartmodem | | 230 |
| **We Stock AMDEK Monitors** | | |
| Watanabe Intelligent Plotter 990 | 6-Pen | 1240 |
| BMC BM12A Green Phosphor Monitor | | 85 |
| **MEMORY for IBM 256K+ parity** | | **500** |
| Staticide anti-static spray | | 6 |
| dBASE II | | 445 |

**ALL BOOK and SOFTWARE PRICES DISCOUNTED**

### Synertek Systems

| | |
|---|---|
| **SYM-1 Microcomputer SALE** | **189** |
| SYM BAS-1 BASIC or RAE 1/2 Assembler | 85 |
| KTM-2/80 Synertek Video and Keyboard | 349 |
| KTM-3/80 Synertek Tubeless Terminal | 385 |

### ZENITH data systems

| | |
|---|---|
| Z90-80 64 K | 2170 |
| Z90-82 64K, 1 double dens. drive | 2395 |
| Z67 10 Megabyte + Floppy Drive | 4495 |
| Z37 1.3 Megabyte Dual Floppy | 1495 |
| Z25 High Speed Printer | 1195 |
| Z19 Video Terminal (VT-52 compatible) | 695 |
| ZVM-121 Green Phosphor Monitor | 109 |
| All Zenith Software discounted | |
| **ZT-1 Intelligent Communications Terminal** | **550** |
| **Z100 16-bit/8-bit System** | **CALL** |

### ATARI SPECIALS

| | | | |
|---|---|---|---|
| 800 Computer | 675 | Microsoft BASIC | 72 |
| 400-16K | 269 | **MISSILE COMMAND** | **29** |
| 810 Disk Drive | 440 | **ASTEROIDS** | **32** |
| 825 Printer | 629 | **STAR RAIDERS** | **37** |
| 850 Interface | 170 | Space Invaders | 32 |
| Inside Atari DOS | 18 | Music Composer | 45 |
| Paddle Pair | 19 | Chess | 30 |
| Joystick Pair | 19 | **Anchor Modem-Atari** | **89** |
| 16K RAM | 69 | **PAC-MAN** | **36** |
| 32K RAM | 99 | **CENTIPEDE** | **36** |
| Pilot | 68 | First Book of Atari | 11 |

Write for prices on other Atari items.

252 Bethlehem Pike
Colmar, PA 18915

215-822-7727

## A B Computers

**WRITE FOR CATALOG**

Add $1.25 per order for shipping. We pay balance of UPS surface charges on all prepaid orders. Prices listed are on cash discount basis. Regular prices slightly higher. Prices subject to change.

## KMMM Pascal for PET/CBM $85

A subset of standard Pascal with extensions.

- Machine language Pascal Source Editor with cursor oriented window mode
- Machine Language P-Code Compiler
- P-Code to machine language translator for optimized object code
- Run-time package
- Floating point capability
- User manual and sample programs

Requires 32K   Please specify configuration.

## EARL for PET (disk file based) $65

**Editor, Assembler, Relocater, Linker**

Generates relocatable object code using MOS Technology mnemonics. Disk file input (can edit files larger than memory). Links ,multiple object programs as one memory load. Listing output to screen or printer. Enhanced editor operates in both command mode and cursor oriented "window" mode.

## RAM/ROM for PET/CBM

**4K or 8K bytes of soft ROM with optional battery backup.**

RAM/ROM is compatible with any large keyboard machine. Plugs into one of the ROM sockets above screen memory to give you switch selected write protectable RAM.

Use RAM/ROM as a software development tool to store data or machine code beyond the normal BASIC range. Use RAM/ROM TO LOAD A ROM image where you have possible conflicts with more than one ROM requiring the same socket. Possible applications include machine language sort (such as SUPERSORT), universal wedge, Extramon, etc.

| | |
|---|---|
| RAM/ROM - - 4K | $85 |
| RAM/ROM - - 8K | 120 |
| Battery Backup Option | 30 |

## SUBSORT by James Strasma $35

Subsort is an excellent general purpose machine language sort routine for PET/CBM computers. Sorts both one and two dimensioned arrays at lightning speed in either ascending or descending order. Other fields can be subsorted when a match is found, and fields need not be in any special order. Sort arrays may be specified by name, and fields are random length. Allows sorting by bit to provide 8 categories per byte. The routine works with all PET BASICs, adjusts to any memory size, and can co-exist with other programs in high memory.

## SuperGraphics 2.0

**NEW Version with TURTLE GRAPHICS**

SuperGraphics, by John Fluharty, provides a 4k machine language extension which adds 35 full featured commands to Commodore BASIC to allow fast and easy plotting and manipulation of graphics on the PET/CBM video display, as well as SOUND Commands. Animations which previously were too slow or impossible without machine language subroutines now can be programmed directly in BASIC. Move blocks (or rocketships, etc.), or entire areas of the screen with a single, easy to use BASIC command. Scroll any portion of the screen up, down, left, or right. Turn on or off any of the 4000 (8000 on 8032) screen pixels with a single BASIC command. In high resolution mode, draw vertical, horizontal, and diagonal lines. Draw a box, fill a box, and move it around on the screen with easy to use BASIC commands. Plot curves using either rectangular or polar co-ordinates (great for Algebra, Geometry and Trig classes.)

The SOUND commands allow you to initiate a note or series of notes (or even several songs) from BASIC, and then play them in the background mode without interfering with your BASIC program. This allows your program to run at full speed with simultaneous graphics and music.

Seven new TURTLE commands open up a whole new dimension in graphics. Place the TURTLE anywhere on the screen, set his DIRECTION, turn him LEFT or RIGHT, move him FORWARD, raise or lower his plotting pen, even flip the pen over to erase. Turtle commands use angles measured in degrees, not radians, so even elementary school children can create fantastic graphic displays.

Specify machine model (and size), ROM type (BASIC 3 or 4)

| | |
|---|---|
| SuperGraphics (disk or tape) | $ 40 |
| SuperGraphics in ROM | 55 |

Volume discounts available on ROM version for schools.

## FLEX-FILE for PET/CBM Computers

Self Calculating DATA BASE REPORT WRITER MAILING LIST

FLEX-FILE is a set of flexible, friendly programs to allow you to set up and maintain a data base. Print files with a versatile Report Writer or a Mail Label routine. Programmers will find it easy to add subroutines to their own programs to make use of Data Base files.

### RANDOM ACCESS DATA BASE

Record size limit is 250 characters. The number of records per disk is limited only by the size of each record and the number of records per disk is limited only by the size of each record and the amount of free space on the disk. File maintenance lets you step forward or backward through a file, add, delete, or change a record, go to a numbered record, or find a record by specified field. The Find command locates any record when you enter all (or a portion of) the desired key. Field lengths may vary from record to record to allow maximum packing of information. Files may be sorted by any field, and any field may be specified as a key. Sequential files from other programs may be converted to Flex-File format, and Flex-File records may be converted to sequential (WordPro, PaperMate, other word processors may also use Flex-File data). Maximum record size, fields per record, and order of fields may be changed at any time.

### MAILING LABELS

With typical record size of 127 characters, each disk can handle over 1000 records (about 2800 with 8050 drive). Labels may be printed any number wide, and may begin in any column position. There is no limit on the number or order of fields on a label, and two or three fields may be joined together on one line (like first name, last name, and title). A "type of customer" field allows selective printing.

### REPORT WRITER

Print any field in any column. For numeric fields, use decimal point justification (and round to any accuracy). Define any column as a series of mathematical functions performed on other columns. These functions include arithmetic operations and various log and trig functions. Pass results of operations such as running total from row to row. At the end of the report, print total and/or average for any column. Complete record selection, including field within range, pattern match, and logical functions can be specified individually or in combination with other parameters.

**FLEX-FILE BY Michael Riley $80**

Please specify equipment configuration when ordering.

## COLORCHART Color Video Board $125

Up to 8 colors and high resolution graphics (128 x 192 in 2 colors). Generates composite video for use with monitor.

## PEDISK II from cgrs Microtech

| | |
|---|---|
| 5" 40 track, 1 drive, 143K | $525 |
| 5" 40 track, 1 drive, 286K | 690 |
| 8" IBM 3740 format, 77 track, 250K | 995 |

## PROGRAM YOUR OWN EPROMS

**Branding Iron for PET/CBM $79**

EPROM Programmer with software for all ROM versions. Includes all hardware and software to program or copy 2716 and 2532 EPROMs.

**PORTMAKER DUAL RS232 SERIAL PORT 63**

Two ports with full bipolar RS232 buffering. Baud rates from 300 to 4800. For PET/CBM, AIM, SYM.

**CBM Software**

| | |
|---|---|
| Legal Time Accounting Package | 445 |
| Medical Accounting Package | |
| Dow Jones Portfolio Management | 129 |
| Personal Tax Calculator | 65 |
| Wordcraft 80 Wordprocessor Package | 325 |
| Pascal Development Package | 235 |
| Assembler Development Package | 99 |
| Intelligent Terminal Emulator | 30 |
| Softpac-1 (Competitive Software) | 29 |

## FORTH for PET

BY L. C. Cargile and Michael Riley **$50**

Features include:

- full FIG FORTH model.
- all FORTH 79 STANDARD extensions.
- structured 6502 Assembler with nested decision making macros.
- full screen editing (same as when programming in BASIC).
- auto repeat key.
- sample programs.
- standard size screens (16 lines by 64 characters).
- 150 screens per diskette on 4040, 480 screens on 8050.
- ability to read and write BASIC sequential files.
- introductory manual.
- reference manual.

Runs on any 16K or 32K PET/CBM (including 8032) with ROM 3 or 4, and CBM disk drive. Please specify configuration when ordering.

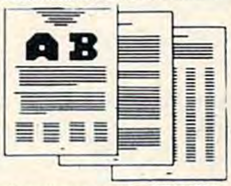
Available soon:

**Metacompiler for FORTH $30**

simple metacompiler for creating compacted object code which can be executed independently (without the FORTH system).

## PaperMate 60 COMMAND WORD PROCESSOR

by Michael Riley




Paper-Mate is a full-featured word processor for CBM/PET. Paper-Mate incorporates 60 commands to give you full screen editing with graphics for all 16K or 32K machines (including 8032), all printers, and disk or tape drives.

For writing text, Paper-Mate has a definable keyboard so you can use either Business or Graphics machines. Shift lock on letters only, or use keyboard shift lock. All keys repeat.

Paper-Mate text editing includes floating cursor, scroll up or down, page forward or back, and repeating insert and delete keys. Text block handling includes transfer, delete, append, save, load, and insert.

All formatting commands are imbedded in text for complete control. Commands include margin control and release, column adjust, 9 tab settings, variable line spacing, justify text, center text, and auto print form letter (variable block). Files can be linked so that one command prints an entire manuscript. Auto page, page headers, page numbers, pause at end of page, and hyphenation pauses are included.

Unlike most word processors, CBM graphics as well as text can be used. Paper-Mate can send any ASCII code over any secondary address to any printer.

Paper-Mate functions with 16/32K CBM/PET machines, with any printer, and with either cassette or disk.

To order Paper-Mate, please specify configuration.

**Paper-Mate on disk or tape 40.00**

PaperMate works on 16K VIC Computer also.

## BASIC INTERPRETER $200

Designed to support the CBM 8096 (8032 with add-on 64K board). A full interpreter implementation to automatically take advantage of the extra memory available to the 8032.

| | |
|---|---|
| BPI General Ledger – 8032/8050 | 300 |
| BPI Accounts Receivable – 8032/8050 | 300 |
| BPI Inventory – 8032/8050 | 300 |
| BPI Job Costing | 300 |
| BPI Payroll | 300 |

**JINSAM Data Base Management System for CBM.** Comprehensive version available for most configurations.

**COPY-WRITER Word Processor 159**

Works like expensive word processors, plus has added features like double column printing, and shorthand generator. For PET/CBM and Apple.

**CASH MANAGEMENT SYSTEM $45**

Easy to use. Keeps track of cash disbursements, cash receipts, cash transfers, expenses for up to 50 categories.

252 Bethlehem Pike
Colmar, PA 18915

215-822-7727

**A B Computers**

**WRITE FOR CATALOG**

Add $1.25 per order for shipping. We pay balance of UPS surface charges on all prepaid orders. Prices listed are on cash discount basis. Regular prices slightly higher. Prices subject to change.

*There are some solid arguments in favor of avoiding BASIC's* INPUT *command. This article offers some suggestions and substitutions for Apple, PET/CBM and VIC users.*

# Banish INPUT Statements!

Richard Cornelius
Department of Chemistry
Wichita State University
Wichita, KS

Nearly any useful microcomputer program requires the user to enter information through the keyboard. Game paddles, joysticks, light pens, and voice recognition all have their places, but rarely do these techniques displace keyboard input entirely in a program. Given the importance of keyboard input, attention should be given to handling it properly.

On microcomputers, the BASIC language supports two kinds of statements that can be used to transfer information from the keyboard to the program: GET and INPUT. Each of these statements has its strong points and its weak points. The principal advantage of the INPUT statement is that it is easy to use. Unfortunately, it has a serious weakness: when an INPUT statement is used, control of the program is relinquished to the computer. Until RETURN is pressed, the computer is in control and may do some things that are not desirable.

The dangers of INPUT are so great that I have totally abandoned its use when writing programs that others will use. In this article I will tell you why I don't use it and will show you how I do what INPUT statements do (and do it better) by using GET statements. My experience is limited primarily to the PET and Apple II computers, but the same principles can be applied to other computers as well. [*References here to "PET" apply as well to the VIC.*]

### The INPUT Statement

First let us examine the INPUT statement to see why it is troublesome. The simplest form of the INPUT statement is illustrated by the following statement:

```
100 INPUT N
```

On either the Apple or the PET, when the program reaches this statement a question mark and a flashing cursor appear on the screen (provided, of course, that on the Apple text page one is being displayed), and program operation comes to a halt. The user may type in nearly any character, but the computer takes no action until RETURN is pressed. While the computer waits for that magic RETURN key stroke, even the STOP key on the PET or CTRL-C on the Apple are inoperative. When RETURN is pressed, the variable N is set to be whatever number was typed in. A more useful form of the input statement provides a prompting statement for the user. A typical statement might look like this on the Apple:

```
110 INPUT "ENTER YOUR NAME, PLEASE: ";
    NAME$
```

When the quotation marks are used on the Apple, the question mark of the INPUT statement is suppressed. If you want a question mark to appear, you can always add it within the quotation marks. In the sample statement 110 no question mark has been used since no question is asked. A space is placed within the quotation marks after the colon to prevent the prompting statement and the user's input from running together. On the PET the question mark cannot be suppressed. The best approach (if you must use INPUT statements) is to use a prompting statement in the form of a question. On the PET computer the function of statement 120 might be better served by a statement such as this one:

```
120 INPUT "WHAT IS YOUR NAME"; NAME$
```

When the program encounters this line, it will print on the screen

```
WHAT IS YOUR NAME?
```

because of the mandatory question mark provided by the computer.

Each of the BASIC statements that has been presented also results in the presence of a flashing cursor after the input prompt. The presence of the cursor on the screen can be highly distracting and may or may not be desirable. Programmers need a cursor during editing procedures and may be conditioned to expect one on the screen, but the best practice is for the programmer to decide whether a cursor should appear. The use of the INPUT statement removes that decision from the hands of the programmer.

### A Greater Weakness

Although the removal of control from the programmer is undesirable, a more significant shortcoming of the INPUT statement is that it wrests control from the user. The INPUT statement essentially halts all action by the computer until RETURN is pressed. After RETURN is pressed,

# Simple... and cheap!

## $995*

## The Olivetti DY211 Daisy Wheel Printer

- Superior print quality
- Easy to use on single sheet letterheads
- Switch selectable 10, 12, 15 pitch and proportional spacing
- Bi-directional with shortest path seeking
- Paper handling accessories easily available
- Interchangeable printing wheels and ribbon cartridges
- 8-bit parallel interface standard

a.I.D.


ADVANCED INTEGRATED DEVELOPMENT

5901 John Martin Dr., Suite 140
Minneapolis, MN 55430
Telephone: 612/561-1645


**ORDERING INFORMATION:**

| | |
|---|---|
| *Shown with optional Single Sheet Feed: | $495.00 |
| **Other options available:** | |
| Bi-Directional Tractor Feed: | $149.00 |
| RS 232C Interface: | $100.00 |
| IEEE 488 Interface: | $100.00 |

Send check or money order. Include $15.00 for shipping anywhere in the Continental U.S. C.O.D.'s no extra charge.

the computer examines what has been entered and, if it can, assigns the data that have been entered to the variable in the INPUT statement. The computer may also take some other action, depending upon what has been entered. For example, if "THREE" (followed by a RETURN) is typed as a response to the question mark resulting from statement 100, the computer will respond with

**?REENTER (on the Apple)**

or

**?REDO FROM START (on the PET)**

If the number 11,324 is typed and then RETURN is pressed, the computer will come back with

**?EXTRA IGNORED**

because it understands the comma as a delineator which ends one input value and gets ready for the next. It tries to interpret the number 11,324 as the number 11 followed by the number 324.

Statements 110 and 120, which use the string variable NAME$, do increase somewhat the control the programmer has in formulating a response, since either numbers or letters may be typed in. However, if the user types in her name as "DOE, JANE" the computer will once again respond

**?EXTRA IGNORED**

as it assigns "DOE" to NAME$ and discards "JANE", for which it had no variable name. These error messages may be familiar and decipherable to a computer programmer, but they will likely appear strange and cryptic to the uninitiated.

Not only do these computer-generated responses perplex the user, but they also defy the programmer. If the user, after finding "THREE" an unsatisfactory input, tries "FOUR", "FIVE", etc., the text on the screen may be scrolled off the screen into oblivion. On the Apple the text window could be defined (by using POKEs to positions 34 and 35) to a single line, but then the coding becomes awkward and tedious. Furthermore, no additional explanatory message can be provided to aid the user. The only messages that will appear are those that the computer generates. As a result, the user cannot control the program because the program cannot control the machine.

Other related problems also occur. For example, the cursor control keys on the PET remain active while the INPUT statement has control of the machine. They can be used to move the cursor any place on the screen. If the insert key is pressed before the cursor control keys, the results can be totally meaningless to the novice user. Finally, CLEAR can wipe out the entire screen. On the Apple the left arrow key is used to back up in order to reenter characters, but if it is pressed before any characters are entered or after the cursor is moved back to the original spot, the cursor moves to the beginning of the next line. These movements are beyond the control of the programmer. Thus, the user may inadvertently enter data someplace other than where the programmer intended and could destroy a portion of a display that was intended to remain on the screen.

Most **COMPUTE!** readers have enough computer experience to understand how the problems with the INPUT statement arise and, as a result, will probably get into difficulty only rarely. The person just beginning, however, may become discouraged when greeted by uninterpretable responses and by disrupted screen images. That person may not come back to the computer again or may return less frequently. Even the experienced user will find that programs free of these problems are easier and more enjoyable to work with. The moral of the story is that if you want your programs to be widely used (who doesn't?), then steer clear of INPUT statements.

### The GET Statement

INPUT is the only BASIC command that directly accepts multi-key entries and assigns to variables the values entered from the keyboard. If INPUT is to be avoided, what can be used to replace its function? The answer is the GET statement. Do you protest that GET is only for single character entry of data? The one-character-at-a-time feature of GET is precisely its advantage. By working this advantage to the fullest, the INPUT statement can be simulated and greatly improved upon.

The GET statement operates differently on the different computers. On the Apple the GET statement halts the operation of the program and displays the flashing cursor until a key is pressed. Any key press (except for the SHIFT or CTRL keys by themselves) will continue the operation of the program. The cursor appears (if the text screen on the Apple is showing) while the computer awaits action at the keyboard. On Commodore machines the GET statement works quite differently. When the statement is encountered in the program, the program is not halted. Instead, whichever key has been pressed (if any) is transmitted to the variable in the GET statement.

Although important differences exist in the way that the GET statement is handled on the two different machines, probably neither way should be regarded as superior. Each approach has its advantages and disadvantages. Perhaps more important is that the way that one computer handles the GET statement can be imitated by appropriate statements on the other computer. To make the computer halt its action, show a flashing cursor,

# VOICE I/O THAT WORKS!

## for the COMMODORE and APPLE II computers

Voice I/O has come a long way from the barely intelligible computer speech of only a few years ago. It is now possible to enter data or commands to your computer just by talking to it and the computer can talk back with clear, pleasant, human sounding voice.

The COGNIVOX models VIO-1002 (for Commodore) and VIO-1003 (for the Apple II+) are at the forefront of a new generation of Voice I/O peripherals that are easy to use, offer excellent performance and are affordably priced.

**SOME SPECIFICATIONS**

COGNIVOX can be trained to recognize up to 32 words or phrases chosen by the user. To train COGNIVOX to recognize a new word, you simply repeat the word three times under the prompting of the system.

COGNIVOX will also speak with a vocabulary of 32 words or phrases chosen by the user. This vocabulary is independent of the recognition vocabulary, so a dialog with the computer is possible. Memory requirements for voice response are approximately 700 bytes per word.

For applications requiring more than 32 words, you can have two or more vocabularies and switch back and forth between them. Vocabularies can also be stored on disk.

**HOW IT WORKS**

COGNIVOX uses a unique single-chip signal processor and an exclusive non-linear pattern matching algorithm to do speech recognition. This gives reliable operation at low cost. In fact, the performance of COGNIVOX in speech recognition is equal or better to units costing many times as much.

For voice output, COGNIVOX digitizes and stores the voice of the user, using a data compression algorithm. This method offers four major advantages: First there are no restrictions to the words COGNIVOX can say. If a human can say it, COGNIVOX will say it too. Second, it is very easy to program your favorite words. Just say them in the microphone. Third, you have a choice of voices: male, female, child, foreign. Fourth and foremost, COGNIVOX sounds very, very good. Nothing in the market today can even come close to the quality of COGNIVOX speech output. You can verify this yourself by calling us and asking to hear a COGNIVOX demo over the phone. Hearing is believing.

**A COMPLETE SYSTEM**

COGNIVOX comes assembled and tested and it includes microphone, software, power supply, built in speaker/amplifier and extensive user manual. All you need to get COGNIVOX up and running is to plug it in and load one of the programs supplied.

It is easy to write your own talking and listening programs too. A single statement in BASIC is all that you need to say or recognize a word. Full instructions on how to do it are given in the manual.

COGNIVOX model VIO-1002 will work with all Commodore computers with at least 16k of RAM. Model VIO-1003 requires a 48k APPLE II+ with 1 disk drive and DOS 3.3.

**ORDER YOUR COGNIVOX NOW**

Call us at (805) 685-1854 between 9am and 4pm PST and charge your COGNIVOX to your credit card or order COD. Or send us a check in the mail, specifying your computer. Price for either model of COGNIVOX is $295 plus $4 shipping in the U.S. (foreign add 10% we ship AIR MAIL).

**VOICETEK**

Dept R, P.O. Box 388 Goleta, CA 93116

Also available for the AIM-65.
Call or write for details.

---

# VIC-20

**VIC-20 INTERFACING BLUE BOOK**

Did you know that your VIC can be used to control a 99¢ toy motor so effectively that it runs like a precision machine? Or that you can build an accurate digital thermometer using the VIC and four other parts costing less than $5?

These and other 18 interfacing projects selected for usefulness, ease of construction and low cost are detailed in the VIC-20 Interfacing Blue Book, a veritable gold mine of practical information on how to build a variety of interfaces for your computer.

Projects include: Connecting VIC to your stereo; Pickproof digital lock; Capacitance meter; Liquid level sensor; Telephone dialer; Voice output; 8K/16K RAM/ROM expansion; 128K RAM expansion; 8-bit precision D/A; 8-bit A/D converter; MX-80 printer interface and more.

Written by a college professor in a friendly and informative style, the Blue Book gives you theory of operation, schematics, program listings, parts list, construction hints and sources of materials for each one of the 20 projects.

If you want to get the most out of your VIC this book is a must. Even if you don't plan to build any of the projects, the Blue Book is a valuable source of information on what can be done with the VIC. Cost is $14.95 (less than 75¢ per project!).

**WORD WHIZ**

Here is a no-frills word processor that does the job and is so small it leaves plenty of memory for your text. Yet if offers full screen editing and easy save of work in progress on cassette, by taking advantage of VIC's built-in text manipulation capabilities. WORD WHIZ prints out on the 1515 printer and is a bargain at $9.95.

**WORD WHIZ/80**

For classy looking output, this version of WORD WHIZ will drive an EPSON MX-80 (See Interfacing info in Blue Book above). Get letter quality printing for only $14.95.

Above prices include postage in the U.S. CA res. add 6% tax. Foreign add $2.

**microsignal**

900 Embarcadero Del Mar, Unit A
Goleta, CA 93117

VIC-20

VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20
VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20 VIC-20

---

**CBM™ CATALOG**

Now there is a catalog reporting system for COMMODORE CBM computers.

**FEATURES:**

- realtime inquiry
- report generator
- enhanced directory function
- wildcard search for volumes and files
- written in basic
- fast and easy to use

cat no. r110-v01 $19.95
specify 4040, 8050, 8250

**MSC software**
1513 Aviation Blvd.
suite a1333
Redondo Beach, CA 90278

trademark
Commodore Business Machines

---

# New Product

## FOR COMMODORE SYSTEMS

# The Commander

This 4K ROM contains exclusive programmable commands. These powerful commands contain an enhanced **COMMON** function which **RETAINS ALL VARIABLES AND ARRAYS.**

A list of some of these **COMMANDS WITH COMMON,** which until now were only available on large systems, are **INSERT, DELETE, APPEND, and RE-DIMENSION.**

**INSERT** - Loads a program or subroutine into the beginning, middle, or between specific line numbers of a running program, without losing variables or arrays. Program execution will continue at any line number, even a new line number just inserted. Insert also allows inserting any part of a program or subroutine.

**DELETE** - Deletes any portion of the running program between specified line numbers, under program control, with **COMMON** function, and continues execution. All deleted memory is reclaimed, and all variables/arrays are retained.

**APPEND** - Appends another program or subroutine (or any part thereof) to the end of the running program, and continues execution without losing variables.

**RE-DIMENSION** - Allows dynamic re-dimension of arrays, while program is running, without losing variables or any array data.

Also included are: **ENHANCED GET, STRING, FRAME, PRINTUSING and IMAGE, RETURN CLEAR, WINDOW, SPEED DATA** and **OVERLAY** commands.

Your Commodore needs THE COMMANDER©. These new commands give the Commodore system powerful features, all under program control. The commands are flexible and easy to use, in either program or direct mode.

THE COMMANDER©: $70.00

(Includes demo/subroutine diskette)
(Florida residents add 5% sales tax)
(Specify socket: $9000 or $A000 )

**METRON** COMPUTER SYSTEMS
4505 Jackson Street • Hollywood, FL 33021
305-962-5183

master charge
VISA

and wait for a keystroke on the Apple requires a single statement:

**140 GET G$**

On the PET we can halt program action by creating a loop which is closed as long as no key is pressed:

**150 GET G$**
**160 IF F$ = "" THEN 150**

Creating a flashing cursor requires a little more work, but that too can be done:

```
200 CURSER$(1) =CHR$(18) + "" + CHR$(146) + CH
    R$(157)
210 CURSER$(2) = "" + CHR$(157)
220 PASS = 1
230 DELAY = 0
240 PRINT CURSER$(PASS);
250 DELAY = DELAY + 1
260 GET G$
270 IF G$ <> "" THEN 310
280 IF DELAY < 24 THEN 250
290 PASS = 3 - PASS
300 GOTO 230
310 PRINT G$
```

In this routine CURSER$ is spelled with an "E" to prevent the syntax error which arises if the variable name CURSOR$ were used. The error is due to the embedded BASIC word "OR" in "CURSOR$". CURSER$(1) is defined as a reverse field character [CHR$(18)] plus a space (" ") plus a reverse field off character [CHR$(146)] plus a cursor left [CHR$(157)]. CURSER$(2) is defined as a space plus a cursor left character. When CURSER$(1) is printed a white block appears, but the position of printing is moved back to the starting position. When CURSER$(2) is printed a space appears, and the printing position is shifted as for CURSER$(1). Thus, depending upon the number of times that the program has PASSed through statement 230, a white block is either printed or eliminated. Statements 250 through 280 cycle for a suitable time interval for the flash, all the while looking to see whether a key has been pressed. Changing the value of the constant in 270 changes the speed at which the cursor flashes.

To have the computer check "on the fly" whether a key has been pressed, and to record which key has been pressed, requires only a single statement on either machine. On a Commodore computer:

**400 GET G$**

On the Apple the same action can be accomplished with the aid of an IF statement:

**410 IF PEEK (49152) >127 THEN GET G$**

The PEEK to location 49152 checks to see whether a key has been pressed. If a key has been pressed, the value of that location will be 128 or greater.

## GET Replaces INPUT

Now that both the INPUT and the GET statements have been examined, it is time to use GET statements to mimic the desirable features of the INPUT statement while eliminating all of the undesirable ones. Input subroutines for the Apple and the PET are given in Programs 1 and 2. The power of these subroutines is that they examine each character *before* it is printed. Characters that are "unacceptable" to the program are never printed to the screen.

Because of the way these particular subroutines are written, only the digits zero through nine and, depending upon the computer, the left arrow key or the DEL key, are acceptable keystrokes. Even the use of the latter keys is restricted so that they are inoperative if the length of the input string (INPUUT$) is zero. Simple changes in IF statements in these subroutines could be made to allow letters but not numbers, or to accept whichever characters are meaningful within the context of the program. Each of these subroutine listings will now be examined in detail.

In the first program, which is for the Apple computer, the first executable statements are in the section labelled "Initialization." Statements 1100-1120 define string variables with easily recognizable names using the CHR$ function. The variable BELL$ will be used to sound the built-in speaker to indicate invalid entry. The CHR$(7) used in defining BELL$ is a CTRL-G, which can even be used from the keyboard to make a bell sound. LEFTARROW$ is the left arrow key (or CTRL-H), which will be used to back up and make changes. The name REETURN$ has an extra "E" in it so that the BASIC interpreter will not give the syntax error that would result from having the valid BASIC word RETURN embedded in the name of a variable. ESC$ is defined as the ESCape key. The name DEELEET$ is another name in which extra E's prevent the generation of syntax errors (due to DEL and LET). When DEELEET$ is printed, the computer backspaces, prints a space, and then backspaces again so that the net effect is to shorten the input data by one character and reposition the (hidden) cursor.

The "Main Program" section prints out statements describing what the program does and then, in statement 1270, calls the input subroutine. Since the five-digit number requested is entered as a string variable in the subroutine, statement 1280 defines a number variable having the value of the number which was entered. When the program continues on to say "PRESS ANY KEY TO CONTINUE," statement 1330 prevents the cursor from appearing while the computer waits for a keystroke.

# BATTERIES INCLUDED

village by the grange, 71 mccaul st. (f6) toronto m5t 2x1 telephone 596-1405

## ARBITER 1.4 MULTI-USER DISK SYSTEM FOR COMMODORE 4.0 COMPUTERS

OVER THREE HUNDRED IN USE ACROSS ONTARIO

Since September 1981 **BATTERIES INCLUDED** has been installing the ARBITER system in classrooms of Commodore BASIC 4.0 computers. The computers are connected to CBM Disk Drives and printers. All users have access to all disk drives and printers plus a host of commands to make this system configuration really usable!

THE ARBITER 1.4 SYSTEM IS READY TO GO!

**FEATURES**

1) Easy installation.
2) Uses no RAM or Utility Sockets.
3) Up to 32 computers in one system.
4) System self initializes on power up.
5) Operation is completely transparent to the user.
6) Extended commands allow a friendly multi-user environment.
7) System design virtually eliminates interleaved printer output.

**$150.00**

**SPECIAL COMMANDS**

@S – Allows students to protect files with a five character password. A three character user ID is forced into the file name.

@L – Allows the students to load protected files if the password code is known.

LISTC – Used to produce program listings with a Commodore printer. Clumsy OPEN, CMD, LIST, PRINT#, CLOSE sequence not needed. It overcomes the listing problems found on other multi-user hardware systems.

LISTP – Used to get program listings on systems which have an ASCII printer. The cursor control characters are expanded and displayed in brackets. e.g. <home>

ALL FILE TYPES ARE SUPPORTED – During relative or sequential file access a delay has been built in so the computer will retain control of the system until the file is closed.

TEACHER UTILITY – A utility is supplied on disk to allow the teacher to produce a hardcopy listing and output from any of the protected or unprotected files selected. Once the files are chosen from the disk directory the teacher may do other tasks while the job is completed.

IF YOUR CLASSROOM WAS DESIGNED TO TEACH COMPUTER LITERACY OR STRUCTURED BASIC THEN THIS SYSTEM WAS DESIGNED FOR YOU.

Arbiter and Arbiter 1.4 are copyrights of Batteries Included.

---

# SWARM-100

## JUST PLUG IT IN

• No soldering • No messy wires

## SOFTWARE SELECTABLE

1. Software select one of two operating systems. (BASIC 2.0 / BASIC 4.0)
2. Software select utility ROMs at conflicting addresses.

**$150.00**


**For 24 Pin ROM Machines Only.**

Master Charge and VISA accepted.

# BATTERIES INCLUDED

Village by the Grange
71 McCaul Street
Toronto, Ontario
Canada M5T 2X1
(416) 596-1405

The heart of the program is the input subroutine which begins in statement 1400. Two initialization steps are the first actions taken. The POKE in statement 1410 clears any keystroke that might have been entered before the subroutine was entered, and the variable (INPUUT$) in statement 1420 is initialized to contain nothing. The PEEK in statement 1430 is the same as in statement 1330 and prevents the cursor from appearing on the screen. Statements 1450 to 1480 control the program flow in response to various keystrokes: if RETURN is pressed then the program RETURNs from the subroutine; if ESC is used, the program terminates; if the left arrow is pressed, the routine at statement 1600 is executed, and if something other than a number is typed, the program simply looks for another key. If a number is typed, then the program checks the number of digits that have been entered (statement 1490), prints out whatever member has been entered (statement 1500), and builds the input string INPUUT$ (so named to prevent a syntax error from the presence of INPUT in a variable name). Statements 1600 through 1650 backspace and delete a character both from the input string and from the screen (unlike a true INPUT statement) provided that something appears on the screen already.

Writing the version of the sample program on the PET is a little neater thanks to the DELete key on the keyboard. This key gives an ASCII code of 20 and in statement 2120 is assigned to the variable DELEETE$. Unlike on the Apple, DEL is not a valid BASIC word, so a single "E" can follow the "D" in DELEETE$. The structure of the two programs is the same, but small differences exist in the details. The mechanism for clearing the input buffer in the PET version is simply repeating the GET G$ statement ten times. The statements 2450 and 2460 make the program wait for a key to be pressed before going on. The only other significant difference is that "Q" for "Quit" is used to get out of the routine at any time. No parallel to the Apple's ESCape key exists on the 40-column PET.

When these routines are used in place of INPUT statements, the program retains control of the machine and, thus, the programmer can give control of the program to the user. In the specific example shown, commas cannot be a confusing issue because they never even appear on the screen. If, of course, a programmer wants commas to be accepted, then simple changes in the IF statements in the subroutines can be used to effect those changes. In addition, single keystrokes (the ESCape key on the Apple and "Q" on the PET) end the program. It is not necessary to hit RETURN as it would be if an INPUT statement were used.

Users operating a program containing GET-simulated INPUT statements may still be able to enter data which has no meaning to the program. The advantage over a standard INPUT statement is that these invalid entries can be handled by the program itself. Invalid entries with the actual INPUT statement may be totally beyond the control of the programmer and, therefore, the user can be powerless.

**Program 1. PET Version**

```
2100 REM **INITIALIZATION**
2110 REM
2120 DELEETE$ = CHR$(20)
2130 REETURN$ = CHR$(13)
2140 MAXLNGTH = 5
2200 REM
2210 REM **MAIN PROGRAM**
2220 REM
2230 PRINT CHR$(147)
2240 PRINT "HERE IS AN EXAMPLE OF HOW GET STATE
     MENTS"
2250 PRINT "CAN BE USED TO MIMIC AN INPUT STATE
     MENT."
2260 PRINT "PRESS 'Q' TO QUIT."
2270 PRINT : PRINT
2280 PRINT "ENTER A FIVE DIGIT NUMBER: ";
2290 GOSUB 2400: REM INPUT SUBROUTINE
2300 NUMBER = VAL(INPUUT$)
2310 PRINT: PRINT
2320 PRINT "YOU HAVE ENTERED THE NUMBER";NUMBER
     ;DELEETE$;"."
2330 PRINT
2340 PRINT "PRESS ANY KEY TO CONTINUE."
2350 GET G$
2360 IF G$ = "" THEN 2350
2370 IF G$ = "Q" THEN PRINT: PRINT "DONE.": END

2380 GOTO 2200
2400 REM
2410 REM **INPUT SUBROUTINE**
2420 REM
2430 FOR I = 1 TO 10: GET G$: NEXT:REM CLEARS I
     NPUT BUFFER
2440 INPUUT$ = ""
2450 GET G$
2460 IF G$ = "" THEN 2450
2470 IF G$ = REETURN$ THEN RETURN
2480 IF G$ = DELEETE$ THEN 2600
2490 IF G$ = "Q" THEN PRINT: PRINT: PRINT "DONE
     .": END
2500 IFG$ < "0" OR G$ > "9" THEN GOTO 2450
2510 IF LEN(INPUUT$) >= MAXLNGTH THEN GOTO 2450

2520 PRINT G$;
2530 INPUUT$ = INPUUT$ + G$
2540 GOTO 2450
2600 REM
2610 REM **DELETE IS PRESSED**
2620 REM
2630 IF LEN(INPUUT$) = 0 THEN GOTO 2450
2640 IF LEN(INPUUT$) = 1 THEN PRINT DELEETE$;:
     GOTO 2440
2650 INPUUT$ = LEFT$(INPUUT$,LEN(INPUUT$)-1)
2660 PRINT DELEETE$;
2670 GOTO 2450
```

**Program 2. Apple Version**

```
1100 REM  **INITIALIZATION**
1110 BELL$ = CHR$ (7)
```

```
1120 LEFTARROW$ = CHR$ (8)
1130 REETURN$ = CHR$ (13)
1140 ESC$ = CHR$ (27)
1150 DEELEET$ = LEFTARROW$ + " " + LEFTARROW$
1160 MAXLNGTH = 5
1200 REM  **MAIN PROGRAM**
1210 HOME : VTAB 3
1220 PRINT "HERE IS AN EXAMPLE OF HOW GET STATE
   MENTS"
1230 PRINT "CAN BE USED TO MIMIC AN INPUT STATE
   MENT."
1240 PRINT "USE ESCAPE TO EXIT THE PROGRAM."
1250 PRINT : PRINT
1260 PRINT "ENTER A FIVE DIGIT NUMBER: ";
1270 GOSUB 1400: REM INPUT SUBROUTINE
1280 NUMBER = VAL (INPUUT$)
1290 PRINT : PRINT
1300 PRINT "YOU HAVE ENTERED THE NUMBER ";NUMBE
   R;"."
1310 PRINT
1320 PRINT "PRESS ANY KEY TO CONTINUE."
1330 IF PEEK (49152) < 128 THEN 1330
1340 GET G$
1350 IF G$ = ESC$ THEN PRINT : PRINT "DONE.": E
   ND
1360 GOTO 1210
1400 REM  **INPUT SUBROUTINE**
1410 POKE 49168,0: REM CLEARS KEYBOARD STROBE
1420 INPUUT$ = ""
1430 IF PEEK (49152) < 128 THEN 1430
1440 GET G$
1450 IF G$ = REETURN$ THEN RETURN
1460 IF G$ = ESC$ THEN PRINT : PRINT : PRINT "D
   ONE.": END
1470 IF G$ = LEFTARROW$ THEN 1600
1480 IF G$ < "0" OR G$ > "9" THEN PRINT BELL$;:
    GOTO 1430
1490 IF LEN (INPUUT$) > = MAXLNGTH THEN PRINT B
   ELL$;: GOTO 1430
1500 PRINT G$;
1510 INPUUT$ = INPUUT$ + G$
1520 GOTO 1430
1600 REM  **LEFT ARROW IS PRESSED**
1610 IF LEN (INPUUT$) = 0 THEN PRINT BELL$;: GO
   TO 1430
1620 IF LEN (INPUUT$) = 1 THEN PRINT DEELEET$;:
    GOTO 1420
1630 INPUUT$ = LEFT$ ( INPUUT$, LEN (INPUUT$) -
    1)
1640 PRINT DEELEET$;
1650 GOTO 1430
```

©

*For the Texas Instruments 99/4A computer, this computer-assisted game could be entertainment at a party or a teaching tool. Requires less than 7000 bytes.*

# CHARADES

Stephen Davis
Dallas

Remember the "good old days" when you had friends over for a party and you all ended up playing games? These days, you probably have your guests lined up at your computer to take turns at Invaders or some other one-man game.

Here is a chance to use your computer to help you entertain the whole group with the old parlor game charades. Charades has always been popular as a party game because it allows many players to participate, it is lively and fast-moving, and it generally leads to a good time.

### Fun And Educational

As well as being a wild and crazy adult game, charades can be great entertainment for the youngsters as well. By altering the data statements in the program, you can substitute a custom list of words or phrases on any number of subjects, making the program quite versatile as an educational tool.

In the original game, you needed someone to keep time and score, and you had to take time to think of phrases, write them on slips of paper, and draw them out of a hat. Now the computer can take care of all those chores for you. For those who have never played, and for those who are just a little rusty, here is a brief rundown of the rules:

Although as few as two can play, it is suggested that an even number of players of six or more participate because the group will be divided into two teams. Players from each team take turns pantomiming phrases to be guessed by the other members of their team. Phrases may be broken into words or syllables, but the player may not talk, write, or form words with his lips while he is pantomiming. He has two minutes to convey the phrase to his teammates, and the time he uses determines his score.

A player should begin by pantomiming the category of the phrase. Categories used in this program include Movies, Books, People, Songs, and Quotes and Clichés. Some of the traditional signals for these categories in charades are:

*Movie* – Hold one hand in front of your face and turn the other one in a circle, as if cranking an old-time movie camera.

*Book* – Put hands together, as if praying, then open them like a book.

*Song* – Hold arms out and open mouth, as if singing.

*Person* – Pat yourself on the head.

*Quote* – Hold hands out with two fingers out on each, as if putting quotes around something.

Holding up a certain number of fingers indicates the number of words in the phrase, which word you are acting, or the number of syllables in a word. Pinching your ear means that the word you are acting "sounds like" the one in your phrase. If you are creative, you will be good at this game.

This program displays the phrase for each player to study before he pantomimes it, so situate your TV screen so that your teammates cannot see it. However, turn up the volume because the program provides an audible "time's up" tone (just like the one on TV game shows that so rudely informs the contestant that, indeed, she did *not* win the washer and dryer).

The program listed here is written in TI BASIC for the Texas Instruments 99/4A Home Computer. It takes advantage of several special routines that the TI offers, including sound capabilities that not only provide audible prompts, but also make timing loops as accurate as possible. Most of the commands can be easily converted to other BASICs, but explanations of the various routines and their functions are documented below. This program includes 125 phrases, and, including data, consumes less than 7000 bytes.

If you get hooked on this game, you may want to substitute your own phrases for variety. Adding words that are of interest to your group (i.e., computers), foreign words, or even X-rated terms, presents all kinds of possibilities. Let your micro liven up your next party with this new slant on an old game.

### Learning How The Program Works

Line 110 sets up arrays for 125 phrases (M$) and a counter to check for duplications (Z). Line 120 assures a different set of random numbers for each game, and the counter "Q" keeps track of how many phrases have been played. CALL CLEAR simply clears the screen. The PRINT statements (and long strings in DATA statements) have been composed for the 28-character screen display of the TI.

The colon in TI PRINT statements indicates a carriage return so that the command PRINT doesn't have to be repeated for each vertical space or new line of text. The subroutine at 1460 (referred

to in line 180) plays the song "Charade." At 280, the GOSUB 1190 waits for the player to press a key before clearing the screen and moving on to the next routine. The loop at 530 reads the 125 phrases into the array (M$), and the loops at 320 and 340 assure that each team has alternating turns and that five rounds make a game.

To shorten or lengthen the game, change the number of rounds (five) in 320. The routine beginning at 950 first indicates whose turn it is, then it generates a random number (X) which, when used as a subscript to M$, will select which phrase will be played next. GOSUB 1380 checks to see if the phrase selected has been used so as to avoid duplication of phrases during the game. In other words, once it is "drawn out of the hat," it is discarded and can't be used again. (It is unlikely that you will play long enough to use all the phrases, but after about 100 have been used, you will naturally notice that it takes a bit longer for the computer to select an unused phrase.) At 1010 the program determines and prints the category of the phrase (there are five groups of 25 phrases); the phrase itself is printed at 1140. Again, the routine at 1190 is used to wait for a signal from the player to clear the screen and start the clock.

The routine at 1280 is the clock, which counts down the time (T) and thus the score. The first CALL SOUND statement in the loop (line 1300) plays an inaudible tone (40,000 hz) at -30db for 750 milliseconds (3/4 of a second), then line 1310 gives the clock a "tick" by sounding a short (20 millisecond) 220 hz tone at -10db.

CALL SOUND is used as a timing device because it can be more accurately adjusted than delay loops; however, a For-Next loop of, say 1 to 250 might be used instead at line 1300. If you hold down a key when your phrase has been guessed, the clock will stop, thanks to lines 1320 and 1330, and the last number displayed (T) becomes your score for that round. Each loop takes a total of one second. To give players more or less than the two minutes allowed here, change the number (120) in line 1280.

Lines 1350 and 1360 provide a loud "time's up" tone and reprint the phrase. After five rounds, a C-major three-note fanfare (at lines 1240-1260) announces the end of that game. The score of the winning team is displayed. If you have played 12 games (and by that time it should be well past your bedtime), lines 490 and 510 end the game before you run out of data.

### If You Have A Talkative Micro

The whole idea of charades, of course, is to convey an idea without speaking. But if you have the Texas

# Not since Gutenberg invented the printing press....

has there been so much talk about printing!

## The Smith-Corona TP-1 Daisy Wheel Printer

# $695.00

- Microprocessor Electronics
- Serial or Parallel Interface
- Simple, Reliable Mechanism

## Act Now: Limited Supply, Low Cost

In the 15th century, a German craftsman and inventor named Johannes Gutenberg originated a method of printing from movable type that was used without significant change until the 20th century.

Today, **Smith Corona,** one of the largest manufacturers of small printers in the world, gives a whole new perspective to printing with their electronic text printer — **TP-1.** Gutenberg would surely approve of the **TP-1,** a microprocessor controlled, high quality **daisy wheel printer.** It produces perfectly formed, executive quality printouts at the speed of 120 words per minute. Typewriter quality printing at dot matrix prices.

Simple, durable and dependable, **TP-1** may be used with word processing systems, microcomputers and most small business systems.

Now, all your letters, documents, forms and reports can have the crisp, professional look you demand — for business or personal use — at an affordable price.

Don't delay. Order your **TP-1** TODAY at the low, low price of **$695.**

**Micro-Printer Marketing** offers same day shipping, nationwide service and invites dealer inquiries. Catalogues available. No shipping charges on prepaid orders.

Additional daisy print wheels ..... $4.95

Additional ribbons ............... $2.95

**Micro™ Printer Marketing**

## Call Micro-Printer Marketing

CALL TOLL FREE **1-800-523-9859**

In PA Call Collect **215/433-3366**

MasterCard and Visa Accepted

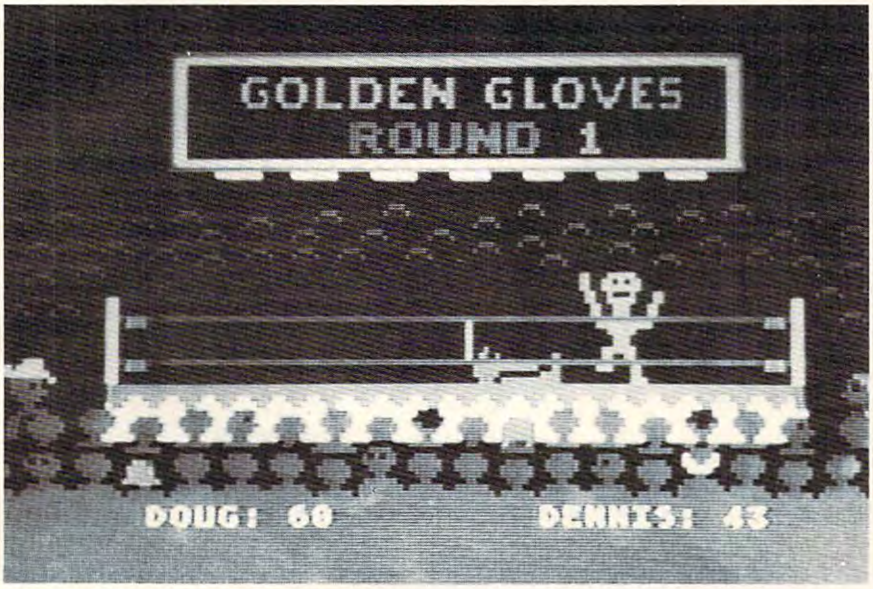
Instruments Speech Synthesizer unit and either the Speech Editor module or the TI Extended BASIC module, then it is okay to let your computer do the talking. If you wish, try adding these lines to the program:

```
185 CALL SAY("DO YOU WANT INSTRUCTIONS")
275 CALL SAY(P$)
335 CALL SAY("THIS IS ROUND NUMBER")
336 CALL SAY(STR$(ROUND))
425 CALL SAY("NUMBER 2 #YOU WIN#")
465 CALL SAY("NUMBER 1 #YOU WIN#")
475 CALL SAY("DO YOU WANT TO PLAY AGAIN")
515 CALL SAY("GAMES OVER. GOODBYE")
965 CALL SAY("NUMBER")
966 CALL SAY(STR$(TEAM))
967 CALL SAY("IT IS YOUR TURN")
968 CALL SAY(P$)
969 CALL SAY("TO SEE YOUR WORDS")
1145 CALL SAY(P$)
1146 CALL SAY("TO START")
1165 CALL SAY(P$)
1166 CALL SAY("TO STOP. GO NOW")
1355 CALL SAY("STOP YOUR TIME IS UP")
```

---

```
100 REM     *CHARADES*     TI BASIC VERSION
     3/20/82
110 DIM M$(125),Z(125)
120 RANDOMIZE
130 P$="       PRESS ANY KEY"
140 Q=1
150 CALL CLEAR
160 PRINT TAB(8);"* CHARADES *"
170 PRINT ::::TAB(6);"COPYRIGHT(C)1982":TA
    B(7);"BY STEVE DAVEIS"::::::
180 GOSUB 1460
190 INPUT "WANT INSTRUCTIONS? (Y/N)":Y$
200 IF Y$<>"Y" THEN 290
210 PRINT :"DIVIDE GROUP INTO 2 TEAMS.":"YO
    U WILL PLAY 5 ROUNDS EACH.":"WHEN
    INSTRUCTED,1 PLAYER"
220 PRINT "FROM TEAM# DISPLAYED SHALL":"PRE
    SS A KEY TO REVEAL HIS":"PHRASE. HE
    SHOULD STUDY IT"
230 PRINT "BEFORE PRESSING A KEY TO":"START
     CLOCK. HE HAS 120 SEC.":"TO
    PANTOMIME THE CATEGORY"
240 PRINT "& PHRASE TO HIS TEAM.":"HE MAY N
    OT TALK OR WRITE.":"WHEN THE PHRASE
    IS GUESSED,"
250 PRINT "HOLD DOWN A KEY UNTIL CLOCK":"ST
    OPS. A TONE WILL SOUND":"WHEN TIME
    IS UP."
260 PRINT "THE LESS TIME YOU USE,":"THE HIG
    HER YOUR SCORE."
270 PRINT "CATEGORIES INCLUDE MOVIES,":"SON
    GS,BOOKS,PEOPLE,AND":"QUOTES & CLI
    CHES."
280 GOSUB 1190
290 GOSUB 530
300 SCOR(1)=0
310 SCOR(2)=0
320 FOR ROUND=1 TO 5
330 PRINT "ROUND #";ROUND:::
340 FOR TEAM=1 TO 2
350 GOSUB 950
360 SCOR(TEAM)=SCOR(TEAM)+T
370 NEXT TEAM
380 NEXT ROUND
390 GOSUB 1240
400 IF SCOR(1)>SCOR(2)THEN 460
410 IF SCOR(1)=SCOR(2)THEN 440
420 PRINT ::"CONGRATULATIONS ,TEAM #2!"::"Y
    OU WIN WITH A SCORE OF ":SCOR(2)
430 GOTO 470
440 PRINT ::"IT'S A TIE! THAT DOESN'T    HA
    PPEN OFTEN!"
450 GOTO 470
460 PRINT ::"CONGRATULATIONS, TEAM #1!"::"Y
    OU WIN WITH A SCORE OF":SCOR(1)
470 PRINT ::"WANT TO PLAY AGAIN? (Y/N)"
480 INPUT Y$
490 IF Q>=120 THEN 510
500 IF Y$="Y" THEN 300
510 PRINT "GAME OVER. OUT OF DATA":"TYPE RU
    N TO START AGAIN"
520 END
530 PRINT "INITIALIZING DATA,STAND BY"
540 FOR I=1 TO 125
550 READ M$(I)
560 NEXT I
570 CALL CLEAR
580 RETURN
590 REM *MOVIES*
600 DATA A MAN AND A WOMAN,MAN WITH THE GOL
    DEN ARM,SOME LIKE IT HOT,MARY POPPINS
610 DATA WHITE CHRISTMAS, MUTINY ON THE BOU
    NTY, ON THE WATERFRONT, YOUNG
    FRANKENSTEIN
620 DATA AGONY AND THE ECSTASY, THE WIZARD ~
    OF OZ, YOU ONLY LIVE TWICE, THE
    LITTLE FOXES
630 DATA DIAL M FOR MURDER, NORTH BY NORTHW
    EST, PSYCHO, LADY SINGS THE BLUES
640 DATA MEET ME IN ST.LOUIS, THE GREAT ZIE
    GFELD,LAURA,THE EMPIRE STRIKES BACK
650 DATA WHERE THE BOYS ARE, DOCTOR ZHIVAGO
    , DOCTOR STRANGELOVE,2001 A SPACE
    ODYSSEY, THE TURNING POINT
660 REM *BOOKS*
670 DATA VALLEY OF THE DOLLS, THE CARPETBAG
    GERS, GONE WITH THE WIND, EVERYTHING
    YOU WANTED TO KNOW ABOUT SEX
680 DATA CATCHER IN THE RYE, THE BIBLE, MAG
    NIFICENT OBSESSION, OLIVER TWIST
690 DATA WOMEN IN LOVE, JANE EYRE, REBECCA,
     ALICE IN WONDERLAND
700 DATA THE HOBBIT, FUTURE SHOCK, GOODBYE ~
    MR. CHIPS, MOBY DICK
710 DATA HUCKLEBERRY FINN, WAR AND PEACE, L
    ITTLE WOMEN, GULLIVER'S TRAVELS
720 DATA BRAVE NEW WORLD, THE SCARLET LETTE
    R, TALE OF TWO CITIES, GIANT, LOLITA
730 REM *PEOPLE*
740 DATA MARILYN MONROE, MARIE ANTOINETTE, ~
    GROUCHO MARX, JOHN KENNEDY
750 DATA MARTIN LUTHER KING, SOPHIA LOREN, ~
    WALTER CRONKITE, SEAN CONNERY
760 DATA ELEANOR ROOSEVELT, JUDY GARLAND, E
    DGAR HOOVER, COLUMBUS
770 DATA GREER GARSON, RONALD REAGAN, LADY ~
    BIRD JOHNSON, NELSON EDDY
780 DATA JOHNNY CARSON, GEORGE WALLACE, CYD
     CHARISSE, GRETA GARBO
790 DATA DOLLY PARTON, JOAN CRAWFORD, BETTE
    DAVIS, PAT NIXON, GEORGE GERSHWIN
800 REM *QUOTES&CLICHES*
```

# ARTWORX SCORES ANOTHER TECHNICAL KNOCKOUT.




*Scene from GOLDEN GLOVES*

**HODGE PODGE:** by Marsha Meredith (Atari and Apple)
NOW AVAILABLE FOR ATARI!!! This captivating program is a marvelous learning device for children from 18 months to 6 years. HODGE PODGE consists of many cartoons, animation and songs which appear when any key on the computer is depressed. A must for any family containing young children.
**PRICE** .................... **$19.95** diskette

**BETA FIGHTER:** by Douglas McFarland (Atari, 16K)
See who will be the ace gunner in this action game set on a spectacular Martian landscape. BETA FIGHTER can be played with one or two players and uses player/missile graphics and delightful sound effects.
**PRICE** ........ **$16.95** cassette **$20.95** diskette

**DRAWPIC:** by Dennis Zander (Atari 16K)
DRAWPIC provides the user with an unbelievably easy way to create screens in graphics modes 3-7. Just sit back with your joystick and use POINT PLOT, DRAW LINE, RUBBER BAND fill and COLOR SET to create beautiful images on your Atari. Full or partial screen images are saved as string data in the program and can be instantly recalled and combined into new images using machine language subroutines. These graphic images can be easily incorporated into your own programs. The images of HODGE PODGE and the landscape of BETA FIGHTER were made using DRAWPIC.
**PRICE** ........ **$29.95** cassette **$33.95** diskette

☐ **ROCKET RAIDERS** by Richard Petersen (Atari 24K)
Defend your asteroid base against pulsar bombs, rockets, lasers, and the dreaded "stealth saucer" as aliens attempt to penetrate your protective force field. Precise target sighting allows you to fire at the enemy using magnetic impulse missiles to help protect your colony and its vital structures.
**PRICE** .............. **$19.95** cassette **$23.95** diskette

☐ **FOREST FIRE TWO:** by Richard Petersen (Atari 24K)
FOREST FIRE has been enhanced and now offers a two player mode for head to head competition to see who can survive, suffer the least damage and put their fire out first. User input now determines landscape, wind and weather conditions, offering limitless game variation. FOREST FIRE's excellent color graphics have been made even better, turning your computer into a super-detailed fire scanner.
**PRICE** .............. **$16.95** cassette **$20.95** diskette

☐ **FORM LETTER SYSTEM:** (Atari, North Star and Apple)
This is the ideal program for creating personalized form letters! FLS employs a simple-to-use text editor for producing fully justified letters. Addresses are stored in a separate file and are automatically inserted into your form letter along with a personalized salutation. Both letter files and address files are compatible with **ARTWORX** MAIL LIST 3.0 and TEXT EDITOR programs.
**PRICE** .............................. **$39.95** diskette

☐ **PILOT:** by Michael Piro (Atari, 16K)
Pilot your small airplane to a successful landing using both joysticks to control throttle and attack angle. PILOT produces a true perspective rendition of the runway, which is constantly changing. Select from two levels of pilot proficiency.
**PRICE** .............. **$16.95** cassette **$20.95** diskette

☐ **TEXT EDITOR:** (Atari and North Star)
This program is very "user friendly" yet employs all essential features needed for serious text editing with minimal memory requirements. Features include common sense operation, two different justification techniques, automatic line centering and straightforward text merging and manipulation. TEXT EDITOR files are compatible with **ARTWORX** FORM LETTER SYSTEM.
**PRICE** .............................. **$39.95** diskette

☐ **MAIL LIST 3.0:** (Atari, Apple and North Star)
The very popular MAIL LIST 2.2 has now been upgraded. Version 3.0 offers enhanced editing capabilities to complement the many other features which have made this program so popular. MAIL LIST is unique in its ability to store a maximum number of addresses on one diskette (typically between 1200 and 2500 names!). Entries can be retrieved by name, keyword(s) or by zip codes. They can be written to a printer or to another file for complete file management. The program produces 1, 2 or 3-up address labels and will sort by zip code (5 or 9 digits) or alphabetically (by last name). Files are easily merged and MAIL LIST will even find and delete duplicate entries! The address files created with MAIL LIST are completely compatible with **ARTWORX** FORM LETTER SYSTEM.
**PRICE** .............................. **$49.95** diskette

☐ **THE VAULTS OF ZURICH:** by Felix and Greg Herlihy (Atari, 24K, PET)
Zurich is the banking capital of the world. The rich and powerful deposit their wealth in its famed impregnable vaults. But you, as a master thief, have dared to undertake the boldest heist of the century. You will journey down a maze of corridors and vaults, eluding the most sophisticated security system in the world. Your goal is to reach the Chairman's Chamber to steal the most treasured possession of all: THE OPEC OIL DEEDS!
**PRICE** .............. **$21.95** cassette **$25.95** diskette

☐ **BRIDGE 2.0** by Arthur Walsh (Atari (24K), Apple TRS-80, PET, North Star and CP/M (MBASIC) systems)
Rated #1 by Creative Computing, BRIDGE 2.0 is the only program that allows you to both bid for the contract and play out the hand (on defense or offense!). Interesting hands may be replayed using the "duplicate" bridge feature. This is certainly an ideal way to finally learn to play bridge or to get into a game when no other (human) players are available.
**PRICE** .............. **$17.95** cassette **$21.95** diskette

☐ **ENCOUNTER AT QUESTAR IV:** by Douglas McFarland (Atari, 24K)
As helmsman of Rikar starship, you must defend Questar Sector IV from the dreaded Zentarians. Using your plasma beam, hyperspace engines and wits to avoid Zentarian mines and death phasers, you struggle to stay alive. This BASIC/Assembly level program has super sound, full player missile graphics and real time action.
**PRICE** .............. **$21.95** cassette **$25.95** diskette

## *NEW PROGRAMS!*

**GOLDEN GLOVES:** by Douglas Evans (Atari 24K)
Use your joystick to jab, block and duck as each player attempts to land the knockout punch. This unique real-time program brings all of the excitement of ringside to your Atari. GOLDEN GLOVES is a one or two-player game, or you can be a spectator as the computer controls both fighters.
**PRICE** ....... **$22.95** cassette **$26.95** diskette

**CRAZITACK:** by Peter Adams (Atari 16K)
The Crazies are attacking us and the only defenses are three MX bases. Missiles can be launched singly or in a salvo, but it is doomsday when you run out of missiles.
**PRICE** ....... **$17.95** cassette **$21.95** diskette

**DOMINATION:** by Alan Newman (Atari 24K)
Between one and six players compete for power via economic, diplomatic and military means in this award-winning game. You must make decisions quickly, exercise skillful hand-eye coordination, outguess your opponents and cope with random events.
**PRICE** ....... **$17.95** cassette **$21.95** diskette

**POKER TOURNEY:** by Edward Grau (Atari 32K, Northstar)
You are entered in a high stakes Draw Poker Tournament facing six opponents including Lakewood Louie, Shifty Pete and Dapper Dan. Each has his own style of play and of bluffing. POKER TOURNEY utilizes the Joker, has true table stakes play and each hand is played based on pot odds. The Atari version's graphics and sound are superb of course (programmed by Jerry White) making POKER TOURNEY the class program of its type.
**PRICE** ....... **$18.95** cassette **$22.95** diskette

**HAZARD RUN:** by Dennis Zander (Atari, 16K)
The sheriff has spotted you and you must make the treacherous run through Crooked Canyon past Bryan's Pond to the jump at Hazard Creek and safety. You can even put the joystick-controlled GEE LEE car up on two wheels to make it through some tight spots. A lead foot is not always the answer as you dodge trees, rocks and chickens in this nerve-racking game. HAZARD RUN employs full use of player/missile graphics, re-defined characters and fine scrolling techniques to provide loads of fast action and visual excitement.
**PRICE** ........ **$27.95** cassette **$31.95** diskette

## *ORDERING INFORMATION*

*Call ARTWORX toll-free number to order direct:*
*800-828-6573*
*In New York, Alaska, Hawaii call:*
*(716) 425-2833*

*All orders are processed and shipped within 48 hours.*

*Shipping and handling charges:*
*Within North America: Add $2.00*
*Outside North America: Add 10% (Air Mail)*
*New York State residents add 7% sales tax*

*Quantity Discounts:*
*Deduct 10% when ordering 3 or more programs*

*Ask for ARTWORX at your local computer store.*

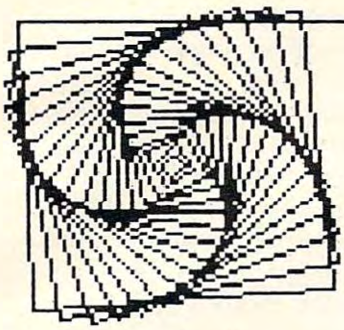
*Write for FREE Catalogue listing more information about these and other quality ARTWORX programs.*

Artworx
150 North Main Street Fairport, NY 14450

```
810 DATA A STITCH IN TIME SAVES NINE, DONT ~
    LOOK A GIFT HORSE IN THE MOUTH,CLEAN
    AS A WHISTLE, NEVER SAY DIE
820 DATA REMEMBER THE ALAMO, IGNORANCE IS B
    LISS, HASTE MAKES WASTE, CONTENTED
    AS A COW
830 DATA ALL THAT GLITTERS IS NOT GOLD, PUR
    R LIKE A KITTEN, I SHALL RETURN,
831 SHARP AS A TACK
840 DATA TO BE OR NOT TO BE, I'LL THINK ABO
    UT THAT TOMORROW, I WANT TO BE ALONE,
    THE BUCK STOPS HERE
850 DATA WE HAVE NOTHING TO FEAR BUT FEAR I
    TSELF, THAT'S ALL FOLKS, WHAT'S UP
    DOC, THERE'S NO PLACE LIKE HOME
860 DATA DONT COUNT YOU CHICKENS BEFORE THE
    Y HATCH, PARTING IS SUCH SWEET
    SORROW, HOLD YOUR HORSES
870 DATA IT'S ALWAYS DARKEST BEFORE THE DAW
    N, HINDSIGHT IS 20/20 VISION
880 REM *SONGS*
890 DATA SANTA CLAUS IS COMING TO TOWN, STA
    RDUST, MY FUNNY VALENTINE, FEELINGS
900 DATA MIDNIGHT BLUE, PEOPLE, CAMP TOWN R
    ACES, SOME ENCHANTED EVENING
910 DATA DO RE MI, I WANNA HOLD YOUR HAND, ~
    YESTERDAY, DOWNTOWN
920 DATA HOUSE OF THE RISING SUN, MY COUNTR
    Y TIS OF THEE, THE LADY IS A TRAMP,
    THE MAN I LOVE
930 DATA ST.LOUIS BLUES, AMERICAN PIE, STOR
    MY WEATHER, OVER THE RAINBOW
940 DATA YOU'VE GOT A FRIEND, MOON RIVER, I
     GOT PLENTY OF NOTHIN, TRY TO
    REMEMBER, YOU'LL NEVER KNOW
950 REM
960 PRINT "TEAM #";TEAM;" -IT'S YOUR TURN":
    :
970 GOSUB 1190
980 X=INT(RND*125)+1
990 GOSUB 1380
1000 REM
1010 IF X<=25 THEN 1070
1020 IF (X>=26)*(X<=50)THEN 1090
1030 IF (X>=51)*(X<=100)THEN 1130
1050 PRINT ::"(SONG)"::::
1060 GOTO 1140
1070 PRINT ::"(MOVIE)"::::
1080 GOTO 1140
1090 PRINT ::"(BOOK)"::::
1100 GOTO 1140
1110 PRINT ::"(PERSON)"::::
1120 GOTO 1140
1130 PRINT ::"(QUOTE&CLICHE)"::::
1140 PRINT M$(X):::::
1150 GOSUB 1190
1160 PRINT "(HOLD DOWN A KEY TO STOP)"
1170 GOSUB 1280
1180 RETURN
1190 PRINT :P$::
1200 CALL KEY(0,KEY,STATUS)
1210 IF STATUS=0 THEN 1200
1220 CALL CLELAR
1230 RETURN
1240 CALL SOUND(300,523,2,392,3,330,3)
1250 CALL SOUND(200,494,2,294,3,247,3)
1260 CALL SOUND(400,523,2,392,3,330,3)
1270 RETURN
1280 FOR T=120 TO 1 STEP -1
1290 PRINT T
1300 CALL SOUND(750,40000,30)
1310 CALL SOUND(20,220,10)
1320 CALL KEY(0,KEY,STATUS)
1330 IF STATUS=0 THEN 1340 ELE 1350
1340 NEXT T
1350 CALL SOUND(1100,220,0)
1360 PRINT:M$(X):::TAB(10);"* * *"::::: 1370
     RETURN
1380 REM TEST FOR DUP
1390 FOR Y=1 TO Q
1400 IF X=Z(Y)THEN 980
1410 NEXT Y
1420 Z(Q)=X
1430 Q=Q+1
1440 RETURN
1450 REM TUNE
1460 DUR=250
1470 CALL SOUND(DUR,262,1)
1480 CALL SOUND(DUR,277,1)
1490 CALL SOUND(DUR,262,1)
1500 CALL SOUND(DUR*2,392,1)
1510 CALL SOUND(DUR,349,1)
1520 CALL SOUND(DUR*3,262,1)
1530 CALL SOUND(100,40000,30)
1540 CALL SOUND(DUR,262,1)
1550 CALL SOUND(DUR,277,1)
1560 CALL SOUND(DUR,262,1)
1570 CALL SOUND(DUR*2,233,1)
1580 CALL SOUNDD(DUR,208,1)
1590 CALL SOUND(DUR,262,1)
1600 CALL SOUND(DUR*3,196,1)
1610 RETURN
```

©

A Monthly Column



# Friends Of The Turtle

David D. Thornburg
Associate Editor

## Battle Of The UFL's

The development of User Friendly Languages (UFL's) is proceeding so quickly that any report is likely to be outdated by the time it appears. Nonetheless, there is enough interest in the UFL's for the Atari, TI, and Apple computers to warrant an overview of the best offerings for these machines. The user friendly languages of principal interest seem to be PILOT and LOGO. It so happens that Atari PILOT (and Apple SuperPILOT) incorporate turtle graphics. While turtle graphics (common to all LOGO's) is not essential for a language to be user-friendly, it helps.

Rather than detail all UFL's for each computer, I will restrict the analysis to Atari PILOT, TI LOGO and Apple LOGO. The differences between Apple LOGO and the Apple versions of LOGO produced by Terrapin and Krell are deserving of separate comment later. (I have just received Krell LOGO and will need to use it some more before writing about it.)

What makes the following comparison interesting is the tremendous difference in price and features of the three chosen language systems. The table summarizes all three configurations. I have listed the bare minimum configuration needed to make the language work. If you want to save your programs, the cost of a recorder must be added to the Atari and TI systems. Since all three systems require a separate display, I have left that item out of the cost analysis.

The entries in this table reflect questions readers have been sending to Friends of the Turtle.

The Atari PILOT system is the least expensive. This results from the low cost of the computer and from the fact that Atari PILOT can be used with a minimum amount of RAM. The increased memory requirement of TI LOGO results in a profoundly increased cost for that system – a cost difference we would not have if we were comparing BASIC's. Since Apple LOGO requires both 64K of RAM and a disk drive, it is the most expensive of the systems. However, Apple LOGO is by far the most powerful of the languages under consideration.

The turtle graphics implementations are ex-

**Table. System Comparison**

| Feature | Atari PILOT | TI LOGO | Apple LOGO |
|---|---|---|---|
| Minimum System | Atari 400 Pilot cartridge | TI 99/4A 32K memory exp. LOGO cartridge | Apple II or II + language card floppy disk LOGO disk |
| List Price | $429 | $980 | $2625 |
| Visible Turtle | No | Yes | Yes |
| Turtle Graphics Resolution | 160 X 80 | 256 X 192 (may "run out of ink") | 280 X 240 (vertical scale is changeable) |
| Number of Simultaneous Colors | 4 | 16 | 6 |
| Total Color Range | 16 hues X 8 luminances | 16 | 6 |
| Character Font Editor | No | Yes | No |
| Multiple Dynamic Turtles | No | 32 | No |
| Real Number Arithmetic | No | No | Yes |
| Unlimited Tail-end Recursion | Yes | No | Yes |
| Recursion Depth Before Crash | 8 | end of memory | end of memory |
| Access to Joysticks, etc. | Yes | No | Yes |
| Full Stroke Keyboard | No | Yes | Yes |
| TV Sound Generator | Yes | Yes (in LOGO II) | No |
| Direct Memory Access | Yes | No | Yes |

cellent in all three systems. Atari PILOT is the only one that does not have a visible turtle, but this can be remedied somewhat with the Visiturt program I published a few months ago (**COMPUTE!**, April 1982, #23). The Atari system has the lowest resolution, but has the greatest color accuracy and range of the three languages. A major annoyance with the TI system is the "out of ink" error that arises when trying to create complex pictures. Since TI creates high resolution graphics by dynamic character definition (a topic for a later column), it is not as versatile as a true memory mapped display. Multiple velocity turtles (turtles that have speeds as well as positions and orientations) are only available on TI LOGO. Up to 32 such animated characters can be created with any of 26 shapes formed in a 16 X 16 dot matrix. While the Atari hardware allows for such animated characters (called players), PILOT users must gain access to these through machine language instructions. Of the three systems, only TI allows the user to interactively modify or define the shapes of characters and velocity turtles.

If you are content with integer arithmetic, any of the systems will do. If you must have access to decimal fractions, only Apple LOGO will meet your needs. Interestingly enough, the restriction to integer arithmetic can result in minor graphics problems (drawing a regular seven-sided polygon, for example) in Atari PILOT and TI LOGO, although these problems can be easily overcome by careful programming.

Recursion is of two types. A simple jump to the beginning of a procedure is called tail-end recursion. Recursion involving the use of a procedure that ultimately returns to the calling procedure is more difficult since the computer must keep track of the sequence and names of all calling procedures. As a result, recursion can use up all free memory just by keeping track of this information. Atari PILOT allows unlimited tail-end recursion (as does Apple LOGO), but allows only eight nested procedure calls.

TI LOGO differs from both Atari PILOT and Apple LOGO in that the user is not provided with access to joysticks nor to the direct reading and alteration of memory. Both Atari PILOT and Apple LOGO have the equivalent of BASIC PEEK and POKE commands to allow the examination and alteration of the contents of arbitrary memory locations.

The keyboard quality is highest for the Apple II, although TI's decision to use a conventional keyboard makes that machine easy to use as well. My experience is that the Atari 400 membrane keyboard is acceptable to children, but is annoying to adults accustomed to typewriters..Since the Atari 800 has a fine full-stroke keyboard, this option is available to those willing to pay the higher price.

In summary, each system has strong features and drawbacks. You are certain to like some aspects of each system. For the price, the Atari PILOT system is beyond comparison. On the other hand, Apple LOGO is a powerhouse of a language, and its features are well worth its price. The ease with which animated sprites can be created and used in TI LOGO makes this system a natural choice for anyone interested in animation.

All three manufacturers are in this business for the long haul, so your selection should be based purely on needs and budget.

### Apple LOGO And The Silentype Printer

Those of you who use the Silentype printer with your Apple computer have probably wondered how to get copies of the displays of turtle graphics created by LOGO procedures. The easiest way I have found is to initialize the printer before loading LOGO. When you initialize a file diskette, it contains a program named HELLO. Normally (for LOGO) there will be no statements in this program. However, if you were to boot this disk first rather than start with the LOGO disk, the HELLO program would be automatically run. Since your Apple already has one dialect of BASIC in ROM (either integer or Applesoft), then you could use a BASIC HELLO program to initialize the Silentype printer.

The Silentype manual shows the numerous ways in which the printer's graphic features can be set up. The default mode lets the printer print bidirectionally. One set of dots is printed as the head moves from left to right and the second set is drawn as it moves from right to left. While this significantly improves the printing speed, the slack in the Silentype mechanism causes this mode to produce unacceptable vertical misalignment when printing high resolution graphics. The unidirectional printing mode does not have this problem.

Second, the Silentype normally prints images just as they appear on the screen. If you have a few white lines on a black background, that is how the printer will print the picture. Normally one expects the reverse of this for line drawings – the background should be white and the lines should be dark. As a result, the printer needs to have its color fields reversed.

Both of these changes are made in the BASIC program shown below. This program assumes that the Silentype printer interface is located in slot #1 and that the disk drive is located in slot #6.

```
10 D$="":REM D$ CONTAINS CTRL-D

20 PRINT D$;"PR #1"
```

```
30 PRINT
40 POKE -12529,255
50 POKE -12524,0
60 PRINT D$;"PR #0"
70 PRINT "GRAPHICS PRINTER INITIALIZED"
80 PRINT "INSERT LOGO DISK AND PRESS RETURN"
90 INPUT A$
100 PRINT D$;"PR #6"
110 END
```

Once this program has been saved in the HELLO file, your printer will be automatically initialized. To set up the printer, you must first start the computer with the file diskette that has this HELLO program. Once the display instructs you to insert the LOGO disk, you should do that and press RETURN. Now you will have LOGO in the computer and also have a properly initialized printer.

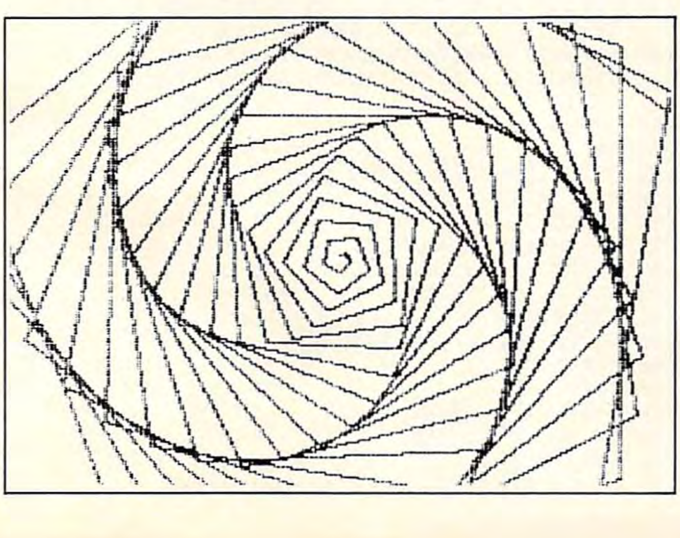
To print a high resolution screen image from LOGO you can use the procedure:

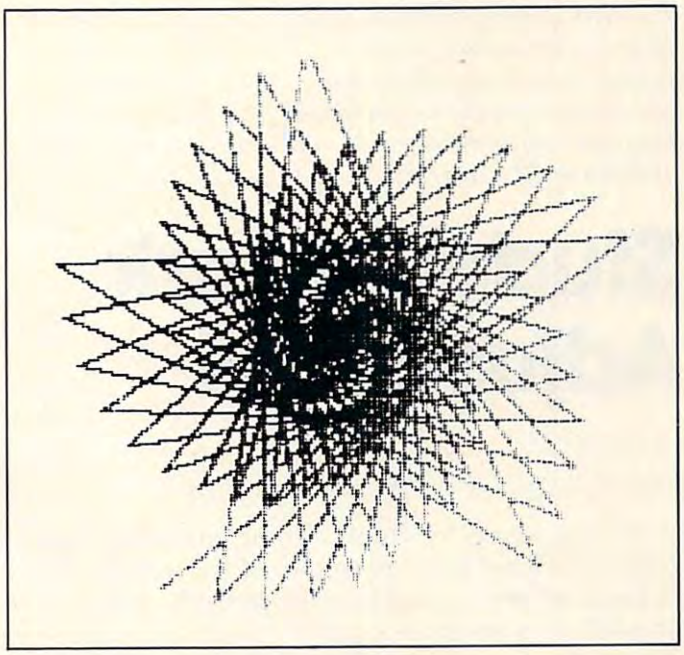
```
TO PICT
.PRINTER 1
PRINT CHAR 17
.PRINTER 0
END
```

From then on, any time you enter PICT the current graphics screen will be copied onto the printer.

If you try printing an image of a square or a circle, you may notice that the image is squashed vertically. This results from a difference in the aspect ratio of the printer and your TV display. To print pictures with a perfect aspect ratio, you must enter

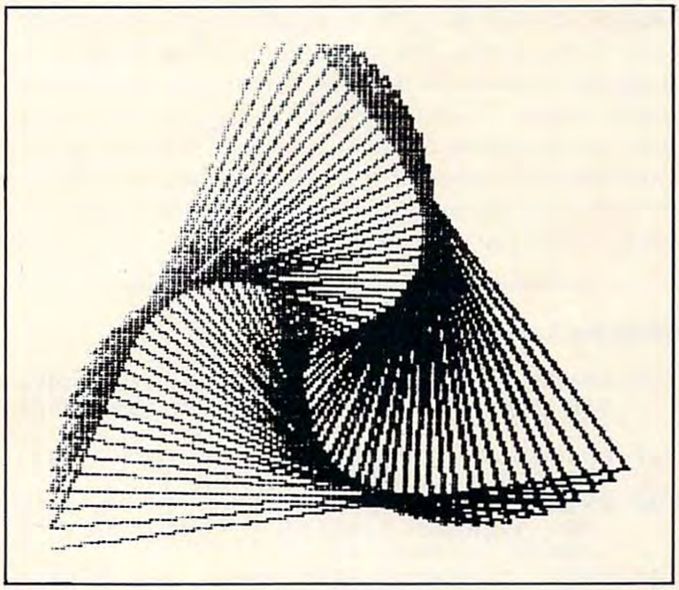
```
SETSCRUNCH 1
```





before drawing the figure you want to print. Once the aspect ratio has been changed to this value (from its default value of 0.8), all your pictures will come out perfectly. The accompanying figures show some of the results.

The Silentype printer is an excellent tool for capturing your LOGO graphic images. It is time you put it to work!



*Teachers sometimes need to adjust a mark distribution. Reasons can include an exam that was too difficult or perhaps two sections of a course taught by different teachers who have very different standards. This useful set of programs calculates accurate adjustments. It is for Atari and Microsoft (Apple, PET, OSI, etc.) BASICs.*

# Student Mark Adjustment

R. D. Wink
Peterborough, Ontario

Adjusting marks by simply adding a constant can cause problems at the extremes of the distribution (a mark of 106% could become possible, for example). This program applies a quadratic regression to the three points (0,0); (X1,Y1) and (M,M) where M is the maximum mark, X1 is the actual mean or median mark, and Y1 is the required mean or median mark.

The output in Programs 1 and 3 is written in single-column form, but it can easily be arranged into several columns depending upon the dimensions of your computer screen.

Since the quadratic curve has a turning point which must be avoided in this application, the change in median must be relatively small to avoid ridiculous results. For example, if the original median turns out to be 50%, the new median must not be below 25% or above 75%. In practice, this is not a serious limitation. You will know if you have exceeded the range of the program because the new marks will rise above the value of M at the high end of the distribution.

### Program 1. Atari Version

```
100 PRINT "{CLEAR}THIS PROGRAM WILL COMP
    UTE A MARK":PRINT " CONVERSION TABLE
    "
110 PRINT "{2 DOWN}PLEASE INPUT THE MAXI
    MUM POSSIBLE MARK";:INPUT M
120 PRINT "{DOWN}PLEASE INPUT THE MEDIAN
     (OR AVERAGE)":PRINT "OF YOUR SET OF
     MARKS";:INPUT X2
130 PRINT "{DOWN}PLEASE INPUT THE MEDIAN
     THAT YOU":PRINT "WOULD LIKE THE CLA
    SS TO HAVE";:INPUT Y2
135 DIM A$(1)
140 GOSUB 370
150 X=0
160 PRINT "{CLEAR}";:R=0:S=5:C=0
170 REM INITIALIZE TAB VARIABLES
180 GOSUB 320
190 REM PRINT HEADINGS
200 POSITION 0,2
210 FOR K=1 TO 20
220 GOSUB 290
230 POKE 85,R+2:PRINT X;:POKE 85,S+2:PRI
    NT P
240 X=X+1
250 NEXT K
260 C=C+1:R=R+10:S=S+10
270 IF C=3 THEN 350
280 GOTO 200
290 P=INT(Y2*X*(X-M)/(X2*(X2-M))+X*(X-X2
    )/(M-X2)+0.5)
300 IF X>M THEN POSITION 2,22:GOSUB 370:
    END
310 RETURN
320 PRINT "OLD  NEW  OLD  NEW  OLD  NEW"
330 PRINT "MARK MARK MARK MARK MARK MARK
    "
340 RETURN
350 GOSUB 370
360 GOTO 160
370 PRINT "PRESS RETURN TO CONTINUE";:IN
    PUT A$
380 RETURN
```

### Program 2. Atari Version

```
5 REM MARK ADJUSTMENT PROGRAM
15 REM
20 REM INPUT SECTION 20-45
25 PRINT "MAXIMUM POSSIBLE MARK";:INPUT
   M
30 PRINT "ACTUAL MEDIAN";:INPUT X1
40 PRINT "DESIRED MEDIAN";:INPUT Y1
45 REM
50 REM COMPUTE AND PRINT CONVERSION TABL
   E
55 PRINT "ORIGINAL MARK{13 SPACES}FINAL
   MARK"
60 FOR X=0 TO M
65 Y=Y1*X*(X-M)/(X1*(X1-M))+X*(X-X1)/(M-
   X1)
70 POKE 85,9:PRINT X;:POKE 85,32:PRINT I
   NT(Y+0.5)
75 NEXT X
80 END
```

### Program 3. Microsoft Version

```
100 PRINT "{CLEAR}THIS PROGRAM WILL COMPUTE ~
    A MARK        CONVERSION TABLE"
110 INPUT "{02 DOWN}PLEASE INPUT THE MAXIMUM P
    OSSIBLE MARK";M
120 INPUT"{DOWN}PLEASE INPUT THE MEDIAN (OR AV
    ERAGE)     OF YOUR SET OF MARKS";X2
130 INPUT"{DOWN}PLEASE INPUT THE MEDIAN THAT Y
    OU WOULD    LIKE THE CLASS TO HAVE";Y2

140 GOSUB 370
150 X=Ø
160 PRINT "{CLEAR}":R=Ø:S=5:C=Ø
170 REM INITIALISE TAB VARIABLES
180 GOSUB 320
190 REM PRINT HEADINGS
200 PRINT "{HOME}{03 DOWN}"
210 FOR K = 1 TO 20
220 GOSUB 290
230 PRINT TAB(R);X;TAB(S);P
240 X=X+1
250 NEXT K
260 C=C+1:R=R+1Ø:S=S+1Ø
270 IF C=3 GOTO 35Ø
280 GOTO 2ØØ
290 P=INT(Y2*X*(X-M)/(X2*(X2-M))+X*(X-X2)/(M-X
```

```
    2)+.5)
300 IF X>M THEN GOSUB 380:END
310 RETURN
320 PRINT "OLD  NEW  OLD  NEW  OLD  NEW"
330 PRINT "MARK MARK MARK MARK MARK MARK"
340 RETURN
350 GOSUB 370
360 GOTO 160
370 INPUT"{DOWN}PRESS RETURN TO CONTINUE      .{0
    3 LEFT}";A$
380 PRINT "{HOME}":FOR K= 1 TO 23:PRINT:NEXT:R
    ETURN
```

---

**Program 4. Microsoft Version**

```
20 REM INPUT SECTION 20-45
25 INPUT"MAXIMUM POSSIBLE MARK";M
30 INPUT"ACTUAL MEDIAN";X1
40 INPUT"DESIRED MEDIAN";Y1
45 REM
50 REM COMPUTE AND PRINT CONVERSION TABLE
55 PRINT"ORIGINAL MARK                FINAL MA
  RK"
60 FOR X=0 TO M
65 Y=Y1*X*(X-M)/(X1*(X1-M))+X*(X-X1)/(M-X1)
70 PRINT TAB(9);X;TAB(32);INT(Y+.5)
75 NEXT X
```

©

---

## BUSINESS and UTILITIES

NEW

PORTFOLIO MANAGEMENT (Apple only) — Price: $69.95 two Diskettes

The PORTFOLIO MANAGEMENT package was written by a stock broker to help manage portfolios for individual customers. With this program data files can be easily created and kept up to date. A variety of reports can be generated for clients which are attractively and professionally laid out. The user may define his/her own investment catagories. PORTFOLIO MANAGEMENT is a top quality, professional tool which will not only provide you with new conveniences but will also serve to enhance your appearance as an efficient and up to date advisor to your clients. Comes complete on two diskettes along with a 30 page instruction manual.

NEW

PERSONAL FINANCE SYSTEM (Available for all computers) — Price: $39.95 Diskette

PFS is a single diskette, menu-oriented system composed of ten different programs. Besides recording your expenses and tax deductible items, PFS will sort and summarize expenses by payee, and display information on expenditures by any of 26 user defined codes by month or by payee. PFS will even produce monthly bar graphs of your expenses by category! This powerful package requires only one disk drive, minimal memory (24K Atari, 32K North Star) and will store up to 600 records per disk (and over 1000 records per disk by making a few simple changes to the programs). You can record checks plus cash expenses so that you can finally see where your money goes and eliminate guesswork and tedious hand calculations. Contains high speed machine language sort. PFS has been demonstrated on network (CBS) TV!

FAMILY BUDGET (Apple and Atari only) — Price: $34.95 Diskette

FAMILY BUDGET is a very convenient financial record-keeping program. You will be able to keep track of cash and credit expenditures as well as income on a daily basis. You can record tax deductible items and charitable donations. FAMILY BUDGET also provides a continuous record of all credit transactions. You can make daily cash and charge entries to any of 21 different expense accounts as well as to 5 payroll and tax accounts. Data are easily retrieved giving the user complete control over an otherwise complicated (and unorganized!) subject.

NEW

TEXT MASTER (Apple 32K, diskette only) — Price: $49.95 Diskette

TEXT MASTER is a general purpose text editor for the Apple II computer. It features powerful, English-oriented commands which permit the complete manipulation of textural information. The information treated may be correspondence, computer programs, data to be used by other programs, and more. TEXT MASTER also interfaces with any printer connected to your Apple. The minimum system requirements are 32K of RAM, Applesoft on ROM, at least one disk, and a lower case adapter. TEXT MASTER can process any length file segment by segment. Thus it is possible to process files as large as a diskette. Comes complete with an efficient 16 page manual. TEXT MASTER is equivalent in capability to many word processing systems costing 3 x or more in price. The commands available in TEXT MASTER are: COLLECT, USE, EXECUTE, NONUM, INSERT, MERGE, STOP, RENUM, DELETE, SAVE, COPY, FREE, REPLACE, WAIT, MOVE, LENGTH, LIST, RESAVE, CLEAR, AUTO, RETRY, APPEND, SET, MANUAL, CHANGE, SCRATCH, SHOW, CATALOG, MODIFY, COMPARE, NUM, DISPLAY, HELP.

INTELINK (Atari only) — Price: $49.95 Diskette

This software package contains a menu-driven collection of programs for facilitating efficient two-way communications through a full duplex modem (required for use). In one mode of operation you may connect to a data service (e.g., The SOURCE or MicroNet) and quickly load data such as stock quotations onto your diskette for later viewing. This greatly reduces "connect time" and thus the service charge. You may also record the complete contents of a communications session. Additionally, programs written in BASIC, FORTRAN, etc. may be built off-line using the support text editor and later "uploaded" to another computer, making the Atari a *very* smart terminal. Even Atari BASIC programs may be uploaded. Further, a command file may be built off-line and used later as controlling input for a time-share system. That is, you can set up your sequence of time-share commands and programs, and the Atari will transmit them as needed; batch processing. All this adds up to saving both connect time and your time.

NEW

PAYFIVE (Apple II plus diskette, two drives required) — Price: $149.95

This is an enormously flexible employee payroll system with extraordinarily good human engineering features. PAYFIVE prints checks and compiles the required federal, state and local forms for up to 148 employees. The pay methods may be hourly, salary, commission or any combination. There are multiple options for pay periods, and they also can be used in any combination. PAYFIVE includes *many* other features and comes extremely well documented with a 200 page manual. The manual may be purchased separately for $30, and that payment later applied to the software purchase.

SHOPPING LIST (Atari only) — Price: $12.95 Cassette/$16.95 Diskette

SHOPPING LIST stores information on items you purchase at the supermarket. Before going shopping, it will remind you of all the things you might need, and then display (or optionally print) your shopping list and the total cost. Adding, deleting, changing and storing data is very easy. Runs with 16K.

NEW

TAX OPTIMIZER (Available for all computers) — Price: $59.95 Diskette

The TAX OPTIMIZER is an easy-to-use, menu-oriented software package which provides a convenient means for analyzing various income tax strategies. The program is designed to provide a quick and easy data entry. Income tax is computed by all tax methods (regular, income averaging, maximum and alternate minimum tax). The user may immediately observe the tax effect of critical financial decisions. TAX OPTIMIZER has been thoroughly field tested in CPA offices and comes complete with the current tax tables in its data files. TAX OPTIMIZER is tax deductible!

STOCK MASTER/STOCK PLOT (Apple 48K) — Price: $59.95 Diskette

This is a full-featured stock portfolio management and analysis system. Ten years of records on up to thirty stocks may be maintained. You may record prices, revenue, earnings/share, ROE, quarterly earnings and dividends, transactions, long/short term gains, P/E, indexes (DJ, S & P, NASDAQ) and MORE! You may plot (HIRES) the price history of any stock against any other, time, or the indexes. Portfolio value may be evaluated at any time. Comes complete with superbly written instructions and sample files on a second diskette.

TURNKEY AND MENU (Atari only) — Price: $17.95 Diskette

TURNKEY is a utility program which allows you to create autoboot/autorun diskettes easily. Simply load and run TURNKEY, load the program diskette to be modified, and answer the questions! The TURNKEY diskette also comes with DOS 2.0 and includes another program, MENU. MENU lists the contents of your diskette alphabetically, and permits the running of any BASIC program on the diskette by typing a single key. TURNKEY and MENU provide you with the ability to run any program on your diskette by simply turning on the computer and pressing a single key.

STOCKAID (Atari only) — Price: $29.95 Diskette

STOCKAID provides a powerful set of tools for stock market analysis. With STOCKAID you can display point and figure charts, as well as bar charts with oscillators. You can also examine long term moving averages and on-balance volume features. STOCKAID allows you to input daily data with a single diskette storage capability of 239 days x 16 stocks. Included are stock dividend and split adjustment capabilities. A very professional package!

NEW

NYINDEX (Atari only) — Price: $29.95 Diskette

NYINDEX is a comprehensive software package for storing, retrieving and plotting New York Stock Exchange information. The daily data treated includes the composite index, advances, declines, new highs and new lows. Graphical displays include the above plus the index oscillator, cumulative advances/declines and moving averages. Data entry and editing is easy. The diskette includes more than two years of daily data. NYINDEX is an excellent companion to STOCKAID.

PLAYER-MISSILE GRAPHICS TABLET (Atari only) — Price: $19.95 Diskette

The PLAYER-MISSILE GRAPHICS TABLET was designed to take the drudgery out of developing four color displays in GRAPHICS MODE 7. No longer will you have to read the locations of those tiny blocks on your graph paper and calculate PLOTs and DRAWTOs. With PMG you will be able to easily design colorful graphic displays with your joystick and save them on diskette for later recall.

LIFE CYCLE ANALYSIS AND DEPRECIATION (Apple diskette only) — Price: $39.95 Diskette

This software package creates a data file of business expenses for equipment which can later be used to calculate and display a variety of reports. You may project annual costs, find the present worth, create depreciation schedules and justify tax deductions. The evaluation techniques conform to standards set by federal agencies. This is an invaluable package for any businessman who has invested in equipment. LIFE CYCLE ANALYSIS features an easy to use data file creation section and provides formatted hardcopy reports for use in presentations or for tax record keeping purposes. When used for generating tax information, this package is tax deductible! Requires 48K. Comes on two diskettes.

NEW

MICROMAGIC (Apple diskette only) — Price: $39.95 Diskette

The emphasis of this program is clearly the MAGIC! MICROMAGIC offers outstanding versatility in its ability to function as a stand alone entertainment package or as a utility program to create stunning *animated* graphics for use in other programs. The secret lies in MICROMAGIC's special on screen graphics editor. You control a graphics cursor directly from the keyboard, creating high resolution images using all 16 available colors. When you are done with a picture, it can be saved on disk with a single key command. Up to 24 images can be saved as "frames" of a movie, and then played back at high speed to create short animated sequences. The effects are truly stunning.

This package comes complete with demonstration programs so that even novice users can get immediate results. No programming skills are necessary to use MICROMAGIC. If you have been frustrated by the effort required to create graphics images with your computer, MICROMAGIC will delight you.

SHAPE MAGICIAN (Apple II, 48K, diskette only) — Price: $29.95

At last! An utility for painlessly creating graphics shapes for the Apple. Create, edit and save up to 30 shapes which can then be used to develop arcade games or to simply enhance your programs. Add that professional touch!

## ORDERING INFORMATION

All orders are processed and shipped within 48 hours. Please enclose payment with order and include the appropriate computer information. If paying by VISA or Master Card, include all numbers on card. Purchase orders accepted.

Shipping and Handling Charges
Within North America: Add $2.00
Outside North America: Add 15% (Air Mail)

Delivery
All orders (excluding books) are sent First Class.

Quantity Discounts
Deduct 10% when ordering 3 or more programs; 20% when ordering 5 or more. Dealer discount schedules available upon request.

8" CP/M Disks
Add $2.50 to the listed diskette price for each 8" floppy disk (IBM soft sectored CP/M format). Programs run under Microsoft MBASIC or BASIC-80.

5¼" CP/M Disks
All software available on 8" CP/M disks is also available on 5¼" disks, North Star, Osborne, Superbrain and NEC format.

Ask for DYNACOMP programs at your local software dealer. Write for detailed descriptions of these and other programs from DYNACOMP.


**DYNACOMP, Inc.** (Dept. E)
1427 Monroe Avenue
Rochester, New York 14618
24 hour message and order phone: (716) 442-8731
Toll free order phones: (800) 828-6772
(800) 828-6773
Office phone (9AM-5PM EST): (716) 442-8960


New York State residents please add 7% NYS sales tax.

## EDUCATION

HODGE PODGE (Apple 48K only) — Price: $14.95 Cassette/$18.95 Diskette

Let HODGE PODGE be your child's teacher. Pressing any key on your Apple will result in a different and intriguing "happening" related to the letter or number of the chosen key. The program's graphics, color and sound are a delight for children from ages 1½ to 7. HODGE PODGE is a non-intimidating teaching device which brings a new dimension to the use of computers in education. See the excellent reviews of this very popular program in INFOWORLD and SOFTALK.

TEACHER'S AIDE (Atari and PET only) — Price: $13.95 Cassette/$17.95 Diskette

TEACHER'S AIDE consists of three basic modules contained in one program. The first module provides addition and subtraction exercises of varying levels of difficulty. The second module consists of multiplication problems in which the student may be tested both on the final answer and/or on the subtotal answers in the long hand procedure. Several levels of complexity are provided here as well. The third module consists of division problems; one particularly nice feature of the division module is that the long hand division steps can be displayed along with the remainder in order to clearly demonstrate the procedure by which the remainder is derived. Using TEACHER'S AIDE is not merely a drill, but rather a learning experience.

## STATISTICS and ENGINEERING

DIGITAL FILTER (Available for all computers) — Price: $39.95 Cassette/$43.95 Diskette

DIGITAL FILTER is a comprehensive data processing program which permits the user to design his own filter function or choose from a menu of filter forms. In the explicit design mode the shape of the frequency transfer function is specified by directly entering points along the desired filter curve. In the menu mode, ideal low pass, high pass and bandpass filters may be approximated to varying degrees according to the number of points used in the calculation. These filters may optionally also be smoothed with a Hanning function. In addition, multi-stage Butterworth filters may be selected. Features of DIGITAL FILTER include plotting of the data before and after filtering, as well as display of the chosen filter functions. Also included are convenient data storage, retrieval and editing procedures.

DATA SMOOTHER (Not available for Atari) — Price: $19.95 Cassette/$23.95 Diskette

This special data smoothing program may be used to rapidly derive useful information from noisy business and engineering data which are equally spaced. The software features choice in degree and range of fit, as well as smoothed first and second derivative calculation. Also included is automatic plotting of the input data and smoothed results.

FOURIER ANALYZER (Available for all computers) — Price: $19.95 Cassette/$23.95 Diskette

Use this program to examine the frequency spectra of limited duration signals. The program features automatic scaling and plotting of the input data and results. Practical applications include the analysis of complicated patterns in such fields as electronics, communications and business.

TFA (Transfer Function Analyzer) — Price: $19.95/$23.95 Diskette

This is a special software package which may be used to evaluate the transfer functions of systems such as hi-fi amplifiers and filters by examining their response to pulsed inputs. TFA is a major modification of FOURIER ANALYZER and contains an engineering-oriented decibel versus log-frequency plot as well as data editing features. Whereas FOURIER ANALYZER is designed for educational and scientific use, TFA is an engineering tool. Available for all computers.

HARMONIC ANALYZER (Available for all computers) — Price: $24.95 Cassette/$28.95 Diskette

HARMONIC ANALYZER was designed for the spectrum analysis of repetitive waveforms. Features include data file generation, editing and storage/retrieval as well as data and spectrum plotting. One particularly unique facility is that the input data need not be equally spaced or in order. The original data is sorted and a cubic spline interpolation is used to create the data file required by the FFT algorithm.

FOURIER ANALYZER, TFA and HARMONIC ANALYZER may be purchased together for a combined price of $51.95 (three cassettes) and $63.95 (three diskettes).

REGRESSION I (Available for all computers) — Price: $19.95 Cassette/$23.95 Diskette

REGRESSION I is a unique and exceptionally versatile one-dimensional least squares "polynomial" curve fitting program. Features include very high accuracy; an automatic degree determination option; an extensive internal library of fitting functions; data editing; automatic data, curve and residual plotting; a statistical analysis (e.g., standard deviation, correlation, coefficient, etc.) and much more. In addition, new fits may be tried without reentering the data. REGRESSION I is certainly the cornerstone program in any data analysis software library.

REGRESSION II (Available for all computers) — Price: $19.95 Cassette/$23.95 Diskette

PARAFIT is designed to handle those cases in which the parameters a imbedded (possibly nonlinearly) in the fitting function. The user simply inserts the functional form, including the parameters (A(1), A(2), etc.) as one or more BASIC statement lines. Data, results and residuals may be manipulated and plotted as with REGRESSION I. Use REGRESSION I for polynomial fitting, and PARAFIT for those complicated functions.

MULTILINEAR REGRESSION (MLR) (Available for all computers) — Price: $24.95 Cassette/$28.95 Diskette

MLR is a professional software package for analyzing data sets containing two or more linearly independent variables. Besides performing the basic regression calculation, this program also provides easy to use data entry, storage, retrieval and editing functions. In addition, the user may interrogate the solution by supplying values for the independent variables. The number of variables and data size is limited only by the available memory.

REGRESSION I, II and MULTILINEAR REGRESSION may be purchased together for $51.95 (three cassettes) or $63.95 (three diskettes).

ANOVA (Not available on Atari cassette or for PET/CBM) — Price: $39.95 Cassette/$43.95 Diskette

In the past the ANOVA (analysis of variance) procedure has been limited to the large mainframe computers. Now DYNACOMP has brought the power of this method to small systems. For those conversant with ANOVA, the DYNACOMP software package includes the 1-way, 2-way and N-way procedures. Also provided are the Yates $2^{K-P}$ factorial designs. For those unfamiliar with ANOVA, do not worry. The accompanying documentation was written in a tutorial fashion (by a professor in the subject) and serves as an excellent introduction to the subject. Accompanying ANOVA is a support program for building the data base. Included are several convenient features including data editing, deleting and appending.

BASIC SCIENTIFIC SUBROUTINES, Volumes 1 and 2 (Not available for Atari)

DYNACOMP is the exclusive distributor for the software keyed to the popular texts *BASIC SCIENTIFIC SUBROUTINES, Volumes 1 and 2* by F. Ruckdeschel (see advertisements in BYTE magazine). These subroutines have been assembled according to chapter. Included with each collection is a menu program which selects and demonstrates each subroutine.

Volume 1
Collection #1: Chapters 2 and 3 - Data and function plotting; complex variables and functions.
Collection #2: Chapter 4 - Extended matrix and vector operations.
Collection #3: Chapters 5 and 6 - Random number generators (Poisson, Gaussian, etc.); series approximations.
Price per collection: $16.95 Cassette/$20.95 Diskette
All three collections are available for $44.95 (three cassettes) and $53.95 (three diskettes).
Volume 2
Collection #1: Chapter 1 - Linear, polynomial, multidimensional, parametric least squares.
Collection #2: Chapter 2 - Series approximation techniques (economization, inversion, reversion, shifting, etc.)
Collection #3: Chapter 3 - Functional approximations by iteration and recursion.
Collection #4: Chapter 4 - CORDIC approximations to trigonometric, hyperbolic, exponential and logarithmic functions.
Collection #5: Chapter 5 - Table interpolation, differentiation and integration (Newton, LaGrange, splines).
Collection #6: Chapter 6 - Methods for finding the real roots of functions.
Collection #7: Chapter 7 - Methods for finding the complex roots of functions.
Collection #8: Chapter 8 - Optimization by steepest descent.
Price per collection: $14.95 Cassette/$18.95 Diskette

All eight collections are available for $99.95 (eight cassettes) and $129.95 (eight diskettes).

Because the texts are a vital part of the documentation, *BASIC SCIENTIFIC SUBROUTINES, Volumes 1 and 2* are available from DYNACOMP:

BASIC SCIENTIFIC SUBROUTINES, Vol. 1 (319 pages): $19.95 + 75¢ postage
BASIC SCIENTIFIC SUBROUTINES, Vol. 2 (790 pages): $23.95 + $1.50 postage

See reviews in KILOBAUD, Dr. Dobbs, and ACCESS.

NEW

SOFTNET (Apple II and TRS-80 48K diskette only) — Price: $129.95

SOFTNET may be used to create models of liquid pipeline systems to evaluate their flow performance. Up to 150 nodes with up to 150 connecting elements may be simulated, and models may be combined to form yet larger models. If you are involved in water distribution systems, chemical fluid flow problems, building plumbing, or similar situations, this is an ideal analysis tool.

NEW

FILTER ANALYSIS (Apple only) — Price: $19.95 Cassette/$23.95 Diskette

FILTER ANALYSIS is the ideal program for determining the frequency response of passive filters. Any number of RLC components may be included, and any number of poles treated. FILTER ANALYSIS features its own mini-language which makes circuit description simple. Results may be printed in tabular form or plotted in HIRES graphics (decibels versus log-frequency).

ACTIVE CIRCUIT ANALYSIS (Available for all computers) — Price: $35.95 Cassette/$39.95 Diskette

With ACAP you may analyze the response of an active or passive component circuit. The circuit may be probed at equal steps in frequency, and the resulting complex voltages at each component juncture examined; the frequency response of a filter or amplifier may be completely determined with respect to both amplitude and phase. In addition, ACAP prints a statistical analysis of the range of voltage responses which result from tolerance variations in the components. ACAP is easy to learn and use. Circuit descriptions may be saved onto cassette or diskette to be recalled at a later time for execution or editing. ACAP should be part of every circuit designer's program library. Requires 48K.

LOGIC SIMULATOR (Available for all computers) — Price: $35.95 Cassette/$39.95 Diskette

Test your complicated digital logic design with respect to given set of inputs to determine how well the circuit will operate. The elements which may be simulated include multiple input AND, OR, NOR, EXOR, EXNOR and NAND gates, as well as inverters, J-K and D flip-flops, and one-shots. Inputs may be clocked in with varying clock cycle lengths/displacements and delays may be introduced to probe for glitches and race conditions. A timing diagram for any given set of nodes may be plotted. Save your breadboarding until the circuit is checked by LOGIC SIMULATOR. Requires 48K.

NEW

BEAM DEFLECTION (Available for all computers) (disk/diskette only) — Price: $29.95 Diskette/$32.45 Disk

BEAM DEFLECTION is the first in DYNACOMP's new series of structural analysis software packages. It consists of two programs. The first program permits the development of data files which describe the problem. For example, the ends of the beam may be pinned, clamped or free. The beam may be uniformly supported by an elastic bed, or held up by springs variously placed and having differing spring constants. The elasticity and cross section of the beam may vary along its length. The load may be uniformly distributed or it may include discrete forces. The beam may be pinned at various points along its length. And so on. All this information may be easily entered and edited using the data input program.

Following this the analysis program is called. The calculated results are the stress and deflections of the beam, both in numerical and graphical form. Since the input data is saved, cases may be easily re-run with modification, thereby permitting iterative design.

The documentation which comes with BEAM DEFLECTION clearly shows how to use the software. In addition, three text problems are described and demonstrated to ensure that you understand how to use the program. Also, helpful theoretical information is supplied in the appendix.

STATTEST (Not available on Atari cassette or for PET/CBM) — Price: $19.95 Cassette/$23.95 Diskette

This is a statistical inference package which helps you make wise decisions in the face of uncertainty. In an interactive fashion you can build and edit data files and test the differences in means, variances and proportions. STATTEST will also perform data analysis as well as do linear correlation and regression. This menu-directed statistical workhorse is rounded out with a chi-square contingency test and a (uniform and normal) random sample generator. The documentation is written by a college professor who guides you through the various tests.

## ABOUT DYNACOMP

DYNACOMP is a leading distributor of small system software with sales spanning the world (currently in excess of 50 countries). During the past three years we have greatly enlarged the DYNACOMP product line, but have maintained and improved our high level of quality and customer support. The achievement in quality is apparent from our many repeat customers and the software reviews in such publications as COMPUTRONICS, 80 Software Critique, A.N.A.L.O.G., Softalk, Creative Computing and Kilobaud. DYNACOMP software has also been chosen for demonstration on network television. Our customer support is as close as your phone. It is always friendly. The staff is highly trained and always willing to discuss products or give advice.

A Monthly Column

# The World Inside The Computer

Fred D'Ignazio is a computer enthusiast and author of several books on computers for young people. He is presently working on two major projects: he is writing a series of books on how to create graphics-and-sound adventure games. He is also working on a computer mystery-and-adventure series for young people.
As the father of two young children, Fred has become concerned with introducing the computer to children as a wonderful tool rather than as a forbidding electronic device. His column appears monthly in **COMPUTE!**

## The Talking Head

Fred D'Ignazio
Associate Editor

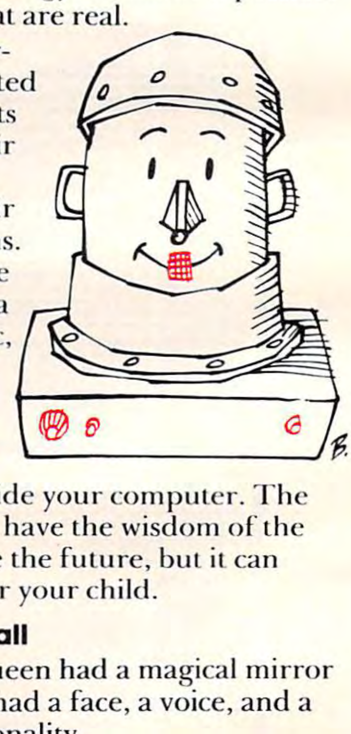
In her book, *Machines That Think,* Pamela McCorduck described the ancient popularity of talking heads. Wise men built the heads, then consulted them for useful advice. For example, a medieval pope, Sylvester II, supposedly built a talking head that answered only when spoken to. It was, in a sense, an early computer: to all questions, it gave only two answers – "yes" or "no"; yet, like modern computers, it was credited with having great wisdom and the ability to foretell the future.

A host of brilliant and famous men kept brazen (brass) heads as advisors, pets, and oracles in their homes. Albertus Magnus, for example, had a head in the form of a "a lovely woman who could speak." The head provided much sage advice, but occasionally its answers were flippant and mischievous. According to legend, the head so offended Albertus Magnus's pupil, Thomas Aquinas, that he kidnapped the head and burned it.

### Conjuring Up A Talking Head

The first talking heads were products of alchemy and magic. They belong alongside all the other creatures of fantasy, myth, and legend.

Today, a thousand years after the first heads appeared, modern technology has made it possible to build talking heads that are real. The heads are computer-controlled robots, mounted on robot bodies. Scientists are building them in their labs. Youthful hobbyists are building them in their workshops and bedrooms.



But if you don't have the time or skill to build a head out of metal, plastic, and servomotors, don't despair. You can "conjure" up a talking head on a program and call it forth from the world inside your computer. The head you create may not have the wisdom of the ages or be able to foresee the future, but it can become a great friend for your child.

### Mirror, Mirror, On The Wall

In *Snow White,* the evil queen had a magical mirror on the wall. The mirror had a face, a voice, and a puckish, irreverent personality.

Your TV picture screen can be like the queen's mirror. When your child turns on the computer and runs the "talking head" program, the mirror will darken. Then, magically, a sleeping face will

appear. Its eyes will blink open. It will wink at the child, talk to the child, and answer his or her questions in a musical voice.

### The Face Of A Friend

Following this column is the "Talking Head" program for Atari, PET, and Apple computers. The talking head program is just the first step toward a much more elaborate "computer friend" program. I described this program in my last column (**COMPUTE!**, August 1982, #27). Next month we will give the friend the ability to play with your child. The following month, we will give your child the ability to teach the friend and shape its character.

### The "Talking Head" Program

*Lines 40 to 130:* Program initialization.

*Lines 500 to 550:* The "Friend Master."

Consists entirely of GOSUBs to subroutines. You need to think of the interaction between your child and the friend in terms of episodes or frames. Each time the friend talks, that's a frame. Each time your child responds, that's another frame. Each GOSUB in this section handles a single frame. You can add new frames by adding new GOSUBs at this master level.

*Lines 1000 to 1110:* "Friend Wake-Up."

In this frame, the friend's face appears. A bell rings, the friend wakes up, blinks its eyes open, and winks at the child.

*Lines 2000 to 3110:* "The Friend Talks."

This is a general-purpose talking subroutine. It enables the friend to read in sequences of DATA statement messages and print them on the screen, in large letters, to the right of the friend's face. The friend can handle words up to nine letters long. It prints words, one per line, on up to five lines per screen. If the friend has a message of more than five words, it will take more than a single screen. That's okay. Each of the friend's messages can be up to nine screens (45 words) long.

*Lines 3200 to 3270:* Friend accepts child's name.

*Lines 4000 to 4880:* Sound effects – including the wake-up bell and the friend's voice.

*Lines 5000 to 5470:* Drawing the friend's face – including the basic face (ski cap, ears, nose, and chin).

Additional routines to animate mouth (5200-5280) and eyes (5300-5470).

*Lines 5500 to 5550:* Clear message window for new message.

*Lines 6000 to 6022:* Friend's messages to the child.

Each new message should begin on line 6000 + some multiple of 10 (for example, 6010, 6020, 6030, etc.).

Each new message begins with the number of message screens in the message (a number from 1 to 9).

Each list of words to appear on a single screen ends with a "-1."

The friend's name (line 6011) is given as "GED." To give the friend a new name, just replace the old name with one of your choosing.

After the friend has asked and received the child's name (the first GOSUB 2010 and GOSUB 3210), you can add the child's name to any message by placing the token character "*" in the message list (e.g., 6022 TO,SEE,YOU,*,-1).

### Warning And Acknowledgment

Remember, this program is just the beginning. It makes a good talking head. But, as yet, the head is not good at answering. Since the head cannot carry on a conversation with the child, it is not yet a computer friend.

Next month we'll give the head the ability to carry on a conversation with your child and play games. Then it will start being a real friend.

I would like to thank Bruce Mitchell for some valuable programming assistance. Also, thanks to Richard M. Kruse for the doorbell sound.

**Program 1. Atari Version**

```
40 GRAPHICS 2+16
50 FOR P=1 TO 800:NEXT P
100 REM *** DIMENSION VARIABLES
110 DIM M$(9):REM * MESSAGE
120 N=1:REM * MESSAGE POINTER
130 DIM NAME$(9):REM * CHILD'S NAME
500 REM *** FRIEND MASTER
510 GOSUB 1010:REM * FRIEND WAKE-UP
520 GOSUB 2010:REM * FRIEND TALK
530 GOSUB 3210:REM * STORE CHILD'S NAME
540 GOSUB 2010:REM * FRIEND TALK
550 END
1000 REM *** FRIEND WAKE-UP
1010 GOSUB 5010:REM * DRAW FACE
1020 GOSUB 5410:REM * DRAW SLEEP EYES
1030 GOSUB 5210:REM * DRAW CLOSED MOUTH
1035 FOR P=1 TO 800:NEXT P
1040 GOSUB 4010:REM * WAKE-UP BELL
1050 GOSUB 5460:REM * DRAW OPEN EYES
1060 FOR P=1 TO 600:NEXT P
1070 GOSUB 5320:REM * WINK EYE
1080 FOR P=1 TO 100:NEXT P
1085 M=0:GOSUB 4820:REM * WINK NOISE
1090 GOSUB 5460:REM * DRAW OPEN EYES
1100 FOR P=1 TO 800:NEXT P
1110 RETURN
2000 REM *** FRIEND TALK
2005 RESTORE 6000+N*10:REM * SELECT MESS
     AGE
2006 N=N+1:REM * SET POINTER TO NEXT SET
      OF FRIEND MESSAGES
2010 READ SNUM:REM * SNUM = NUMBER OF SC
     REENS IN CURRENT SET OF FRIEND MESS
     AGES
2015 FOR K=1 TO SNUM
2020 GOSUB 3010:REM * FRIEND TALK--1 SCR
```

```
     EEN
2033 FOR P=1 TO 800:NEXT P
2035 GOSUB 5510:REM * CLEAR MESSAGE WIND
     OW
2040 NEXT K
2050 RETURN
3000 REM *** FRIEND TALKING--1 SCREEN
3010 PY=2:REM * MESSAGE VERTICAL (Y) STA
     RT LOCATION
3020 PY=2:REM * MESSAGE VERTICAL (Y) STA
     RT LOCATION
3030 PX=14:REM * HORIZONTAL (X) CENTER O
     F MESSAGE ON SCREEN
3040 READ M$
3050 IF M$="-1" THEN RETURN
3051 IF M$="*" THEN M$=NAME$
3055 GOSUB 5260:REM * OPEN MOUTH
3060 POSITION INT(PX-(LEN(M$)/2)+0.5),PY
     :REM * CENTER LINE
3070 PRINT #6;M$
3075 GOSUB 4810:REM * FRIEND SOUND
3080 FOR P=1 TO 10:NEXT P:REM * KEEP MOU
     TH OPEN
3090 GOSUB 5210:REM * CLOSE MOUTH
3095 FOR P=1 TO 50:NEXT P:REM * KEEP MOU
     TH CLOSED
3100 PY=PY+2
3110 GOTO 3040
3200 REM *** FRIEND ASKS CHILD'S NAME
3210 OPEN #1,4,0,"K:"
3215 POSITION 11,4
3217 FOR I=1 TO 9
3220 GET #1,A
3230 IF A=155 THEN 3265
3240 PRINT #6;CHR$(A);
3250 NAME$(LEN(NAME$)+1)=CHR$(A)
3260 NEXT I
3265 FOR P=1 TO 75:NEXT P
3267 GOSUB 5510:REM * CLEAR MESSAGE WIND
     OW
3270 CLOSE #1:RETURN
4000 REM *** WAKE-UP BELL
4010 BEL=105:TIM=7.5:GOSUB 4040
4020 BEL=132:TIM=8.5:GOSUB 4040
4030 SOUND 0,0,0,0:RETURN
4040 VLM=15:INC=0.79+TIM/50
4050 SOUND 0,BEL,10,VLM
4060 VLM=VLM*INC
4070 IF VLM>1 THEN 4050
4080 RETURN
4625 FOR P=1 TO 15:NEXT P
4770 RETURN
4800 REM *** FRIEND VOICE
4810 M=INT(RND(1)*51)+15
4820 FOR A=M+25 TO M STEP -8
4830 SOUND 0,A,10,10
4840 FOR T=1 TO 10
4850 NEXT T
4860 NEXT A
4875 SOUND 0,0,0,0
4880 RETURN
5000 REM *** FRIEND'S FACE
5010 GRAPHICS 2+16
5040 POSITION 2,1:PRINT #6;"{3 SPACES}*"

5050 POSITION 2,2:PRINT #6;"  / \"
5060 POSITION 2,3:PRINT #6;" ====="
5070 POSITION 2,4:PRINT #6;"/{5 SPACES}\
     "
5090 POSITION 1,6:PRINT #6;"<:  ^  :>"
5100 POSITION 2,9:PRINT #6;"\_____/"

5110 RETURN
5200 REM *** CLOSE MOUTH
5210 POSITION 2,7:PRINT #6;":{5 SPACES}:
     "
5220 POSITION 2,8:PRINT #6;": --- :"
5230 RETURN
5250 REM *** OPEN MOUTH
5260 POSITION 2,7:PRINT #6;":  ___  :"
5270 POSITION 2,8:PRINT #6;":  \_/  :"
5280 RETURN
5300 REM *** LEFT EYE WINK
5320 POSITION 2,5:PRINT #6;": O - :"
5330 FOR P=1 TO 150:NEXT P
5340 RETURN
5400 REM *** EYES ASLEEP
5410 POSITION 2,5:PRINT #6;": - - :"
5440 RETURN
5450 REM *** EYES AWAKE
5460 POSITION 2,5:PRINT #6;": O O :"
5470 RETURN
5500 REM *** CLEAR MESSAGE WINDOW
5510 FOR Y=2 TO 8 STEP 2
5520 POSITION 10,Y
5530 PRINT #6;"{9 SPACES}"
5540 NEXT Y
5550 RETURN
6000 REM *** MESSAGES
6010 DATA 3
6011 DATA HI, I'M, GED,-1
6012 DATA YOU,TURNED,ME,ON,-1
6013 DATA WHO'S,OUT,THERE?,-1
6020 DATA 2
6021 DATA I'M,SO,HAPPY,-1
6022 DATA TO,SEE,YOU,*,-1
```

**Program 2. PET/CBM Version with suggested changes for Apple II+**

```
40 PRINT"{CLEAR}":REM HOME FOR APPLE
50 FOR P=1 TO 800:NEXT P
100 REM *** DIMENSION VARIABLES
120 N=1:REM * MESSAGE POINTER
355 GOSUB 5260:REM * OPEN MOUTH
500 REM *** FRIEND MASTER
510 GOSUB 1010:REM * FRIEND WAKE-UP
520 GOSUB 2010:REM * FRIEND TALK
530 GOSUB 3210:REM * STORE CHILD'S NAME
540 GOSUB 2010:REM * FRIEND TALK
550 PRINT:PRINT:PRINT:PRINT:PRINT:END
1000 REM *** FRIEND WAKE-UP
1010 GOSUB 5010:REM * DRAW FACE
1020 GOSUB 5410:REM * DRAW SLEEPING EYES
1030 GOSUB 5210:REM * DRAW CLOSED MOUTH
1035 FOR P=1 TO 800:NEXT P
1040 GOSUB 4000:REM * WAKE-UP BELL
1050 GOSUB 5460:REM * DRAW OPEN EYES
1060 FOR P=1 TO 600:NEXT P
1070 GOSUB 5320:REM * WINK EYE
1080 FOR P=1 TO 100:NEXT P
1085 M=0:GOSUB 4820:REM * WINK NOISE
1090 GOSUB 5460:REM *DRAW OPEN EYES
1100 FOR P=1 TO 800:NEXT P
1110 RETURN
2000 REM *** FRIEND TALK
2005 REM * SELECT MESSAGE
2006 N=N+1:REM * SET POINTER TO NEXT SET OF FRI
    END MESSAGES
2010 READ SNUM:REM * SNUM = NUMBER OF SCREENS I
    N CURRENT SET OF FRIEND MESSAGES
2015 FOR K=1 TO SNUM
2020 GOSUB 3010:REM * FRIEND TALK--1 SCREEN
2033 FOR P=1 TO 1000:NEXT P
2035 GOSUB 5510:REM * CLEAR MESSAGE WINDOW
2040 NEXT K
2050 RETURN
3000 REM *** FRIEND TALKING--1 SCREEN
3010 PY=3:REM * MESSAGE VERTICAL (Y) START LOCA
    TION
```

```
3030 PX=14:REM * HORIZONTAL NEW(X) CENTER OF ME
    SSAGE ON SCREEN
3040 READ M$
3050 IF M$="-1" THEN RETURN
3051 IF M$="*" THEN M$=N$
3052 REM FOLLOWING WOULD BE "VTAB PY:HTAB PX-LE
    N(M$)/2+.5" ON APPLE
3060 POKE 216,PY:PRINT"{UP}";TAB(PX-(LEN(M$)/2)
    +0.5);:REM * CENTER LINE
3070 PRINT M$:GOSUB 5250
3075 GOSUB 4810:REM * FRIEND SOUND
3080 FOR P=1 TO 50:NEXT P:REM * KEEP MOUTH OPEN

3090 GOSUB 5200:REM * CLOSE MOUTH
3095 FOR P=1 TO 100:NEXT P:REM * KEEP MOUTH CLO
    SED
3100 PY=PY+2
3110 GOTO 3040
3200 REM *** FRIEND ASKS CHILD'S NAME
3210 REM
3215 REM USE "VTAB 3:HTAB 10:INPUT N$" HERE FOR
    APPLE
3220 PRINT"{HOME}{03 DOWN}";TAB(10);:INPUT N$
3265 FOR P=1 TO 75:NEXT P
3267 GOSUB 5510:REM * CLEAR MESSAGE WINDOW
3270 RETURN

4000 REM *** WAKE-UP BELL
4005 REM DELETE LINES 4010-4070 FOR APPLE
4010 BEL=105:TM=30:GOSUB 4040
4020 BEL=132:TM=20:GOSUB 4040:RETURN
4040 VLM=15:INC=0.79+TIM/50
4045 POKE59467,16:POKE 59466,51
4050 POKE 59464,BEL
4060 FOR W=1 TO TM*10:NEXT
4070 POKE 59467,0
4080 RETURN
4625 FOR P=1 TO 15:NEXT P
4760 NEXT A
4770 RETURN
4800 REM *** FRIEND VOICE
4801 REM DELETE LINES 4810-4880 FOR APPLE AND C
    HANGE LINES 4810 AND 4820
4802 REM TO "RETURN"
4810 M=INT(RND(1)*51)+15
4820 POKE 59467,16:POKE 59466,51
4825 FOR A=M+25 TO M STEP -8
4830 POKE 59464,A
4840 FOR T=1 TO 10
4850 NEXT T,A
4875 POKE 59467,0
4880 RETURN
5000 REM *** FRIEND'S FACE
5010 PRINT"{CLEAR}":REM USE "HOME" FOR APPLE
5040 PRINT "    *"
5050 PRINT "   / \"
5060 PRINT "  ====="
5070 PRINT " /     \"
5090 PRINT:PRINT "<:  ^  :>"
5100 PRINT::PRINT:PRINT " \$$$$$/":REM "$" IS U
    NDERLINE
5110 PRINT
5200 REM *** CLOSE MOUTH
5205 REM USE "VTAB 7" ON APPLE
5210 PRINT"{HOME}{07 DOWN} :     :"
5215 REM USE "VTAB 8" ON APPLE
5220 PRINT"{HOME}{08 DOWN} : --- :"
5230 RETURN
5250 REM *** OPEN MOUTH
5255 REM USE "VTAB 7" ON APPLE
5260 PRINT"{HOME}{07 DOWN} : $$$ :":REM "$" IS ~
    UNDERLINE
5265 REM USE "VTAB 8" ON APPLE
5270 PRINT"{HOME}{08 DOWN} : \$/ :"
5280 RETURN
5300 REM *** LEFT EYE WINK
5305 REM USE "VTAB 5" ON APPLE
5320 PRINT"{HOME}{05 DOWN} : O - :"
5330 FOR P=1 TO 150:NEXT P
5340 RETURN
5400 REM *** EYES ASLEEP
5405 REM USE "VTAB 5" ON APPLE
5410 PRINT "{HOME}{05 DOWN} : - - :"
5420 RETURN
5450 REM *** EYES AWAKE
5455 REM USE "VTAB 5" ON APPLE
5460 PRINT"{HOME}{05 DOWN} : O O :"
5470 RETURN
5500 REM *** CLEAR MESSAGE WINDOW
5505 REM USE "VTAB 1" ON APPLE INSTEAD OF "{
    HOME}"
5510 PRINT"{HOME}":FOR I=1 TO 10
5530 PRINTTAB(10);"                    "
5540 NEXT I
5550 RETURN
6000 REM *** MESSAGES
6010 DATA 3
6011 DATA HI, I'M, GEB,-1
6012 DATA YOU,TURNED,ME,ON,-1
6013 DATA WHO'S,OUT,THERE?,-1
6020 DATA 2
6021 DATA I'M,SO,HAPPY,-1
6022 DATA TO,SEE,YOU,*,-1
```

©

*For Atari and Apple, this is the first of a three-part series on PILOT. This month, an Apple version (Microsoft BASIC) – and next month an Atari BASIC PILOT will be published in Part II. The Apple version requires Applesoft, 32K memory, and one disk drive.*

## Part I

# Turtle PILOT

Alan W. Poole
Loomis, CA

How would you like a powerful new language for your computer that combines PILOT, turtle graphics, and all commands and functions? The programs at the end of this article create a version of PILOT which contains all of these features. Best of all, unlike most languages available for the Apple, this language isn't going to cost you a fortune. This version of PILOT, which I have named Turtle PILOT, has been patterned after Atari PILOT. Most Atari PILOT programs can be converted to Turtle PILOT without very many changes.

At the end of this article are two program listings. The Turtle PILOT Editor is used for typing PILOT programs. The Turtle PILOT Translator writes a program in Applesoft which is equivalent to a PILOT program typed with the Editor. Both programs require an Apple with Applesoft, 32K, and one disk drive. Also included is an example showing everything typed and printed on the screen while entering and translating a program, along with a sample RUN after it was translated.

### Introduction To PILOT

PILOT is a simple language and is very easy to learn, especially if you already understand BASIC. In this article I will assume that the reader is familiar with Applesoft BASIC. To get an idea of how PILOT works, consider the following program and explanations for each line.

```
1 *QUIZ
2 T:HOW MUCH IS 8+4?
3 A:
4 M:12,TWELVE
5 TY:THAT'S CORRECT.
6 TN:NO, TRY AGAIN.
7 JN:*QUIZ.
```

The first line is a label, which is used to identify sections and modules (modules are subroutines) of a program. Labels always begin with an asterisk. In the second line the T is the instruction name for Type. Everything following the colon will be displayed on the screen. The third line Accepts a response from the user. Line four uses the Match instruction. Each item following the colon is compared with the last response. The Y in line five is the Yes conditioner. The Yes conditioner causes the instruction to be executed only if the last Match succeeded. The N in line six is the No conditioner. A line with a No conditioner will be executed only if the last Match failed. The last line causes a Jump back to the first line if the question was answered incorrectly. Notice that a label is used instead of a line number. The line numbers are not actually part of the program, but are used only to make editing easier.

### Parts Of An Instruction

The order of the parts of an instruction is important. Below is a description of each of the elements in an instruction. Although an instruction does not have to contain all of the optional elements, the elements that are included must be in the order they are given below.

**1.** Instruction Name – The instruction name is a single letter and always comes first. It is required with every line other than a label, since labels are not considered instructions.

**2.** Conditioner – The conditioner is either a Y or an N. The conditioner is optional and may be used with any instruction.

**3.** Expression – An instruction with an expression is executed only when the expression is true. The expression must be placed between parentheses. Expressions are optional and may be used with any instruction. Below are some examples of instructions that include expressions.

```
T(S>1Ø):VERY GOOD.
J(L=SQR(N)):*START
AY(X=Ø AND MID$(I$,M,1)=STR$(J)):
```

**4.** Colon – Every instruction must have a colon.

**5.** Object – The last part of an instruction is the object. Everything following the colon is called the object of the instruction. An object is optional with some instructions.

### The Turtle PILOT Instructions

There are 11 instructions in Turtle PILOT, not including the turtle graphics commands. Below is an explanation of each instruction. At the end of each explanation are a few samples of the instruction.

*T: Type.* The Type instruction will print everything in the object on the screen. The Type instruction has an advantage over BASIC's PRINT command. With the Type instruction, words at the end of a line will not usually be divided between two lines of the screen. A string variable can be Typed by placing the name of the string variable in the object preceded and followed by a pound sign (#). An ampersand (&) placed at the end of the object will cause the next printed character to continue on the same line. The ampersand will not be displayed on the screen. A Type instruction without an object will print a blank line.

```
T:
T:HELLO,$NAME$ &
TY:YOUR TOTAL SCORE IS #S#.
```

*A: Accept.* The Accept instruction inputs a response from the user. The Accept instruction is very similar to BASIC's INPUT command, with the advantage that any character can be typed without an error occurring. An object is not required, but the input can be assigned to a variable by placing the variable name in the object.

```
A:
A:NAME$
A(M(K)=Ø):X
```

*M: Match.* The Match instruction compares all of the items in the object to the last response. The items in the object are separated by commas and may include string variable names. If any of the items match with the last response, the Y conditioner is set. Otherwise, the N conditioner is set. The Match instruction does not compare the item with just the start of the last response. It searches for the word through the entire response. For instance, GO would Match with GOING, INGOT, and LINGO. You may put up to 25 items in the object.

```
M:HI
M:A,E,I,O,U,Y,A$,B$
MN(Z<2Ø):END,STOP
```

*J: Jump.* The Jump instruction causes a branch to the line with the label that matches the label in the object of the Jump instruction. This instruction resembles BASIC's GOTO command, except a label is used instead of a line number.

```
J:*START
JY:*PART TWO
```

*U: Use.* The Use instruction is for Using modules (subroutines) in a PILOT program. It is similar to the Jump instruction, but the computer remembers the line from which it came. Program execution will continue at the line with a label that matches the label in the object until an End instruction is encountered. The End instruction will cause a return to the line following the Use instruction. The Use instruction is like BASIC's GOSUB command, with labels used instead of line numbers.

```
U:*PRINT
UY:*FIRST
```

*E: End.* The End instruction will terminate the program unless a Use instruction has been executed. If a Use instruction has been executed, program execution will continue at the line following the Use instruction. No object is used with an End instruction. The End instruction is similar to BASIC's RETURN and END commands.

```
E:
E(N=T):
```

*R: Remark.* The Remark instruction is not executed and is used only for program documentation.

```
R:THIS IS A REMARK
```

*C: Compute.* The Compute instruction may be used for numeric calculations or string manipulation.

```
C:N=N+1
C:S(K)=SIN(A*1Ø)
CY:A$="ABC"
C(T=1):Z$(N,1)=X$+RIGHT$(I$,3)
```

*B: Basic.* The object of the Basic instruction may contain any Applesoft commands.

```
B:HOME
B:GET K$:PRINT K$;
BY:HPLOT 1Ø,Y TO X,5Ø
BN(V>3):COLOR=2: HLIN 1Ø,2Ø AT Y
```

*S: Sound.* The object of the Sound instruction should contain a number from 1-31 for the pitch of the note, a comma, and a number from 1-255 for the duration. The notes range from C below middle C for a pitch value of 1 to F# above C above middle C for a value of 31. These values are the same as the pitch values used in Atari PILOT.

```
S:1Ø,2ØØ
S:3Ø,D
SY:P,SQR(L*2)
```

*G: Graphics.* The Graphics instruction precedes all turtle graphics commands, which will be explained next month in Part II.

### Variables In Turtle PILOT

You may use any variable in a Turtle PILOT program that you could use in an Applesoft program, except variables beginning with Q. The reason for not using Q variables will be explained next month in Part II of this article. You may also use any of Applesoft's mathematical and string functions.

### Using The Editor

The Editor has 11 commands to help you in typing Turtle PILOT programs. The Editor has two modes. First there is the command mode, indicated by a prompt. Any of the Editor's 11 commands may be typed in the command mode. The second mode is the program mode, indicated by a line number that is automatically printed on the left side of the screen. The program mode may be entered through several of the Editor commands. PILOT programs are typed in the program mode. To return to the command mode from the program mode, press RETURN without typing anything. When you have finished using the Editor, press the ESC key. Accidentally pressing the ESC key can be corrected by pressing RETURN immediately. Following is a description of each of the Editor commands.

The ADD command is used for entering the program mode to start or continue a program. It will ADD lines to the end of the program. The ADD command may also be followed by a line number. The lines ADDed to the program will then start at the line number specified. All the lines of the program currently in memory from that line to the end of the program will be erased.

The LIST command will LIST the program in memory. To LIST a single line, type the line number following the LIST command. A range of lines can be LISTed by typing the first and last line numbers separated by a comma. All the lines of a module may be LISTed by specifying the label of the module. Pressing RETURN will abort a LISTing.

The EDIT command is used to change line(s) of a program. It may be followed by a single line number or a range of line numbers with the first and last line numbers separated by a comma. The line to be EDITed will appear on the screen with the cursor at the beginning of the line.

The INSERT command is used for adding a line in the interior of a program. The INSERT command must be followed by a line number. No lines of the program will be deleted, but the numbers of the lines after the INSERTed line will be raised by one.

The DELete command is used for erasing line(s) from a program. A single line, a range of lines, or a label of a module may be specified. The numbers of the lines following the DELeted lines will be lowered.

The NEW command erases the program in memory.

The LOAD command will read a PILOT program from the disk and append it to the program in memory. The NEW command must be used first if the program in memory is not wanted. The name of the program is specified by following the LOAD command by that name.

The SAVE command will store the program in memory on the disk under the name specified. A program must be SAVEd before it can be translated.

The MEM command prints the number of free bytes available.

The CATalog command will print a catalog of the files on the disk. Turtle PILOT programs will appear as text files with ".P" at the end of their names.

The PR# command changes output to the specified slot, allowing a printer to be used with the Editor.

### Using The Translator

The Translator is used to translate your PILOT programs into Applesoft programs. When you RUN the Translator, it will ask you to type the name of the program to be translated. Make sure the program to be translated has been SAVEd on disk. After you have typed the name, the translating will automatically be done. The translated Applesoft program will be sent to a text file on the disk that can be EXECuted later to load it into memory. The computer will tell you what to type to load the program into memory. Once it is in memory, you can RUN, SAVE, LOAD, and LIST it just like a normal Applesoft program.

### Errors

Major syntax errors will be caught by the Editor, and the Editor will have you retype the line. Most other errors will be detected by the Translator while the program is being translated. If an error occurs during a translation, the computer will print "ERROR IN PILOT LINE NO." followed by the number of the line in which the error occurred. The computer will then automatically RUN the Editor. You should LOAD the program you were translating, correct the error, SAVE the program, and RUN the Translator again. Since this is a time consuming process, you should look the program over carefully before trying to translate. If an error occurs while a translated program is RUNning, divide the line number by ten to calculate the corresponding PILOT line number.

### Typing The Programs

A typing error in the Translator program could produce disastrous results. To make it easier to find mistakes, the following line should be included when the Translator is first typed.

```
15 POKE 216,0: GOTO 30
```

This will cause the translated program to be sent to the screen instead of the disk, and error messages will be printed normally. When all mistakes have been corrected, delete line 15.

The turtle PILOT Editor must be SAVEd under the name EDITOR, and the Turtle PILOT Translator must be SAVEd under the name TRANSLATOR. Make sure you SAVE the Translator immediately after typing it and before you RUN it, since it erases itself when it is finished.

This is only the first in a series of three articles about Turtle PILOT. In Part II we'll cover turtle graphics, which is probably the best feature of Turtle PILOT, and we'll take a look at some other features of the language. In Part III we'll convert an Atari PILOT program to Turtle PILOT. If you are already familiar with turtle graphics, experiment with the commands listed in line 10230 of the Translator program and see what you can discover.

*The author has offered to make a copy of the programs for you. Send a blank disk (specify DOS 3.2 or 3.3), a stamped, self-addressed mailer, and a $3.00 copying fee to:*

*Alan Poole*
*4728 King Road*
*Loomis, CA 95650*

**Program 1.**

```
1 REM  TURTLE PILOT TRANSLATOR
2 REM  BY ALLAN POOLE
1Ø GOSUB 1ØØØØ
17 REM
18 REM *** OPEN EXEC FILE ***
19 REM
2Ø N$ = LEFT$ (N$,LEN (N$) - 1) + "EXEC": PRI
    NT D$"OPEN"N$: PRINT D$"DELETE"N$
21 PRINT D$"OPEN"N$: PRINT D$"WRITE"N$
3Ø POKE 33,3Ø: LIST 5ØØØØ - 6ØØØØ: POKE 33,4Ø

4Ø PRINT "5 GOSUB 5ØØØØ"
47 REM
48 REM *** MAIN LOOP ***
49 REM
5Ø FOR LN = 1 TO NL:LN$ = ""
6Ø LN$ = LN$ + STR$ (LN * 1Ø)
7Ø I = Ø: FOR L = 1 TO 12: IF LEFT$ (P$(LN),1
    ) = I$(L) THEN I = L
8Ø NEXT : IF I = Ø THEN 15ØØØ
9Ø GOSUB 2ØØ: GOSUB 3ØØ: GOSUB 4ØØ
1ØØ ON I GOSUB 5ØØ,1ØØØ,15ØØ,2ØØØ,25ØØ,3ØØØ,35
    ØØ,4ØØØ,45ØØ,5ØØØ,6ØØØ,65ØØ
11Ø PRINT LN$: NEXT
117 REM
118 REM *** END OF PROGRAM ***
119 REM
12Ø PRINTLN*1Ø;"END":PRINT "?";CHR$ (34);"YOUR ~
    TRANSLATED PROGRAM IS IN MEMORY";CHR$
    (34)
121 PRINT D$"CLOSE"
13Ø PRINT : PRINT "TO LOAD YOUR TRANSLATED PRO
    GRAM INTO":
131 PRINT "MEMORY,TYPE "CHR$ (34);"EXEC "N$; C
    HR$ (34)
14Ø NEW
197 REM
198 REM *** SPLIT PILOT LINE AT COLON ***
199 REM
2ØØ FOR L = 1 TO LEN (P$(LN)): IF MID$ (P$(LN)
    ,L,1) = ":" THEN T = L:L = 3ØØ
21Ø NEXT
215 IF LEFT$ (P$(LN),1) = "*" THEN L$ = "*":R$
    = P$(LN): RETURN
22Ø L$ = LEFT$ (P$(LN),T - 1): IF T = LEN (P$(
    LN)) THEN R$ = "": RETURN
23Ø R$ = RIGHT$ (P$(LN), LEN (P$(LN)) - T)
24Ø T$ = L$: GOSUB 11ØØØ:L$ = T$
25Ø IF LEFT$ (L$,1) = "G" THEN T$ = R$: GOSUB ~
    11ØØØ:R$ = T$
26Ø RETURN
297 REM
298 REM *** FIND CONDITIONER ***
299 REM
3ØØ C = Ø: IF LEN (L$) < 2 THEN RETURN
31Ø IF MID$ (L$,2,1) = "Y" THEN LN$ = LN$ + "I
    F QC=1 THEN ":C = 1
32Ø IF MID$ (L$,2,1) = "N" THEN LN$ = LN$ + "I
    F QC=Ø THEN ":C = 2
33Ø RETURN
397 REM
398 REM *** FIND EXPRESSION ***
399 REM
4ØØ EX$ = "": IF RIGHT$ (L$,1) < > ")" THEN RE
    TURN
41Ø T = Ø: FOR L = 1 TO LEN (L$) - 1: IF MID$ ~
    (L$,L,1) = "(" THEN T = L:L =3ØØ
42Ø NEXT :EX$ = MID$ (L$,T + 1, LEN (L$) - T -
    1):LN$ = LN$ + "IF"+EX$ + "THEN"
43Ø RETURN
497 REM
498 REM *** T: INSTRUCTION ***
499 REM
5ØØ LN$ = LN$ + "QT$=" + CHR$ (34): IF R$ = ""
    THEN LN$ = LN$ + CHR$ (34) + "
5Ø1 GOSUB 51ØØØ": RETURN
51Ø FOR L = 1 TO LEN (R$):T$ = MID$ (R$,L,1)
52Ø IF T$ = "$" THEN 6ØØ
53Ø IF T$ = "#" THEN 7ØØ
54Ø LN$ = LN$ + T$
55Ø NEXT :LN$ = LN$ + CHR$ (34) + ":GOSUB 51ØØ
    Ø": RETURN
6ØØ IF L > LEN (R$) - 2 THEN 54Ø
61Ø T = Ø: FOR L1 = L + 2 TO LEN (R$): IF MID$
    (R$,L1,1) = "$" THEN T=L1:L1=3ØØ
62Ø NEXT : IF T = Ø THEN 54Ø
63Ø LN$ = LN$ + CHR$ (34) + "+" + MID$(R$,L + ~
    1,T - L) + "+" + CHR$ (34)
631 L = T: GOTO 55Ø
7ØØ IF L > LEN (R$) - 2 THEN 54Ø
71Ø T = Ø: FOR L1 = L + 2 TO LEN (R$): IF MID$
    (R$,L1,1)="#"THEN T=L1:L1 = 3ØØ
72Ø NEXT : IF T = Ø  THEN 54Ø
73Ø LN$=LN$+CHR$(34)+"+STR$(" + MID$ (R$,L + 1
    ,T - L - 1) + ")" + "+" +CHR$(34)
731 L = T: GOTO 55Ø
997 REM
998 REM *** A: INSTRUCTION ***
999 REM
1ØØØ LN$ = LN$ + "GOSUB 52ØØØ"          1Ø1Ø IF
    R$ = "" THEN RETURN
1Ø2Ø IF RIGHT$ (R$,1) = "$" THEN LN$ = LN$ + ":
    " + R$ + "=QI$": RETURN
1Ø3Ø LN$ = LN$ + ":" + R$ + "=VAL(QI$)": RETURN

1497 REM
1498 REM *** M: INSTRUCTION ***
1499 REM
15ØØ FOR L = 1 TO 25:M$(L) = "": NEXT : IF R$ =
    "" THEN 15ØØØ
151Ø T=1:FOR L=1 TO LEN(R$):IF MID$(R$,L,1)< > ~
    ","THEN M$(T)=M$(T)+MID$(R$,L,1)
1511 GOTO 153Ø
```

```
1520 T = T + 1
1530 NEXT
1540 FOR L = 1 TO T
1541 IF RIGHT$(M$(L),1)="$"THEN LN$=LN$+"Q$(" +
     STR$ (L) + ")=" + M$(L)+":"
1542 GOTO 1560
1550 LN$ = LN$ + "Q$(" + STR$ (L) + ")=" + CHR$
     (34) + M$(L) + CHR$ (34) + ":"
1560 NEXT :LN$ = LN$ + "GOSUB 53000": RETURN
1997 REM
1998 REM *** J: INSTRUCTION ***
1999 REM
2000 IF R$ = "" THEN 2100
2010 IF LEFT$ (R$,1) < > "*" THEN R$ = "*" + R$

2020 T = 0: FOR L = 1 TO NL: IF P$(L) = R$ THEN
     T = L:L = 2500
2030 NEXT : IF T = 0 THEN 15000
2040 LN$ = LN$ + "GOTO" + STR$ (T * 10): RETURN

2100 T = 0: FOR L = LN TO 1 STEP - 1: IF LEFT$ ~
     (P$(L),1) = "A" THEN T = L:L = 0
2110 NEXT : IF T = 0 THEN 15000
2120 GOTO 2040
2497 REM
2498 REM *** U: INSTRUCTION ***
2499 REM
2500 LN$ = LN$ + "QU=QU+1": IF LEFT$ (R$,1) < >
     "*" THEN R$ = "*" + R$
2510 T = 0: FOR L = 1 TO NL: IF P$(L) = R$ THEN
     T = L:L = 2500
2520 NEXT : IF T = 0 THEN 15000
2530 LN$ = LN$ + ":GOSUB" + STR$ (T * 10): RETU
     RN
2997 REM
2998 REM *** E: INSTRUCTION ***
2999 REM
3000 LN$ = LN$ + "IF QU=0 THEN END"
3010 PRINT LN * 10 + 5;
3020 IF C=1 THEN PRINT "IF QC=1 THEN";  3030 IF
     C=2 THEN PRINT "IF QC=0 THEN";
3040 IF EX$ < > "" THEN PRINT "IF";EX$;"THEN";
3050 PRINT "QU=QU-1:RETURN": RETURN
3497 REM
3498 REM *** C: INSTRUCTION ***
3499 REM
3500 LN$ = LN$ + R$: RETURN
3997 REM
3998 REM *** R: INSTRUCTION ***
3999 REM
4000 RETURN
4497 REM
4498 REM *** S: INSTRUCTION ***
4499 REM
4500 T = 0: FOR L = 1 TO LEN (R$): IF MID$ (R$,
     L,1) = "," THEN T = L:L = 255
4510 NEXT :LN$ = LN$ + "POKE 768,QS(" + LEFT$ (
     R$,T - 1) + ")
4511 POKE 769," + RIGHT$ (R$, LN (R$) - T) + ":
     CALL 770": RETURN
4997 REM
4998 REM *** G: INSTRUCTION ***
4999 REM
5000 IF R$ = "" THEN LN$ = LN$ + "POKE -16304,0
     :POKE -16297,0": RETURN
5009 REM     FIND LOOPS
5010 F = 0:IF VAL(R$) > 0 THEN LN$=LN$+ "FOR Q1
     =1 TO"+STR$( VAL (R$)) + ":":F=1
5011 R$ = LEFT$ (R$, LEN (R$) - 1)
5012 R$ = RIGHT$ (R$, LEN (R$) - LEN (STR$ ( VA
     L (R$))) - 1)
5019 REM     FIND INDIVIDUAL COMMANDS
5020 FOR L = 1 TO 6:GL$(L) = "": NEXT
5030 T = 1: FOR L = 1 TO LEN (R$)
5040 IF MID$ (R$,L,1) < > ";" THEN GL$(T) =GL$(
     T) + MID$ (R$,L,1): GOTO 5060
5050 T = T + 1
5060 NEXT
5069 REM     TRANSLATE EACH COMMAND
5070 FOR L = 1 TO T
5080 GC = 0:FOR L1=1 TO 11:IF LEFT$ (GL$(L), LE
     N (G$(L1))) < > G$(L1) THEN 5110
5090 GC = L1:L1 = 11: IF GL$(L) = G$(GC) THEN G
     L$(L) = "": GOTO 5110
5100 GL$(L) = RIGHT$ (GL$(L), LEN (GL$(L)) - LE
     N (G$(GC)))
5110 NEXT
5120 IF GC = 0 THEN 15000
5130 ON GC GOSUB 5200,5250,5300,5350,5400,5450,
     5500,5550,5600,5650,5700
5140 LN$ = LN$ + ":": NEXT : IF F = 1 THEN LN$ ~
     = LN$ + "NEXT"
5150 RETURN
5199 REM     CLEAR COMMAND
5200 LN$ = LN$ + "HGR": RETURN
5249 REM     TURNTO COMMAND
5250 LN$ = LN$ + "QA=90-" + GL$(L): RETURN
5299 REM     TURN COMMAND
5300 LN$ = LN$ + "QT=" + GL$(L) + ":GOSUB 54000
     ": RETURN
5349 REM     DRAW COMMAND
5350 LN$ = LN$ + "QL=" + GL$(L) + ":GOSUB 55000
     ": RETURN
5399 REM     PEN COMMAND
5400 T$ = GL$(L): IF T$ = "UP" THEN LN$ = LN$ +
     "QP=1": RETURN
5402 IF T$ = "DOWN" THEN LN$ = LN$ + "QP=0":RET
     URN
5405 LN$ = LN$ + "HCOLOR=": IF T$ = "ERASE" THE
     N LN$ = LN$ + "QB"
5410 IF T$ = "BLACK" THEN LN$ = LN$ + "0"
5415 IF T$ = "GREEN" THEN LN$ = LN$ + "1"
5420 IF T$ = "VIOLET" THEN LN$ = LN$ + "2"
5425 IF T$ = "WHITE" THEN LN$ = LN$ + "3"
5430 IF T$ = "BLACK2" THEN LN$ = LN$ + "4"
5435 IF T$ = "RED" THEN LN$ = LN$ + "5"
5440 IF T$ = "BLUE" THEN LN$ = LN$ + "6"
5445 IF T$ = WHITE2" THEN LN$ = LN$ + "7"
5448 RETURN
5449 REM     SCREEN COMMAND
5450 GOSUB 5400:LN$ = LN$ + ":HPLOT 0,0:CALL 62
     454:QB=" + RIGHT$(LN$,1): RETURN
5499 REM     GOTO COMMAND
5500 T = 0: FOR L1 = 1 TO LEN (GL$(L)): IF MID$
     (GL$(L),L1,1) = "," THEN T = L1
5501 L1 = 255
5510 NEXT :LN$ = LN$ + "QX=" + LEFT$ (GL$(L),T ~
     - 1) + "
5511 QY=" + RIGHT$ (GL$(L), LEN (GL$(L)) - T): ~
     RETURN
5549 REM     FULL COMMAND
5550 LN$ = LN$ + "POKE -16302,0": RETURN
5599 REM     MIX COMMAND
5600 LN$ = LN$ + "POKE-16301,0": RETURN
5649 REM     QUIT COMMAND
5650 LN$ = LN$ + "TEXT": RETURN
5699 REM     GO COMMAND
5700 LN$ = LN$ + "QP=1:QL=" + GL$(L) + ":GOSUB ~
     55000:QP=0": RETURN
5997 REM
5998 REM *** B: COMMAND ***

5999 REM
6000 LN$ = LN$ + R$: RETURN
6497 REM
6498 REM *** LABEL ***
6499 REM
6500 LN$ = LN$ + "REM" + R$: RETURN
9997 REM
9998 REM *** INITIALIZE ***
9999 REM
```

```
10000 TEXT : HOME
10010 HTAB 6: INVERSE : PRINT "                            ~
      "
10020 HTAB 6: PRINT " ";: HTAB 34: PRINT " "
10030 HTAB 6: PRINT " ";: HTAB 9: NORMAL : PRINT
      "TURTLE PILOT TRANSLATOR";
10031 HTAB 34: INVERSE : PRINT " "
10040 HTAB 6: PRINT " ";: HTAB 34: PRINT " "
10050 HTAB 6: PRINT " ";: HTAB 14: NORMAL : PRIN
    T "BY ALAN POOLE";
10051 HTAB 34: INVERSE : PRINT " "
10060 HTAB 6: PRINT " ";: HTAB 34: PRINT " "
10070 HTAB 6: PRINT "                            ~
      ": NORMAL
10080 DIM P$(2500),I$(12),G$(11),GL$(6),M$(25):D
    $ = CHR$ (4)
10090 PRINT : INPUT "WHAT IS THE NAME OF THE PRO
    GRAM? ";N$: IF N$ =""THEN 10090
10100 IF RIGHT$ (N$,2) < > ".P" THEN N$ = N$ + "
    .P"
10105 PRINT : PRINT "PLEASE WAIT..."
10110 ONERR GOTO 16000
10120 PRINT D$"VERIFY"N$: PRINT D$"OPEN"N$: PRIN
    T D$"READ"N$
10130 INPUT NL
10140 FOR L = 1 TO NL:I$ = ""
10150 GET K$: IF ASC (K$) = 13 THEN P$(L) = I$: ~
    GOTO 10170
10160 I$ = I$ + K$: GOTO 10150
10170 NEXT : PRINT : PRINT D$"CLOSE"
10180 FOR L = 1 TO 12: READ I$(L): NEXT
10190 FOR L = 1 TO 11: READ G$(L): NEXT
10200 ONERR GOTO 15000
10210 RETURN
10220 DATA T,A,M,J,U,E,C,R,S,G,B,*
10230 DATA CLEAR,TURNTO,TURN,DRAW,PEN,SCREEN,GOT
    O,FULL,MIX,QUIT,GO
10997 REM
10998 REM *** REMOVE SPACES FROM T$ ***
10999 REM
11000 IF T$ = "" THEN RETURN
11010 T1$ = "": FOR L = 1 TO LEN (T$)
11020 IF MID$ (T$,L,1) < > " " THEN T1$ = T1$ + ~
    MID$ (T$,L,1)
11030 NEXT :T$ = T1$: RETURN
14997 REM
14998 REM *** ERROR ROUTINES ***
14999 REM
15000 PRINT D$"CLOSE": PRINT "ERROR IN PILOT LIN
    E NO. ";LN; CHR$ (7)
15001 PRINT D$"RUN EDITOR"
16000 PRINT D$"CLOSE": PRINT "UNABLE TO LOAD"; C
    HR$ (7) : RUN
49996 REM
49997 REM THE FOLLOWING LINES ARE NOT PART OF TH
    E TRANSLATOR, BUT ARE
49998 REM INCLUDED IN EVERY TRANSLATED PROGRAM.
49999 REM
50000 DIM Q$(25),QS(31)
50010 HCOLOR= 3:QX = 0:QY = 0:QC = - 1:QR = 40:Q
    A = 90:QQ = 3.1415927 / 180
50020 QS(1) = 192:QS(2) = 180: QS(3) = 171:QS(4)
     = 161:QS(5) = 153:QS(6) = 144
50021 QS(7) = 136:QS(8) = 129:QS(9) = 122:QS(10)
     = 115:QS(11) = 108:QS(12) =102
50022 QS(13) = 96:QS(14) = 91:QS(15) = 86
50025 QS(16) = 81:QS(17) = 76:QS(18) = 72:QS(19)
     = 68:QS(20) = 64:QS(21) = 60
50026 QS(22) = 57:QS(23) = 54:QS(24) = 50:QS(25)
     = 47:QS(26) = 45:QS(27) = 42
50027 QS(28) = 40:QS(29) = 37:QS(30) = 35:QS(31)
     = 33
50030 POKE 770,173: POKE 771,48: POKE 772,192: P
    OKE 773,136: POKE 774,208
50031 POKE 775,5: POKE 776,206: POKE 777,1: POKE
     778,3: POKE 779,240
50032 POKE 780,9: POKE 781,202: POKE 782,208: PO
    KE 783,245: POKE 784,174
50040 POKE 785,0: POKE 786,3: POKE 787,76: POKE ~
    788,2: POKE 789,3: POKE 790,96
50041 POKE 791,0: POKE 792,0
50050 RETURN
51000 IF QT$ = "" THEN PRINT : RETURN
51005 QT = 0: IF RIGHT$ (QT$,1) = "&" THEN QT$ =
     LEFT$(QT$, LEN (QT$) - 1):QT=1
51010 FOR Q1 = 1 TO LEN (QT$)
51011 IF MID$ (QT$,Q1,1) = " " AND PEEK (36) > Q
    R - 9 THEN GOSUB 51100
51020 PRINT MID$ (QT$,Q1,1);: NEXT : IF QT = 0 T
    HEN PRINT
51030 RETURN
51100 QF = 0: FOR Q2 = Q1 + 1 TO Q1 + QR - PEEK ~
    (36) - 1
51101 IF Q2 > = LEN (QT$) THEN Q2 = 1000:QF = 1:
     GOTO 51120
51110 IF MID$ (QT$,Q2,1) = " " THEN Q2 = 1000:QF
     = 1
51120 NEXT : IF QF = :0 THEN PRINT :Q1 = Q1 + 1
51130 RETURN
52000 QI$ = ""
52010 GET QK$:QT = ASC (QK$): PRINT QK$;
52020 IF QT = 13 THEN RETURN
52030 IF QT = 8 THEN 52100
52040 IF QT = 21 THEN 52200
52050 IF QT = 24 THEN PRINT CHR$ (92): GOTO 5200
    0
52060 QI$ = QI$ + QK$: IF LEN (QI$) > 245 THEN P
    RINT CHR$ (7);
52070 IF LEN (QI$) > 250 THEN PRINT CHR$ (92): G
    OTO 52000
52080 GOTO 52010
52100 IF QI$ = "" THEN PRINT : GOTO 52010
52110 IF LEN (QI$) = 1 THEN 52000
52120 QI$ = LEFT$ (QI$, LEN (QI$) -1): GOTO 5201
    0
52200 QI$ = QI$ + CHR$ (PEEK ( PEEK (40) + PEEK ~
    (41) * 256 + PEEK (36)) - 128)
52210 POKE 36, PEEK (36) + 1: IF PEEK (36) > QR ~
    - 1 THEN PRINT
52220 GOTO 52010
53000 QM = 0:QC = 0: FOR Q1 = 1 TO 25
53001 IF LEN (QI$) < LN (Q$(Q1)) OR Q$ (Q1) = ""
     THEN 53030
53010 FOR Q2 = 1 TO LEN (QI$) - LEN (Q$(Q1)) + 1

53011 IF Q$(Q1) = MID$ (QI$,Q2, LEN (Q$(Q1))) TH
    EN QC = 1:QM = Q1:Q1=25:Q2 =300
53020 NEXT
53030 NEXT : FOR Q1 = 1 TO 25:Q$(Q1) = "": NEXT ~
    : RETURN
54000 QA = QA - QT: IF QA > 360 THEN QA = QA - 3
    60
54010 IF QA < 0 THEN QA = QA + 360
54020 RETURN
55000 IF QP = 1 THEN 55005
55003 HPLOT QX + 139.0005, - QY + 80.0005
55005 QX = QX + QL * COS (QA * QQ):QY = QY + QL ~
    * SIN (QA * QQ)
55010 IF QX < - 139 THEN QX = - 139
55020 IF QX > 140 THEN QX = 140
55030 IF QY < - 111 THEN QY = - 111
55040 IF QY > 80 THEN QY = 80
55050 IF QP = 1 THEN RETURN
55060 HPLOT TO QX + 139.0005, - QY + 80.0005: RE
    TURN
```

**Program 2.**

```
1 REM TURTLE PILOT EDITOR
2 REM BY ALAN POOLE
```

```
10 GOSUB 20000
17 REM
18 REM *** MAIN LOOP ***
19 REM
20 PRINT : PRINT "<";: GOSUB 100: GOSUB 12000
   : IF I$ = "" THEN 20
25 IF LEFT$ (I$,3) = "CAT" THEN PRINT D$"CATA
   LOG": GOTO 20
28 IF LEFT$ (I$,3) = "PR#" THEN PRINT D$"PR#"
    VAL ( RIGHT$ (I$,1)): GOTO 20
30 GOSUB 500: IF C = 0 THEN PRINT SE$: GOTO 2
   0
40 ON C GOSUB 1000,2000,3000,4000,5000,6000,7
   000,8000,9000
50 GOTO 20
97 REM
98 REM *** INPUT ***
99 REM
100 I$ = ""
110 GET K$:K = ASC (K$): PRINT K$;
120 IF K = 13 THEN RETURN : REM RETURN KEY
130 IF K = 8 THEN 200: REM BACKSPACE
140 IF K = 21 THEN 300: REM RETYPE
150 IF K = 24 THEN PRINT CHR$ (92): GOTO 100: ~
    REM CTRL-X
160 IF K = 27 THEN 15000: RM ESC
170 I$ = I$ + K$: IF LEN (I$) > 195 THEN PRINT
     BELL$;
180 IF LEN (I$) > 200 THEN PRINT CHR$ (92): GO
    TO 100
190 GOTO 110
197 REM
198 REM *** BACKSPACE ***
200 IF I$ = "" THEN PRINT : GOTO 110
210 IF LEN (I$) = 1 THEN 100
220 I$ = LEFT$ (I$, LEN (I$) - 1): GOTO 110
297 REM
298 REM *** RETYPE ***
299 REM
300 I$ = I$ + CHR$ ( PEEK ( PEEK (40) + PEEK (
    41) * 256 + PEEK (36)) - 128)
310 POKE 36, PEEK (36) + 1: IF PEEK (36) > 39 ~
    THEN PRINT
320 GOTO 110
497 REM
498 REM *** DETRMINE NO. OF EDITOR COMMAND ***
499 REM
500 C = 0: FOR L = 1 TO 9: IF C$(L) = LEFT$ (I
    $, LEN (C$(L))) THEN C = L
510 NEXT : IF C = 0 THEN RETURN
519 REM MAKE I$ THE PART RIGHT OF COMMAND
520 IF I$ = C$(C) THEN I$ = "": RETURN
530 I$ = RIGHT$ (I$, LEN (I$) - LEN (C$(C))): ~
    GOSUB 12000: RETURN
997 REM
998 REM *** ADD ***
999 REM
1000 IF I$ = "" THEN 1030
1010 IF VAL (I$) > LN OR VAL (I$) < 1 THEN PRIN
     T RE$: RETURN
1020 LN = VAL (I$)
1030 IF LN > 2499 THEN PRINT "TOO MANY LINES";B
     ELL$: RETURN
1040 LT = LN + 1: GOSUB 10000:LN = LT:P$(LN) = ~
     I$: GOTO 1030
1997 REM
1998 REM *** LIST ***
1999 REM
2000 IF LN = 0 THEN PRINT : RETURN
2010 IF LEFT$ (I$,1) = "*" THEN GOSUB 13000: GO
     TO 2050
2020 IF I$ = "" THEN FL = 1:LL = LN: GOTO 2050
2030 GOSUB 11000
2040 IF FL < 1 OR FL > LN OR LL < 1 OR LL > LN ~
     THEN PRINT RE$: RETURN
2050 FOR L = FL TO LL: HTAB 5 - LEN (STR$ (L)):
      PRINT L;" ";
2051 IF LEFT$ (P$(L),1) = "*" THEN 2057
2055 PRINT " ";: IF LEFT$ (P$(L),1) < > "R" THE
     N PRINT " ":
2057 PRINT P$(L)
2060 IF PEEK ( - 16384) = 141 THEN L = 2500
2070 NEXT : RETURN
2997 REM
2998 REM *** EDIT ***
2999 REM
3000 IF I$ = "" THEN PRINT SE$: RETURN
3010 GOSUB 11000
3020 IF FL < 1 OR FL > LN OR LL < 1 OR LL > LN ~
     THEN PRINT RE$: RETURN
3030 HOME : FOR L1 = FL TO LL: VTAB 18: HTAB 5 ~
     - LEN ( STR$ (L1))
3031 PRINT L1;" ";P$(L1);: VTAB 18: HTAB 6: GOS
     UB 100: GOSUB 10010
3040 VTAB 24: FOR L2 = 1 TO LEN (P$(L1)) / 40 +
      2: PRINT : NEXT :P$(L1) = I$
3050 NEXT : RETURN
3997 REM
3998 REM *** INSERT ***
3999 REM
4000 LT = VAL (I$): IF LT < 1 OR LT> LN THEN PR
     INT RE$: RETURN
4010 IF LN > 2499 THEN PRINT "TOO MANY LINES";B
     ELL$: RETURN
4020 GOSUB 10000
4030 FOR L = LN + 1 TO LT STEP - 1: P$(L) = P$(
     L - 1): NEXT
4040 P$(LT) = I$: LN = LN + 1: RETURN
4997 REM
4998 REM *** DELETE ***
4999 REM
5000 IF I$ = "" THEN PRINT SE$: RETURN
5010 IF LEFT$ (I$,1) = "*" THEN GOSUB 13000: GO
     TO 5030
5020 GOSUB 11000: IF FL < 1 OR FL > LN OR LL<1 ~
     OR LL> LN THEN PRINT RE$: RETURN
5030 FOR L = FL TO LN - (LL - FL + 1):P$(L) = P
     $(L + (LL - FL + 1)): NEXT
5031 LN = LN - (LL - FL + 1)
5040 RETURN
5997 REM
5998 REM *** NEW ***
5999 REM
6000 FOR L = 1 TO LN:P$(L) = "": NEXT :LN = 0: ~
     RETURN
6997 REM
6998 REM *** LOAD ***
6999 REM
7000 N$ = I$: IF N$ = "" THEN PRINT SE$: RETURN
7010 ONERR GOTO 21000
7020 IF RIGHT$ (N$,2) < > ".P" THEN N$ = N$ + "
     .P"
7030 PRINT D$"VERIFY"N$: PRINT D$"OPEN"N$: PRIN
     T D$"READ"N$
7040 INPUT T
7050 FOR L = 1 TO T
7052 GET T$: IF T$ = CHR$ (13) THEN 7058
7055 P$(LN + L) = P$(LN + L) + T$: GOTO 7052
7058 NEXT : PRINT
7060 PRINT D$"CLOSE"
7070 POKE 216,0: REM RESET ONERR
7080 LN = LN + T: RETURN
7997 REM
7998 REM *** SAVE ***
7999 REM
8000 IF I$ = "" THEN PRINT SE$: RETURN
8003 ONERR GOTO 22000
8005 IF RIGHT$ (I$,2) < > ".P" THEN I$ = I$ + "
     .P"
8010 PRINT D$"OPEN"I$: PRINT D$"WRITE"I$
8030 PRINT LN
8040 FOR L = 1 TO LN: PRINT P$(L): NEXT
8050 PRINT D$"CLOSE"I$
8060 POKE 216,0: RETURN
```

```
8997 REM
8998 REM *** MEM ***
8999 REM
9000 PRINT "THERE ARE "; FRE (0) - 3500;" BYTES
     LEFT.": RETURN
9997 REM
9998 REM *** INPUT PROGRAM LINE ***
9999 REM
10000 HTAB 5 - LEN ( STR$ (LT)): PRINT LT;" ";:
     GOSUB 100: GOSUB 12000
10010 IF I$ = "" THEN POP : RETURN
10020 F = 0:P = 0: FOR L = 1 TO LEN (I$):T$ = MI
     D$ (I$,L,1): IF T$ = ":"THENF=1
10030 IF T$ = "(" THEN P = P + 1
10040 IF T$ = ")" THEN P = P - 1
10050 NEXT:IF (F = 0 AND LEFT$ (I$,1) < > "*")OR
      P< > 0THEN PRINT SE$:GOTO10000
10060 F = 0: FOR L = 1 TO 12: IF IN$(L) = LEFT$
     (I$,1) THEN F = 1:L = 12
10070 NEXT : IF F = 0 THEN PRINT SE$: GOTO 10000

10080 RETURN
10997 REM
10998 REM *** FIND TWO NOS. DIVIDED BY COMMA ***

10999 REM
11000 FL = VAL (I$):LL = 0: FOR L = 1 TO LEN (I$
     ) - 1
11001 IF MID$ (I$,L,1) = "," THEN I$ = RIGHT$ (I
     $, LEN (I$) - L):LL = VAL (I$)
11010 NEXT : IF LL = 0 THEN LL = FL
11997 REM
11998 REM *** REMOVE LEADING SPACES ***
11999 REM
12000 IF I$ = "" THEN RETURN
12010 IF LEFT$ (I$,1) < > " " THEN RETURN
12020 IF I$ = " " THEN I$ = "": RETURN
12030 I$ = RIGHT$ (I$, LEN (I$) - 1): GOTO 12000

12997 REM
12998 REM *** FIND FIRST AND LAST LINES OF A MOD
     ULE ***
12999 REM
13000 FOR L = 1 TO LN: IF P$(L) = I$ THEN FL = L
     : GOTO 13020
13010 NEXT : PRINT "LABEL NOT FOUND";BELL$: POP
     : RETURN
13020 LL = 0: FOR L = L TO LN: IF LEFT$ (P$(L),1
     ) = "E" THEN LL = L:L = 2500
13030 NEXT : IF LL = 0 THEN LL = LN
13040 RETURN
14997 REM
14998 REM *** END PROGRAM ***
14999 REM
15000 PRINT : PRINT "DO YOU WANT TO SAVE THE PRO
     GRAM? (Y/N)";: GET I$
15001 IF I$ = CHR$ (13) THEN PRINT : GOTO 100
15010 IF I$ < > "Y" AND I$ < > "N" THEN 15000
15015 PRINT I$
15020 IF I$ = "Y" THEN INPUT "WHAT IS THE NAME O
     F THE PROGRAM? ";I$: GOSUB 8000
15030 PRINT : PRINT "DO YOU WANT TO TRANSLATE A"
     : PRINT "PROGRAM?(Y/N)";:GETI$
15031 IF I$ < > "Y" AND $ < > "N" THEN 15030
15040 PRINT I$: IF I$ = "Y" THEN PRINT CHR$ (4)"
     RUN TRANSLATR"
15050 HOME : END
19997 REM
19998 REM *** INITIALIZE ***
19999 REM
20000 TEXT : HOME
20010 HTAB 8: INVERSE : PRINT "
               "
20020 HTAB 8: PRINT " ";: HTAB 32: PRINT " "
20030 HTAB 8: PRINT " ";: HTAB 11: NORMAL : PRIN
     T "TURTLE PILOT EDITOR";
20031 HTAB 32: INVERSE : PRINT " "
20040 HTAB 8: PRINT " ";: HTAB 32: PRINT " "
20050 HTAB 8: PRINT " ";: HTAB 13: NORMAL : PRIN
     T "BY ALAN POOLE";: HTAB 32
20051 INVERSE : PRINT " "
20060 HTAB 8: PRINT " ";: HTAB 32: PRINT " "
20070 HTAB 8: PRINT "                         ":
      NORMAL
20080 DIM P$(2500),C$(9),IN$(12)
20090 BELL$ = CHR$ (7):D$ = CHR$ (4):SE$ = "SYNT
     AX ERROR" + BELL$
20091 RE$ = "LINE NO. OUT OF RANGE" + BELL$
20100 FOR L = 1 TO 9: READ C$(L): NEXT
20110 FOR L = 1 TO 12: READ IN$(L): NEXT
20120 RETURN
20130 DATA ADD,LIST,EDIT,INSERT,DEL,NEW,LOAD,SAV
     E,MEM
20140 DATA T,A,M,J,U,E,C,R,S,G,B,*
20997 REM
20998 REM *** ERROR ROUTINES ***
20999 REM
21000 POKE 216,0: PRINT D$"CLOSE": PRINT "UNABLE
      TO LOAD";BELL$: GOTO 20
22000 POKE 216,0: PRINT D$"CLOSE": PRINT "DOS ER
     ROR";BELL$: GOTO 20
```

©

*A method for achieving statistical accuracy on microcomputer systems. These programs are written in a version of BASIC so general that they will work as is on Atari, PET, or nearly any other version of BASIC.*

## The Statistics Page

# Accurate Statistical Calculations

Dr. Allen H. Wolach
Department of Psychology
Illinois Institute of Technology

Most statistical programs use formulas that are called computational formulas. Consider the computational formula $\Sigma x_i^2 - N\bar{X}^2$ which is used to calculate the sum of squares term for many statistics. The portion of the formula that is designated $\Sigma x_i^2$ is obtained by squaring individual scores, and then summing the squared scores. The portion of the formula that is designated $N\bar{X}^2$ is obtained by multiplying the number of scores (N) by the square of the mean of the scores ($\bar{X}^2$).

If $\Sigma x_i^2$ is equal to 11178323747 and $N\bar{X}^2$ is equal to 11178304256.25, the sum of squares is 11178323747 minus 11178304256.25 or 19490.75. The sum of squares of 19490.75 is well within the nine digit capacity of most versions of BASIC in microcomputer systems. However, the intermediate calculations exceed the nine digit capacity of most microcomputer systems. If the microcomputer resorts to exponential notation, $\Sigma x_i^2$ becomes $111783237 \times 10^2$ and $N\bar{X}^2$ becomes $111783042 \times 10^2$. When the sum of squares is calculated $\Sigma x_i^2 - N\bar{X}^2$ becomes 19500. Note that the actual value for the sum of squares is 19490.75, not 19500.

Difference formulas can be used instead of computational formulas The difference formula for the sum of squares is $\Sigma(x_i - \bar{X})^2$. The difference formula requires subtracting the mean of all scores from an individual score and then squaring this difference. The squared differences for each of the scores are then summed. Difference formulas never produce intermediate results that are larger than the final value for the sum of squares.

The computational formula for the sum of squares will produce the same results as the difference formula provided that computations for the computational formula are carried to enough digits to accommodate intermediate calculations. On the other hand, the difference formula requires using the mean with each difference that is computed. In order to obtain the mean, one has two partially unsatisfactory options. All of the scores can be saved in memory. Then the mean can be calculated for use in each difference. Finally, the scores can be recalled individually from memory, and the mean can be subtracted from each score.

If the data for a given analysis consist of a relatively large number of scores, one can easily exceed the memory capability of a microcomputer system. A second option is to enter all of the scores twice. A mean can be calculated the first time the scores are entered. This mean can be used for forming the differences the second time the scores are entered. Although this procedure does not require saving the scores in memory, it doubles the number of scores that must be entered.

A procedure that is intermediate to the computational formula and difference formula procedures can be used. This approach can be called the provisional mean procedure. The procedure requires entering the data only one time and does not require saving individual scores.

In addition, the provisional mean procedure does not produce the large numbers that are generated by the computational formula. The provisional mean procedure involves calculating a difference for each score. This difference is not the difference between the score and the mean. It is the difference between the score and a provisional mean. The provisional mean changes as the successive scores are entered. Program 1 shows the use of the provisional mean procedure to calculate the mean, sum of squares, variance, and standard deviation for a set of data. A difference is calculated in statement 60. The mean and sum of squares are calculated in statements 70 and 80, respectively.

**Program 1.**

```
10 PRINT "ENTER NUMBER OF SCORES";
20 INPUT N
30 FOR I = 1 TO N
40 PRINT "ENTER SCORE";I
50 INPUT X
60 LET D = X - M
70 LET M = M + D/I
80 LET S = S + D*(X - M)
90 NEXT I
100 PRINT "MEAN = ";M
110 PRINT "SUM OF THE SQUARES = ";S
120 PRINT "VARIANCE = ";S/(N -1)
```

ECLECTIC SYSTEMS CORPORATION
Order TOLL FREE 1-800-527-3135

ECLECTIC SYSTEMS

P.O. Box 1166 • 16260 Midway Road
Addison, TX 75001 • (214) 661-1370


ANNOUNCES

# EM© for the SuperPET

## EM© is ANSI standard MUMPS

## Make your SuperPET Super-Load in EM© and Dramatically Improve your Programming Productivity

EM© is more than just a programming language. It is a well integrated data management system combining with one syntax what other operating systems would call 1) an application programming language; 2) a job control language; 3) a linkeage editor; 4) a database management system; and 5) a communications monitor.

**PROGRAM MANAGEMENT:**
EM© provides all programming management facilities needed to manage programs and program files. Programs can be created, edited, cataloged and debugged from within EM©. Programs can be as large as disk capacity. A resident algorithm rids memory of least frequently used variables and program modules so that what you need off-disk normally resides in memory.

**STRING POWER:**
EM© makes string handling easy with its extensive set of string operations and functions. Variable length strings can be used routinely without the obstacles presented by most other programming languages.

**PATTERN MATCHING:**
EM© can "filter" user input with a useful pattern matching that will result in fewer user or device errors. For example: dates, zip codes and names can be tested for validity with a single statement.

**GLOBALS:**
EM© obviates the need for traditional read and write operations on secondary storage devices by allowing data elements to be directly referenced as a set of subscripts; all the details of file organization and retrieval are handled by the system.

**TIMING:**
EM© enables a programmer to associate timing constraints with several operations. This feature allows testing for terminal malfunctions as well as prompting users in time-critical dialogue.

**DATA BASE MANAGEMENT:**
Sorts and merges are not necessary as EM© automatically stores data in a dynamically allocated balanced tree structure. Random access to any data item requires at most three disk reads.

**EM© UNMATCHED IN PROGRAMMING PRODUCTIVITY:**
System houses that program in EM© (MUMPS) find that their costs are lower than those of their competitors using other languages. Fewer lines of code are necessary per application. Dimension statements are not required. Subscripts may be alpha, numeric or any legal string. Data types need not be defined and can change freely throughout as EM© can recognize when it is dealing with alpha, numeric, integer or floating-point data types. EM© gives the professional programmer a full set of software tools designed for real-life tasks and problems he consistently encounters in the production and maintenance of application software. EM© adheres rigidly to ANSI MUMPS standards, which make it transportable to larger processors manufactured by DEC, TANDOM, DATA GENERAL, HARRIS and others. Additionally EM© gives the less-experienced programmer the tools to do a professional job on formidable programming applications.

**You may order EM© or SuperPET by calling ECLECTIC SYSTEMS toll-free at 1-800-527-3135 from 10 AM to 4 PM CDT Monday through Friday, or you can order by mail using the form below. Texas residents call 1-214-661-1370.**

ECLECTIC SYSTEMS CORPORATION
P.O. Box 1166, 16260 Midway Road, Addison, Texas 75001

Here's my order for EM©@ $299 plus $3.75 for shipping and handling (UPS surface unless specified otherwise). Residents of Texas, Louisiana, Oklahoma City and Tulsa, Oklahoma must add applicable taxes.

☐ My certified check or money order is enclosed.
☐ Please charge my VISA # ____________________ or
MasterCard # ____________________ Expiration date __________

Name ____________________ Signature ____________________

Address ____________________

City ____________ State ____________ Zip ____________

```
130 PRINT "STANDARD DEVIATION = ";SQR(S/(N ~
    -1 ))
140 END
```

Suppose that one wanted to calculate a Pearson product moment correlation coefficient. This correlation coefficient requires a sum of products term. The sum of products term can be calculated using modified provisional mean formulas. Program 2 is a BASIC program for calculating the Pearson product moment correlation coefficient.

**Program 2.**

```
10 PRINT "ENTER NUMBER OF PAIRS OF SCORES ~
    ";
20 INPUT N
30 FOR I = 1 TO N
40 PRINT "ENTER SCORES IN PAIR ";I
50 INPUT X, Y
60 LET D1 = X - M1
70 LET M1 = M1 + D1/I
80 LET S1 = S1 + D1*(X - M1)
90 LET D2 = Y - M2
100 LET M2 = M2 + D2/I
110 LET S2 = S2 + D2*(Y - 2)
120 LET D3 = SQR(ABS(D1))*SQR(ABS(D2))
130 LET D4 = (X - M1)*(Y - 2)
140 IF D4 < 0 THEN 170
150 LET G = 1
160 GOTO 180
170 LET G = - 1
180 LET S3 = S3 + G*D3*SQR(ABS(D4))
190 NEXT I
200 LET R = S3/(N*(SQR(S1/N))*SQR(S2/N))
210 PRINT "CORRELATION COEFFICIENT = ";R
220 END
```

Note that the mean of the X scores is calculated in statement 70, and the sum of squares for the X scores is calculated in statement 80. The mean for the Y scores is calculated in statement 100, and the sum of squares for the Y scores is calculated in statement 110. The sum of products term is calculated in statement 180.

Once sum of square and sum of products terms are mastered, accurate t-test and analysis of variance programs can be written for microcomputer systems. ©

A Monthly Feature

# Learning With Computers

Glenn Kleiman
Teaching Tools: Microcomputer Services
Palo Alto, CA

As another school year begins, the "old-timers" among users of personal computers can look back and remember the early days, fondly recalling their 8K PET computers with small keyboards and built-in cassette recorders, TRS-80 Model I computers with 4K of RAM memory, and early Apple II computers with cassette recorders and Integer BASIC. Those who were among the first to use personal computers in schools may be celebrating their fifth year of computers in their classrooms.

Personal computers in education have come a long way in five years. Floppy disk drives, dot matrix printers and color monitors are now common in many schools. Hard disks, network systems, letter quality printers and modems are found in some. Improved versions of BASIC have been developed and LOGO, PILOT, Pascal, FORTRAN, FORTH and other languages are available for some personal computers. Even the manuals have improved tremendously. How many of you recall the first manuals (in some cases, better called pamphlets) that provided all the available information about the early personal computers?

Most importantly, significant advances have been made in educational software. Gone are the days when any program that did not crash was considered acceptable. Educational software is now expected to be user friendly, to make good use of the interactive, graphics and sound capabilities of the computer, and to follow principles of effective pedagogy. Educators are beginning to expect not just single programs, but courseware packages – sets of programs combined with written aids for teachers and students.

In addition to computers' becoming a widespread tool for teaching and learning, computer science is joining chemistry, biology and physics as a standard part of the curriculum in many schools.

It has been a remarkable five years since the first completely assembled personal computers became available. Exciting advances continue to be made – video disks interfaced to computers, computer speech synthesis and speech recognition, computer aids for handicapped individuals, more powerful computer systems at lower prices, advances in computer graphics and music, color printers, more extensive information and communication systems, new languages and more.

Of course, it hasn't all been smooth sailing. All of us who ventured into the world of computers have experienced disappointing software, inadequate documentation, incompatible components, service and supply problems, and other difficulties. Some educators have run into difficulties using computers and, unable to get the support they needed from colleagues, computer dealers or others, became frustrated and gave up. There are classrooms in which a computer sits unused in the closet. But there are many more classrooms in which teachers and students are eager to learn about and use computers.

### A Practical Guide To Computers In Education

A book designed to help teachers get started using computers has recently been published. It is called *Practical Guide to Computers in Education* and is written by Peter Colburn, Peter Kelman, Nancy Roberts, Thomas F. F. Snyder, Daniel H. Watt and Cheryl Weiner (Addison-Wesley, $9.95). A practical guide is much needed, and this one was produced by a group of experienced educators, computer users and writers.

The *Practical Guide to Computers in Education* contains eight chapters plus sections describing the different brands of computers, a list of suggested readings, an extensive list of resources for educators and a short glossary. It is a very practical book, containing descriptions of experiences of many educators who are already using computers.

Chapter 1 is entitled "The Computer Goes to School." It raises the important question: "Will computers transform the schools?" The authors present both positive and negative arguments. The positive arguments point to the diverse capabilities of computers and their currently successful use in schools. The negative arguments focus on the problems that some have experienced with computers and on the fact that many educators do not have sufficient training to make good use of computers. Those who take the negative position typically view computers as the latest educational fad, likely to soon go the way of the New Math.

The authors of *Practical Guide to Computers in Education* try to present a balanced perspective. They don't believe computers can replace teachers or make schools obsolete, but they are certain that the widespread use of computers in society will affect schools. They have mixed feelings about

# Recreational Computing Back Issues

*Recreational Computing* was the first and only personal computing magazine when it started in 1972 (it was called the *PCC Newspaper* back then). Bob Albrecht, David Thornburg, Isaac Asimov, Don Inman, Ramon Zamora, Robert Jastrow, Mac Oglesby, Adam Osborne – the list of authors reads like a Who's Who of microcomputing. These and many other authors contributed some of the finest articles about computers and now-classic games to the pages of *Recreational Computing*.

Last fall, *Recreational Computing* was merged into **COMPUTE!** and we are now offering available back issues. Whatever your interest, you'll find something here – from Spanish BASIC to Computers in Sports Medicine, from Future Fantasy Games to Robot Pets.

**September 1974** A Practical, Low-cost Home/School Microprocessor System, The Computer Illiteracy Problem, Eight Games In BASIC

**March 1975** Build Your Own BASIC, The Computer In Art, Biorhythms

**March/April 1976** A TTY Game, Games With The Pocket Calculator, Dodgem, Square, Tiny BASIC To Go

**July 1976** BASIC Music, Tiny Trek For Altair, 16 Bit Computer Kit, Musical Numbers Guessing Game, Programmer's Toolbox

**September/October 1976** Computer Games In The Classroom, Planets Game, Dungeons And Dragons, Hats Game, Pythagoras And Rational Music

**November/December 1976** Story, Snake, Packl, Frogs Games, Make Believe Computers, The First West Coast Computer Faire, Subroutines, The First Computer

**January/February 1977** Robot Pets, Computers And Space, Tiny Languages, Teaching Using Conversational Programming, High School Computers, Reverse, Tiny PILOT, Mastermind

**March/April 1977** Z-80 PILOT, 6502 Assembly Programming, Tiny BASIC For Beginners, Math Drills & Games, Community Information Systems, Mine, Sales Simulation, Native American Board Games

**July/August 1977** Do-it-yourself CAI, Pet Robots: New Capabilities, PILOT, CAI In BASIC, Programming The HP-25, Capture, Inverse Reverse, 8080 Matrix Subroutines, Women And Computers

**September/October 1977** The $595 PET, More Tiny Languages, Computer Networks, The Bead Game, Biofeedback And Microcomputers Part 1, Home Energy Management, Sandpile Game, A BASIC PILOT

**January/February 1978** Pascal Vs. BASIC, COMAL: Structured BASIC, Video Disks: Magic Lamps for Educators?, A Computer Revolution?, Pounce, The Mechanics of Robots, TRS-80: A Status Report

**March/April 1978** Epic Computer Games, Micros for the Handicapped, Buckets Game, Prayer Wheel Program, Computer Contagion, Measuring Time, Frog Race, The IBM 370 Model 69

**July/August 1978** Computer Whiz Kids, Public Access To Computers, Man-made Minds, Post-human Intelligence, A Modern-day Medicine Show, Live Wire Design, ASCII Graphix, Baseball, Concentration, Gambler's Paradox

**September/October 1978** Kingdom Game, Computers and Museums, Sorcerer of Exidy, Snooping With Your PET, APL, Decimals in Tiny BASIC, Apple Math, TRS-80 Level II: A Grown-up Field Evaluation

**November/December 1978** APL Games, The Return of the Dragons, Animated Games for TRS-80, Runequest, All In The Mind, The L-5 Society, Phantnum, Some Guidelines for Microcomputer Chess, Dataman

**January/February 1979** A Jules Verne Fantasy, Artificial Intelligence, The Apple Corps is With Us, TRS-80 Personal Software, Vending Machine Gets "Brain," Apple II I/O, The Memory Game, REINO: Spanish Kingdom

**March/April 1979** Calculator Comics, "Lord of the Rings," Chess Reconsidered, Database, Beastiary, Color Your Own Graphics, Universe, Easy POKEing with Applesoft BASIC, Air Raid, TRS-80 3-D Plots, Slot, Apple Rose

**May/June 1979** PILOT for Apple II, The Game of Life, Gold Handicapping, Hunt, BASIC vs. Pascal, Inspector Clew-so, Flash for SOL, Faster Jumble, Concept Sans Computer, A Beginner's Guide To FRP

**July/August 1979** Summer Fun, Fooling Around With Your PET, Cryptarithms, Baseball, Newett Awl's Goat, Zork: A Computerized Fantasy Simulation Game, What Light on Yonder Panel Flashes, The Dedicated Word Processor, The FORTE Music Programming Language

**September/October 1979** TRS-80: Outside Connection, The Architecture of Multi-Player Games, The Sounds of Texas Instruments, Dynamic Color Graphics on the New Atari, An Apple PILOT, Gandalf, Spanish BASIC, Designing Animal Games, APL Mastermind

**November/December 1979** SHOGI: Games For You To Program, Atari Sounds, Texas Instrument Graphics and Animation, Interrupt, Match Me, Calendar, Making Music on the PET, Tower of Hanoi, Bingo, Animal Games

**January/February 1980** Computing and Holistic Health, TI Graphics and Animation Part 2, Games To Program, New Directions in Numerical Computing, An Extended BASIC "IF" Facility, Beating Computer Anxiety, Capture for PET, 8080 Tic Tac Toe, Chainwalk, Programming Problems

**March/April 1980** Special Games Issue: Recreation Apple II Hi-res Graphics, Delicious Functions, Galaxy II, Fairy Chess, Raging Robots, Program Instruction Builder, Data Retrieval: An Introduction

**May/June 1980** Introduction to Computer Music, CBBS Phone Numbers, 6502 Machine Language, The Electric Phone Book, Number Translation, Sea Search, Apple Animation, Twister Move Generator, DOZO, Shell Game, Home Video Displays, A Proposed Graphics Language

**July/August 1980** Fantasy Games Issue: Write Your Own Computer Fantasy Simulation, Wizard's Castle, On Future Fantasy Games, Wonderful World of Eamon, In Defense of Hackers, Touch Panels and Interactive Graphics

**September/October 1980** Probability Trees: Big Business on the Micro, The Best of People's Computer Company, Computer Analysis of Athletics, Word Search, Computers in Sports Medicine, Wired, Revolution in Typography?, Textrapolation

**November/December 1980** Computerized Voting, Computer-Using Educators, Hot-rod Computers, House of the Future, Yote, DOZO in Pascal, What is Truth?, Sixth Order Magic Squares on a TRS-80

**January/February 1981** The Education Revolution: Computer Games in the Classroom, An Art-producing Turtle, Computer Literacy Resources, Musical Compositions Using Computers, Microcomputers in China, Twenty Questions, The Pirate's Life for Me, Computers and the Volcanic Fallout

**March/April 1981** Space Exploration: Frontiers for You and Your Micro, Voyage To Antares, A Spaceship Simulator, The Computer as Chess Ally, Star Trek – A Dialogue Approach, Mark of Breeding (fiction), The Fifteen Puzzle

**May/June 1981** Using Computers at Sesame Place, Atari PILOT and Turtle Graphics, Computer Anatomy for Beginners, The Impact of Micros, Nevada-style 8-spot Keno, Sketch Pad, Sum of the Digits, TRS-80 Property Management Program, The Pocket Corner

**July/August 1981** Which Computer Should You Buy?, Commodore's New Rainbow Machine, The Wired Nation: Do We Want It?, Computers at the Junior Museum, 3-D Tic Tac Toe for PET, Number Crossword for all Computers

**September/October 1981** 43 Ways To Make Money With Your Micro, How To Start A Software Exchange, Who Are Computer Criminals?, Micros Behind Bars, Number Systems, Computer Knock-knock Jokes in BASIC and LISP, For Photographers Only, Fibonacci Nim, Roman Numeral Conversum Programum

Special *Recreational Computing* Back Issue Pricing:

| | | | |
|---|---|---|---|
| **Single Issue:** | **$ 3.00** | **Any Fifteen Issues:** | **$25.00** |
| **Any Five Issues:** | **$10.00** | **Any Twenty Issues:** | **$30.00** |
| **Any Ten Issues:** | **$20.00** | | |

For Fastest Service, Call Toll Free **800-334-0868**
In NC Call **919-275-9809**.

Or Send Order and Payment to **COMPUTE!** Publications, P.O. Box 5406, Greensboro, NC 27403.

In the US, please include $.20 per issue ordered for shipping and handling. Outside the US, please include $.30 per issue for surface mail. Orders must be prepaid in US funds or international money order. All orders subject to availability.

whether the effects will be good or bad. When feeling optimistic, they expect (among other effects) that computers will promote active, creative and individualized learning, and provide an antidote to many of the ill effects of television. When feeling pessimistic, they fear that the prevalence of computers in society may result in "more resistance to learning and more truancy from schools which cannot provide the immediate excitement of computer games."

### Subculture Of Educational Computing

I share the view that computers have the potential for both positive and negative effects upon schools. However, I strongly disagree with the authors when they write: "Unfortunately, there appears to be little we, as educators, can do about such computer fallout in the schools except to prepare ourselves and our students for the possibility of it occurring." I believe there is much to be done by all of us who are concerned with education. We can make good use of the educational potential of computers and prepare students for a more computerized society. We can develop high quality software or help others do so. We can use computers to help in mainstreaming handicapped students. The authors of *Practical Guide to Computers in Education* seem to assume that educators must play a passive role in determining the future of schools.

The rest of Chapter 1 introduces some key terms and presents "a brief tour of the subculture of educational computing." The tour consists of descriptions of real teachers and students using computers in various ways. These vignettes provide valuable insights into the possible benefits and agonies of using computers. Similar vignettes appear in other chapters and comprise one of the strong features of this book.

Chapter 2 covers the myriad ways computers can be used in education. It begins with Computer Assisted Instruction, which is subdivided into drill and practice, tutorials, demonstrations, simulations and instructional games. It then goes on to Instruction/Learning Tools, subdivided into word processing, numerical analysis (e.g., VisiCalc), data processing (including accessing information from data bases such as the Source), instrument monitoring devices (for using computers in science labs), graphics and sound. The next section discusses Computer Managed Instruction – systems designed to measure and keep track of student performance and automatically present a sequence of lessons. Then the merits of teaching programming and computer literacy are discussed. I would have liked to see more information about graphics and programming, but overall this is a solid chapter with a great deal of useful information.

Chapter 3 is on "Bits and Bytes." It introduces many of the key terms and describes the parts of a computer system. However, it does not attempt to convey much understanding of how computers work and how they can be programmed to perform such a variety of functions.

Chapter 4 covers "Choosing Your Computer System." The authors take an approach with which I agree completely. They advise you to start by deciding how you want to use computers and checking into the availability of suitable software. Only then can you select a computer system that will meet your needs. They discuss five categories of computer systems: hand-held computers, cassette based computers, floppy disk based computers, hard disk based computers and resource-sharing networks. I noticed only one practical point that was neglected. Much software is available only on floppy disks and is protected so it cannot be copied. This often creates problems for people with hard disk systems – they may be unable to transfer desired software from a floppy disk to their hard disk.

Chapter 5, "Choosing Educational Software," starts out warning about the lack of quality software. Although I have seen my share of atrocious software, I feel that the authors are more negative than is warranted. The software available now is far superior to that available just a year or two ago, and I expect it will continue to improve.

Once past the negative introduction, this is an excellent chapter. It presents a detailed set of guidelines for evaluating software. The guidelines cover program content, pedagogy, program operation and student outcomes. More specific questions to consider are given under each category and examples of existing programs are described. I noticed only one gap worth filling. The authors discuss the important area of what type of feedback the computer provides to the students. However, they limit their discussion to right/wrong and reinforcing feedback, neglecting to discuss the value of remedial feedback – feedback which helps the student understand and correct his mistakes, not just know whether he was right or wrong.

Chapters 6 and 7, "Introducing Computers into the School" and "Integrating Computers into the School," will be very useful to many educators. Sections cover acquiring computers, funding, preparing teachers and administrators, locating computers in the school (e.g., spread among classrooms or in one central room), and providing the basic information needed in a computer facility. These chapters are based upon the experiences of many educators and should be read by all who are introducing computers into schools.

The final chapter is entitled "Issues and